

**O‘ZBEKISTON RESPUBLIKASI
MAKTABGACHA VA MAKTAB TA’LIMI VAZIRLIGI
NAMANGAN DAVLAT PEDAGOGIKA INSTITUTI
“Intellectual fanlar va axborot texnologiyalari” kafedrası**

ZAMONAVIY DASTURLASH TILLARI

fanidan

O‘QUV-USLUBIY MAJMUA

Kurs: Informatika fanini o‘qitish huquqini berish bo‘yicha
kasbiy va pedagogik qayta tayyorlash

Ta’lim dasturi: 10011013

(ishlab chiqarishdan ajralgan va ajralmagan holda)

Umumiy yuklama: 150 soat (ma’ruza – 40, amaliy – 70, laboratoriya – 40)

Namangan – 2026

MUNDARIJA

SO‘Z BOSHI	5
I. FANNING O‘QUV DASTURI.....	7
II. ISHCHI O‘QUV DASTURI	12
III. FANNI O‘QITISHDA FOYDALANILADIGAN INTERFAOL TA’LIM METODLARI	19
IV. NAZARIY MATERIALLAR (MA’RUZA MATNLARI)	23
1-MAVZU. DASTURLASH TILLARI TASNIFI VA RIVOJLANISH TARIXI. KOMPILYATSIYA VA INTERPRETATSIYA. ZAMONAVIY DASTURLASH TILLARI (PYTHON, C++, JAVA, JAVASCRIPT)NING QIYOSIY TAHLILI	23
2-MAVZU. PYTHON DASTURLASH MUHITI: O‘RNATISH, IDE (VS CODE, PYCHARM, THONNY), JUPYTER NOTEBOOK. BIRINCHI DASTUR.....	26
3-MAVZU. O‘ZGARUVCHILAR, MA’LUMOT TURLARI, OPERATORLAR VA IFODALAR. KIRITISH VA CHIQRARISH.....	28
4-MAVZU. SHARTLI OPERATORLAR: IF, ELIF, ELSE, MATCH. MANTIQUIY IFODALAR	31
5-MAVZU. TAKRORLASH OPERATORLARI: FOR, WHILE, BREAK, CONTINUE ...	33
6-MAVZU. SATRLAR VA ULAR USTIDA AMALLAR. SATR METODLARI, FORMATLASH (F-STRING).....	36
7-MAVZU. RO‘YXATLAR VA KORTEJLAR. RO‘YXAT METODLARI, KESIMLAR, LIST COMPREHENSION	38
8-MAVZU. LUG‘ATLAR VA TO‘PLAMLAR	41
9-MAVZU. FUNKSIYALAR: PARAMETRLAR, QAYTARILADIGAN QIYMAT, KO‘RINISH SOHASI, LAMBDA FUNKSIYALAR. REKURSIYA	43
10-MAVZU. MODULLAR VA PAKETLAR. STANDART KUTUBXONA (MATH, RANDOM, DATETIME). PIP PAKET MENEJERI	47
11-MAVZU. FAYLLAR BILAN ISHLASH. CSV VA JSON FORMATLARI. ISTISNOLARNI QAYTA ISHLASH (TRY/EXCEPT)	49
12-MAVZU. OB‘YEKTGA YO‘NALTIRILGAN DASTURLASH: SINFLAR, OB‘YEKTLAR, ATRIBUTLAR VA METODLAR.....	52
13-MAVZU. OYD TAMOYILLARI: INKAPSULYATSIYA, VORISLIK, POLIMORFIZM	55
14-MAVZU. ALGORITMLAR MURAKKABLIGI (O-NOTATSIYA). SARALASH VA QIDIRISH ALGORITMLARI.....	58
15-MAVZU. MA’LUMOTLAR TUZILMALARI: STEK, NAVBAT, BOG‘LANGAN RO‘YXAT, DARAXTLAR VA GRAFLAR ASOSLARI.....	62
16-MAVZU. C++ DASTURLASH TILIGA KIRISH: DASTUR TUZILISHI, MA’LUMOT TURLARI, BOSHQARUV OPERATORLARI.....	65
17-MAVZU. C++ DA MASSIVLAR, FUNKSIYALAR VA STL KONTEYNERLARI (VECTOR, MAP). OLIMPIADA MASALALARINI YECHISHDA C++	68
18-MAVZU. GRAFIK INTERFEYSLI DASTURLAR: TKINTER KUTUBXONASI. TURTLE GRAFIKASI ORQALI DASTURLASHNI O‘QITISH.....	71
19-MAVZU. MA’LUMOTLARNI QAYTA ISHLASH VA VIZUALLASHTIRISH: NUMPY, PANDAS VA MATPLOTLIB ASOSLARI	74

20-MAVZU. DASTURIY TA'MINOT ISHLAB CHIQISH TEXNOLOGIYALARI: GIT VA GITHUB, TESTLASH, KOD SIFATI. SUN'IY INTELLEKT YORDAMCHILARIDAN DASTURLASHDA FOYDALANISH	77
V. AMALIY MASHG'ULOT MATERIALLARI.....	81
1-AMALIY MASHG'ULOT. PYTHON VA VS CODE/THONNY MUHITINI O'RNATISH, BIRINCHI DASTURNI YOZISH	81
2-AMALIY MASHG'ULOT. O'ZGARUVCHILAR, MA'LUMOT TURLARI VA ARIFMETIK AMALLARGA DOIR MASALALAR.....	82
3-AMALIY MASHG'ULOT. KIRITISH-CHIQUVCHILAR VA TURLARNI O'ZGARTIRISHGA DOIR MASALALAR	84
4-AMALIY MASHG'ULOT. SHARTLI OPERATORLARGA DOIR MASALALAR	85
5-AMALIY MASHG'ULOT. MURAKKAB SHARTLAR VA ICHMA-ICH SHARTLI OPERATORLAR	86
6-AMALIY MASHG'ULOT. FOR SIKLI: YIG'INDI, KO'PAYTMA VA SANASH MASALALARI	87
7-AMALIY MASHG'ULOT. WHILE SIKLI VA ITERATSION JARAYONLAR.....	88
8-AMALIY MASHG'ULOT. ICHMA-ICH SIKLLAR VA NAQSHLAR CHIZISHGA DOIR MASALALAR.....	89
9-AMALIY MASHG'ULOT. SONLAR NAZARIYASIGA DOIR MASALALAR: TUB SONLAR, EKUB, EKUK, RAQAMLAR YIG'INDISI	91
10-AMALIY MASHG'ULOT. SATRLAR USTIDA AMALLAR VA SATR METODLARI.....	92
11-AMALIY MASHG'ULOT. SATRLARNI QAYTA ISHLASHGA DOIR MASALALAR: PALINDROM, SO'ZLARNI SANASH.....	93
12-AMALIY MASHG'ULOT. RO'YXATLAR BILAN ISHLASH: YARATISH, QO'SHISH, O'CHIRISH, KESIMLAR	94
13-AMALIY MASHG'ULOT. RO'YXATLARDA QIDIRISH, ENG KATTA VA ENG KICHIK ELEMENTNI TOPISH, SARALASH.....	95
14-AMALIY MASHG'ULOT. LIST COMPREHENSION VA IKKI O'LCHOVLI RO'YXATLAR (MATRITSALAR)	97
15-AMALIY MASHG'ULOT. KORTEJLAR VA TO'PLAMLARGA DOIR MASALALAR.....	98
16-AMALIY MASHG'ULOT. LUG'ATLAR BILAN ISHLASH: CHASTOTA LUG'ATI, TELEFON KITABI	99
17-AMALIY MASHG'ULOT. FUNKSIYALAR YARATISH VA CHAQIRISH.....	100
18-AMALIY MASHG'ULOT. REKURSIV FUNKSIYALAR: FAKTORIAL, FIBONACHCHI SONLARI, XANOY MINORASI.....	101
19-AMALIY MASHG'ULOT. STANDART KUTUBXONA MODULLARIDAN FOYDALANISH (MATH, RANDOM, DATETIME)	102
20-AMALIY MASHG'ULOT. FAYLLARGA YOZISH VA FAYLLARDAN O'QISH ..	104
21-AMALIY MASHG'ULOT. CSV VA JSON FAYLLARI BILAN ISHLASH.....	105
22-AMALIY MASHG'ULOT. ISTISNOLARNI QAYTA ISHLASH. KIRITILGAN MA'LUMOTLARNI TEKSHIRISH	106
23-AMALIY MASHG'ULOT. SINFLAR VA OB'YEKTLAR YARATISH.....	107
24-AMALIY MASHG'ULOT. VORISLIK VA POLIMORFIZMGA DOIR MASALALAR.....	108

25-AMALIY MASHG‘ULOT. PUFAKCHALI, TANLASH VA QO‘SHISH USULIDA SARALASH ALGORITMLARINI DASTURLASH	110
26-AMALIY MASHG‘ULOT. CHIZIQLI VA IKKILIK QIDIRUV ALGORITMLARI ..	111
27-AMALIY MASHG‘ULOT. STEK VA NAVBATNI DASTURLASH.....	112
28-AMALIY MASHG‘ULOT. GRAFLARDA KENGLIK VA CHUQURLIK BO‘YICHA QIDIRUV (BFS, DFS) ASOSLARI	113
29-AMALIY MASHG‘ULOT. C++ DA BIRINCHI DASTUR, KIRITISH-CHIQUARISH VA SHARTLI OPERATORLAR.....	115
30-AMALIY MASHG‘ULOT. C++ DA SIKLLAR VA MASSIVLAR	116
31-AMALIY MASHG‘ULOT. C++ DA FUNKSIYALAR VA STL (VECTOR, SORT)..	117
32-AMALIY MASHG‘ULOT. TURTLE KUTUBXONASI YORDAMIDA GRAFIK DASTURLAR YARATISH	118
33-AMALIY MASHG‘ULOT. TKINTER YORDAMIDA ODDIY GRAFIK INTERFEYSLI DASTUR (KALKULYATOR) YARATISH	120
34-AMALIY MASHG‘ULOT. PANDAS VA MATPLOTLIB YORDAMIDA O‘QUVCHILAR BAHOLARI JADVALINI TAHLIL QILISH	121
35-AMALIY MASHG‘ULOT. GIT VA GITHUB BILAN ISHLASH: REPOZITORIY YARATISH, COMMIT, PUSH.....	122
VI. LABORATORIYA MASHG‘ULOTLARI MATERIALLARI.....	124
1-LABORATORIYA ISHI. CHIZIQLI VA TARMOQLANUVCHI ALGORITMLARGA DOIR DASTURLAR MAJMUASINI YARATISH	124
2-LABORATORIYA ISHI. TAKRORLANUVCHI JARAYONLARGA DOIR DASTURLAR MAJMUASINI YARATISH	125
3-LABORATORIYA ISHI. RANDOM MODULI YORDAMIDA “SONNI TOP” O‘YININI YARATISH	126
4-LABORATORIYA ISHI. MATN STATISTIKASINI HISOBLOVCHI DASTUR YARATISH	127
5-LABORATORIYA ISHI. MATRITSALAR USTIDA AMALLAR BAJARUVCHI DASTUR YARATISH	127
6-LABORATORIYA ISHI. LUG‘ATLAR ASOSIDA “LUG‘AT-TARJIMON” DASTURINI YARATISH.....	128
7-LABORATORIYA ISHI. FUNKSIYALAR KUTUBXONASINI (MODUL) YARATISH VA UNDAN FOYDALANISH	129
8-LABORATORIYA ISHI. REKURSIV ALGORITMLAR: KOMBINATORIKA MASALALARINI DASTURLASH.....	130
9-LABORATORIYA ISHI. FAYL ASOSIDAGI “O‘QUVCHILAR RO‘YXATI” DASTURINI YARATISH (QO‘SHISH, QIDIRISH, SARALASH).....	131
10-LABORATORIYA ISHI. JSON ASOSIDA TEST (VIKTORINA) DASTURINI YARATISH	131
11-LABORATORIYA ISHI. OYD ASOSIDA “KUTUBXONA” TIZIMI MODELINI YARATISH	132
12-LABORATORIYA ISHI. SARALASH ALGORITMLARINING SAMARADORLIGINI TAJRIBADA SOLISHTIRISH	133
13-LABORATORIYA ISHI. MA’LUMOTLAR TUZILMALARIDAN FOYDALANIB MASALALAR YECHISH	134
14-LABORATORIYA ISHI. ONLAYN MASALALAR TIZIMLARIDA (CODEFORCES VA BOSHQALAR) OLIMPIADA MASALALARINI YECHISH.....	135

15-LABORATORIYA ISHI. C++ TILIDA OLIMPIADA TIPIDAGI MASALALARNI YECHISH	135
16-LABORATORIYA ISHI. TURTLE YORDAMIDA FRAKTALLAR VA GEOMETRIK NAQSHLAR CHIZISH	136
17-LABORATORIYA ISHI. TKINTER DA GRAFIK INTERFEYSLI O‘QUV DASTURINI YARATISH.....	137
18-LABORATORIYA ISHI. MA’LUMOTLARNI VIZUALLASHTIRISH: DIAGRAMMALAR QURISH	138
19-LABORATORIYA ISHI. KODNI TESTLASH (UNITTEST) VA XATOLARNI TUZATISH (DEBUGGING).....	139
20-LABORATORIYA ISHI. YAKUNIY LOYIHA: MAKTAB UCHUN FOYDALI DASTUR (TEST TIZIMI, DARS JADVALI GENERATORI VA H.K.) YARATISH, GITHUB GA JOYLASHTIRISH VA HIMOYA QILISH.....	140
VII. MUSTAQIL TA’LIM MASHG‘ULOTLARI.....	142
VIII. KEYSLAR BANKI	147
IX. GLOSSARIY	152
X. TEST SAVOLLARI	158
XI. YAKUNIY NAZORAT (YOZMA ISH) MATERIALLARI VA BAHOLASH MEZONLARI	181
XII. ADABIYOTLAR RO‘YXATI.....	190

SO‘Z BOSHI

Mazkur o‘quv-uslubiy majmua Namangan davlat pedagogika institutida tashkil etilgan 10011013 – “Informatika fanini o‘qitish huquqini berish bo‘yicha kasbiy va pedagogik qayta tayyorlash” kursi tinglovchilari uchun “Zamonaviy dasturlash tillari” fanining o‘quv dasturi asosida tayyorlangan. Majmua fanning maqsad va vazifalari, ta’lim natijalari (TN1–TN10), ma’ruza, amaliy va laboratoriya mashg‘ulotlari mazmuni hamda mustaqil ta’lim topshiriqlarini to‘liq qamrab oladi.

Fan bo‘yicha jami auditoriya yuklamasi 150 soatni tashkil etadi: ma’ruza mashg‘ulotlari – 40 soat (20 ta mavzu), amaliy mashg‘ulotlar – 70 soat (35 ta mashg‘ulot), laboratoriya mashg‘ulotlari – 40 soat (20 ta mashg‘ulot). Mashg‘ulotlar ketma-ketligi tinglovchini dasturlash tillari tasnifi va Python muhitidan boshlab, tilning asosiy konstruksiyalari (o‘zgaruvchilar, shartlar, sikllar, satrlar, ro‘yxatlar, lug‘atlar, funksiyalar), modullar va fayllar, ob‘yektga yo‘naltirilgan dasturlash, algoritmlar va ma’lumotlar tuzilmalari, C++ asoslari, grafik interfeys, ma’lumotlar tahlili hamda Git va dasturiy ta’minot ishlab chiqish texnologiyalariga izchil olib boradi. Har bir mavzuda dasturlash bilan birga uni maktabda o‘qitish metodikasi, o‘quvchilarning tipik xatolari va olimpiada tipidagi masalalar bilan ishlash masalalari ham yoritilgan.

Majmua quyidagi qismlardan iborat:

- fanning o‘quv dasturi;
- ishchi o‘quv dasturi (mavzular va soatlar taqsimoti, nazorat shakllari);
- fanni o‘qitishda foydalaniladigan interfaol ta’lim metodlari;
- nazariy materiallar (M1–M20 ma’ruza matnlari, reja, tayanch iboralar, kod namunalari, nazorat savollari);
- amaliy mashg‘ulot materiallari (A1–A35: maqsad, bajarish tartibi, namunaviy kodlar, variantli topshiriqlar, nazorat savollari);
- laboratoriya mashg‘ulotlari materiallari (L1–L20: dasturiy loyihalar bo‘yicha qadamma-qadam yo‘riqnomalar);
- mustaqil ta’lim topshiriqlari (variantlar, kooperativ topshiriqlar, baholash mezonlari);
- keyslar banki;
- glossariy (o‘zbek, ingliz, rus tillarida atamalar va izohlar);
- test savollari va javoblar kaliti;
- yakuniy nazorat (yozma ish) materiallari va baholash mezonlari;
- foydalanilgan adabiyotlar va axborot manbalari.

Majmuadagi kod namunalari Python 3 va C++ tillarida berilgan va bepul vositalarda (Thonny, VS Code, Jupyter Notebook, onlayn kompilyatorlar) bajarishga mo‘ljallangan. Kutubxonalar (Tkinter, NumPy, pandas, matplotlib)

versiyalari va tashqi platformalar (GitHub, Codeforces) interfeysi o'zgarib turishi mumkinligi sababli, tinglovchi joriy hujjatlarga (docs.python.org, docs.github.com) murojaat qilishi tavsiya etiladi.

Majmuadan informatika o'qituvchilari, qayta tayyorlash kursi tinglovchilari hamda "Matematika va informatika" ta'lim yo'nalishi talabalari foydalanishlari mumkin.

Tuzuvchi: N. Turdaliyeva – NamDPI "Intellectual fanlar va axborot texnologiyalari" kafedrası o'qituvchisi

Taqrizchilar: Sh. Boltabayev – NamDU "Raqamli ta'lim texnologiyalar" kafedrası dotsenti

Sh. Jurayev – NamDU "Raqamli ta'lim texnologiyalar" kafedrası dotsenti

Mazkur o'quv-uslubiy majmua "Intellectual fanlar va axborot texnologiyalari" kafedrasining 2026-yil "____" _____dagi ____-sonli yig'ilishida muhokama qilingan va Aniq va tabiiy fanlar fakulteti Kengashining 2026-yil "____" _____dagi ____-sonli yig'ilishida ma'qullangan.

I. FANNING O‘QUV DASTURI

1. Fan ma’lumotlari

Ko‘rsatkich	Qiymat	Ko‘rsatkich	Qiymat
Fan/modul kodi	ZDT	O‘quv yili	2026–2027
Fan/modul turi	Ixtisoslik fanlari	Ta’lim tili	O‘zbek
O‘quv rejadagi tartib raqami	10011013 – 4.01	O‘qitish oylari	1–9
Fanning nomi	Zamonaviy dasturlash tillari	Jami auditoriya yuklamasi	150 soat
Ma’ruza / Amaliy / Lab.	40 / 70 / 40	Mustaqil ta’lim	6 ta topshiriq

2. Fanning mazmuni

Fanning maqsadi – tinglovchilarga zamonaviy dasturlash tillari (asosan Python, shuningdek C++) asosida algoritmik fikrlash, dasturlash, ob’yektga yo‘naltirilgan dasturlash, algoritmlar va ma’lumotlar tuzilmalari bo‘yicha bilim va ko‘nikmalarni berish hamda ularni maktab informatika kursida dasturlashni o‘qitishga tayyorlashdan iborat.

Fanning vazifalari – dasturlash tillari tasnifi va dasturlash muhitlarini o‘rganish; Python tilining asosiy konstruksiyalari, ma’lumot turlari, funksiyalar, modullar va fayllar bilan ishlashni o‘zlashtirish; ob’yektga yo‘naltirilgan dasturlash tamoyillarini o‘rgatish; saralash, qidirish algoritmlari va ma’lumotlar tuzilmalarini dasturlash; C++ asoslari, grafik interfeys, ma’lumotlar tahlili va Git vositalari bilan tanishtirish; olimpiada tipidagi masalalarni yechish ko‘nikmasini shakllantirish.

3. Fanni o‘zlashtirish uchun zarur boshlang‘ich bilimlar

- Informatika va raqamli texnologiyalar;
- Umumiy o‘rta ta’lim informatika kursi;
- Matematika asoslari.

4. Ta’lim natijalari (TN)

Kod	Ta’lim natijasi
	Kasbiy kompetensiyalar:
TN1	dasturlash tillari tasnifi, kompilyatsiya va interpretatsiya, zamonaviy dasturlash muhitlari haqida bilimga ega bo‘lishi kerak;
TN2	Python tilining sintaksisi, ma’lumot turlari, boshqaruv operatorlari va funksiyalarini bilishi kerak;
TN3	satrlar, ro‘yxatlar, lug‘atlar va to‘plamlar bilan ishlash usullarini bilishi kerak;

Kod	Ta'lim natijasi
TN4	ob'yektga yo'naltirilgan dasturlash tamoyillari (inkapsulyatsiya, vorislik, polimorfizm)ni tushunishi kerak;
TN5	asosiy algoritmlar (saralash, qidirish, rekursiya), ularning murakkabligi va ma'lumotlar tuzilmalarini bilishi kerak;
TN6	C++ tilining asoslari va dasturiy ta'minot ishlab chiqish vositalari (Git, testlash)ni tushunishi kerak;
	Ko'nikmalar:
TN7	Python tilida turli murakkablikdagi masalalarni yechish uchun dasturlar yozish ko'nikmasiga ega bo'ladi;
TN8	fayllar, modullar, istisnolar va OYD vositalaridan foydalanib amaliy dasturlar yaratish ko'nikmasiga ega bo'ladi;
TN9	olimpiada tipidagi masalalarni Python va C++ tillarida yechish ko'nikmasiga ega bo'ladi;
TN10	maktab uchun foydali dasturiy loyihani rejalashtirish, yaratish, GitHub ga joylashtirish va taqdimot qilish bo'yicha amaliy tajribaga ega bo'ladi.

5. Fan mazmuni va mashg'ulotlar shakli

№	Mavzu va uning mazmuni	Soat
	Ma'ruza (M)	40
M1	Dasturlash tillari tasnifi va rivojlanish tarixi. Kompilyatsiya va interpretatsiya. Zamonaviy dasturlash tillari (Python, C++, Java, JavaScript)ning qiyosiy tahlili [1,3]	2
M2	Python dasturlash muhiti: o'rnatish, IDE (VS Code, PyCharm, Thonny), Jupyter Notebook. Birinchi dastur [1,2,9]	2
M3	O'zgaruvchilar, ma'lumot turlari, operatorlar va ifodalar. Kiritish va chiqarish [1,2]	2
M4	Shartli operatorlar: if, elif, else, match. Mantiqiy ifodalar [2,4]	2
M5	Takrorlash operatorlari: for, while, break, continue [2,4]	2
M6	Satrlar va ular ustida amallar. Satr metodlari, formatlash (f-string) [1,2]	2
M7	Ro'yxatlar va kortejlar. Ro'yxat metodlari, kesimlar, list comprehension [1,4]	2
M8	Lug'atlar va to'plamlar [1,4]	2
M9	Funksiyalar: parametrlar, qaytariladigan qiymat, ko'rinish sohasi, lambda funksiyalar. Rekursiya [2,4,15]	2
M10	Modullar va paketlar. Standart kutubxona (math, random, datetime). pip paket menejeri [1,9]	2
M11	Fayllar bilan ishlash. CSV va JSON formatlari. Istisnolarni qayta ishlash (try/except) [1,2]	2
M12	Ob'yektga yo'naltirilgan dasturlash: sinflar, ob'yektlar, atributlar va metodlar [1,5]	2
M13	OYD tamoyillari: inkapsulyatsiya, vorislik, polimorfizm [1,5]	2

№	Mavzu va uning mazmuni	Soat
M14	Algoritmlar murakkabligi (O-notatsiya). Saralash va qidirish algoritmlari [6,12]	2
M15	Ma'lumotlar tuzilmalari: stek, navbat, bog'langan ro'yxat, daraxtlar va graflar asoslari [6,12]	2
M16	C++ dasturlash tiliga kirish: dastur tuzilishi, ma'lumot turlari, boshqaruv operatorlari [7,10]	2
M17	C++ da massivlar, funksiyalar va STL konteynerlari (vector, map). Olimpiada masalalarini yechishda C++ [7,10]	2
M18	Grafik interfeysli dasturlar: Tkinter kutubxonasi. Turtle grafikasi orqali dasturlashni o'qitish [1,9]	2
M19	Ma'lumotlarni qayta ishlash va vizuallashtirish: NumPy, pandas va matplotlib asoslari [8,9]	2
M20	Dasturiy ta'minot ishlab chiqish texnologiyalari: Git va GitHub, testlash, kod sifati. Sun'iy intellekt yordamchilaridan dasturlashda foydalanish [11,14]	2
	Amaliy mashg'ulot (A)	70
A1	Python va VS Code/Thonny muhitini o'rnatish, birinchi dasturni yozish [2,9]	2
A2	O'zgaruvchilar, ma'lumot turlari va arifmetik amallarga doir masalalar [2,4]	2
A3	Kiritish-chiqarish va turlarni o'zgartirishga doir masalalar [2]	2
A4	Shartli operatorlarga doir masalalar [2,4]	2
A5	Murakkab shartlar va ichma-ich shartli operatorlar [2,4]	2
A6	for sikli: yig'indi, ko'paytma va sanash masalalari [2,4]	2
A7	while sikli va iteratsion jarayonlar [2,4]	2
A8	Ichma-ich sikllar va naqshlar chizishga doir masalalar [2,4]	2
A9	Sonlar nazariyasiga doir masalalar: tub sonlar, EKUB, EKUK, raqamlar yig'indisi [4,6]	2
A10	Satrlar ustida amallar va satr metodlari [1,2]	2
A11	Satrlarni qayta ishlashga doir masalalar: palindrom, so'zlarni sanash [2,4]	2
A12	Ro'yxatlar bilan ishlash: yaratish, qo'shish, o'chirish, kesimlar [1,4]	2
A13	Ro'yxatlarda qidirish, eng katta va eng kichik elementni topish, saralash [4,6]	2
A14	List comprehension va ikki o'lchovli ro'yxatlar (matritsalar) [1,4]	2
A15	Kortejlar va to'plamlarga doir masalalar [1,4]	2
A16	Lug'atlar bilan ishlash: chastota lug'ati, telefon kitobi [1,4]	2
A17	Funksiyalar yaratish va chaqirish [2,4]	2
A18	Rekursiv funksiyalar: faktorial, Fibonachchi sonlari, Xanoy minorasi [4,6]	2
A19	Standart kutubxona modullaridan foydalanish (math, random, datetime) [1,9]	2
A20	Fayllarga yozish va fayllardan o'qish [1,2]	2

№	Mavzu va uning mazmuni	Soat
A21	CSV va JSON fayllari bilan ishlash [1,9]	2
A22	Istisnolarni qayta ishlash. Kiritilgan ma'lumotlarni tekshirish [1,2]	2
A23	Sinflar va ob'yektlar yaratish [1,5]	2
A24	Vorislik va polimorfizmga doir masalalar [1,5]	2
A25	Pufakchali, tanlash va qo'shish usulida saralash algoritmlarini dasturlash [6,12]	2
A26	Chiziqli va ikkilik qidiruv algoritmlari [6,12]	2
A27	Stek va navbatni dasturlash [6,12]	2
A28	Graflarda kenglik va chuqurlik bo'yicha qidiruv (BFS, DFS) asoslari [6,12]	2
A29	C++ da birinchi dastur, kiritish-chiqarish va shartli operatorlar [7,10]	2
A30	C++ da sikllar va massivlar [7,10]	2
A31	C++ da funksiyalar va STL (vector, sort) [7,10]	2
A32	Turtle kutubxonasi yordamida grafik dasturlar yaratish [1,9]	2
A33	Tkinter yordamida oddiy grafik interfeysli dastur (kalkulyator) yaratish [1,9]	2
A34	pandas va matplotlib yordamida o'quvchilar baholari jadvalini tahlil qilish [8,9]	2
A35	Git va GitHub bilan ishlash: repozitoriy yaratish, commit, push [11]	2
	Laboratoriya mashg'uloti (L)	40
L1	Chiziqli va tarmoqlanuvchi algoritmlarga doir dasturlar majmuasini yaratish [2,4]	2
L2	Takrorlanuvchi jarayonlarga doir dasturlar majmuasini yaratish [2,4]	2
L3	random moduli yordamida "Sonni top" o'yinini yaratish [1,9]	2
L4	Matn statistikasini hisoblovchi dastur yaratish [2,4]	2
L5	Matritsalar ustida amallar bajaruvchi dastur yaratish [1,4]	2
L6	Lug'atlar asosida "Lug'at-tarjimon" dasturini yaratish [1,4]	2
L7	Funksiyalar kutubxonasini (modul) yaratish va undan foydalanish [1,9]	2
L8	Rekursiv algoritmlar: kombinatorika masalalarini dasturlash [4,6]	2
L9	Fayl asosidagi "O'quvchilar ro'yxati" dasturini yaratish (qo'shish, qidirish, saralash) [1,2]	2
L10	JSON asosida test (viktorina) dasturini yaratish [1,9]	2
L11	OYD asosida "Kutubxona" tizimi modelini yaratish [1,5]	2
L12	Saralash algoritmlarining samaradorligini tajribada solishtirish [6,12]	2
L13	Ma'lumotlar tuzilmalaridan foydalanib masalalar yechish [6,12]	2
L14	Onlayn masalalar tizimlarida (Codeforces va boshqalar) olimpiada masalalarini yechish [6,13]	2
L15	C++ tilida olimpiada tipidagi masalalarni yechish [7,10]	2
L16	Turtle yordamida fraktallar va geometrik naqshlar chizish [1,9]	2
L17	Tkinter da grafik interfeysli o'quv dasturini yaratish [1,9]	2
L18	Ma'lumotlarni vizuallashtirish: diagrammalar qurish [8,9]	2

№	Mavzu va uning mazmuni	Soat
L19	Kodni testlash (unittest) va xatolarni tuzatish (debugging) [9,11]	2
L20	Yakuniy loyiha: maktab uchun foydali dastur (test tizimi, dars jadvali generatori va h.k.) yaratish, GitHub ga joylashtirish va himoya qilish [1,11,14]	2

6. Mustaqil ta'lim topshiriqlari

№	Topshiriq
1	Zamonaviy dasturlash tillarini qiyosiy tahlil qilish va ularning ta'limdagi o'rni bo'yicha taqdimot tayyorlash. [1,3]
2	Python bo'yicha o'quvchilar uchun darajalangan masalalar to'plamini tuzish va yechimlarini tayyorlash. [2,4]
3	Saralash algoritmlarini vizual tushuntiruvchi o'quv materiali tayyorlash. [6,12]
4	Onlayn masalalar tizimida kamida 20 ta masala yechish va hisobot tayyorlash. [6,13]
5	OYD asosida kichik o'quv loyihasini yaratish (kichik guruhlarda hamkorlikda). [1,5]
6	Python da ma'lumotlar tahlili bo'yicha mini-loyiha bajarish va GitHub ga joylashtirish (kichik guruhlarda hamkorlikda). [8,11]

Izoh: fan (modul) yuzasidan tinglovchilar bajaradigan mustaqil ish topshiriqlari variativ tavsifga ega; mustaqil ish topshiriqlarining 1/3 qismi (5- va 6-topshiriqlar) kichik guruhlarda hamkorlikda ishlashga (kooperativlik) mo'ljallangan.

7. Ta'lim texnologiyalari va metodlari

- ma'ruzalar;
- amaliy va laboratoriya ishlarini bajarish va xulosalash;
- interfaol keys-stadilar;
- blits-so'rov;
- guruhlarda ishlash;
- taqdimotlar qilish;
- jamoa bo'lib ishlash va himoya qilish uchun loyihalar.

8. Kreditlarni olish uchun talablar

Fanga oid nazariy va uslubiy tushunchalarni to'la o'zlashtirish, tahlil natijalarini to'g'ri aks ettira olish, o'rganilayotgan jarayonlar haqida mustaqil mushohada yuritish va nazorat uchun berilgan vazifa va topshiriqlarni bajarish, yakuniy nazorat bo'yicha yozma ishni topshirish.

9. Tinglovchi bilimini baholash mezonlari

Baholash mezonlari ushbu majmuaning XI bo'limida batafsil keltirilgan (5 – “a'lo”, 4 – “yaxshi”, 3 – “qoniqarli”, 2 – “qoniqarsiz”).

10. Foydalanilgan adabiyotlar

Adabiyotlar ro'yxati majmuaning XII bo'limida keltirilgan. Mavzular yonidagi kvadrat qavs ichidagi raqamlar ([1,3]) ushbu ro'yxatdagi tartib raqamlariga mos keladi.

Tuzuvchi: D. Sodirova

Kafedra mudiri: A. Matkarimov

II. ISHCHI O'QUV DASTURI

1. Fanning o'quv yuklamasi taqsimoti

Mashg'ulot turi	Soat	Mashg'ulotlar soni	Nazorat shakli
Ma'ruza	40	20	Blits-so'rov, nazorat savollari
Amaliy mashg'ulot	70	35	Amaliy ishni himoya qilish
Laboratoriya mashg'uloti	40	20	Dasturni namoyish etish va himoya qilish
Jami auditoriya	150	75	–
Mustaqil ta'lim	–	6 ta topshiriq	Loyiha / taqdimot himoyasi
Yakuniy nazorat	–	–	Yozma ish

2. Ma'ruza mashg'ulotlarining mavzuli rejasi

№	Mavzu nomi va asosiy savollari	Soat	Metod	TN
M1	Dasturlash tillari tasnifi va rivojlanish tarixi. Kompilyatsiya va interpretatsiya. Zamonaviy dasturlash tillari (Python, C++, Java, JavaScript)ning qiyosiy tahlili 1. Dasturlash tillari tasnifi 2. Rivojlanish tarixi va avlodlar 3. Kompilyatsiya va interpretatsiya 4. Tillarning qiyosiy tahlili 5. Til tanlash mezonlari 6. Maktabda birinchi til masalasi	2	Klaster, blits-so'rov	TN1
M2	Python dasturlash muhiti: o'rnatish, IDE (VS Code, PyCharm, Thonny), Jupyter Notebook. Birinchi dastur 1. Python tilining xususiyatlari 2. O'rnatish va muhit 3. IDE lar va ularni solishtirish 4. Jupyter Notebook 5. Birinchi dastur va xato xabarlar 6. Maktab uchun muhit tanlash	2	Insert, amaliy namoyish	TN1
M3	O'zgaruvchilar, ma'lumot turlari, operatorlar va ifodalar. Kiritish va chiqarish 1. O'zgaruvchi va nom berish 2. Ma'lumot turlari (int, float, str, bool) 3. Operatorlar va ustuvorlik 4. Turlarni o'zgartirish 5. Kiritish va chiqarish 6. Tipik xatolar	2	Blits-so'rov, mini-mashq	TN2
M4	Shartli operatorlar: if, elif, else, match. Mantiqiy ifodalar 1. Mantiqiy ifodalar va taqqoslash 2. if-elif-else tuzilmasi 3. Ichma-ich shartlar 4. match operatori 5. Chekka holatlarni tekshirish 6. Tipik xatolar	2	Sветофор, mini-mashq	TN2

№	Mavzu nomi va asosiy savollari	Soat	Metod	TN
M5	Takrorlash operatorlari: for, while, break, continue 1. for sikli va range 2. while sikli 3. break va continue 4. Ichma-ich sikllar 5. Cheksiz sikl xavfi 6. Kuzatuv jadvali (trace)	2	Kuzatuv jadvali, mini-mashq	TN2
M6	Satrlar va ular ustida amallar. Satr metodlari, formatlash (f-string) 1. Satr tushunchasi va indekslash 2. Kesimlar 3. Satr metodlari 4. f-string formatlash 5. Satrlarni qayta ishlash masalalari 6. Unicode va kodlash	2	Blits-so'rov, juftlikda ishlash	TN3
M7	Ro'yxatlar va kortejlar. Ro'yxat metodlari, kesimlar, list comprehension 1. Ro'yxat yaratish va indekslash 2. Ro'yxat metodlari 3. Kesimlar 4. list comprehension 5. Kortejlar 6. Ikki o'lchovli ro'yxatlar	2	Insert, juftlikda ishlash	TN3
M8	Lug'atlar va to'plamlar 1. Lug'at tuzilishi va metodlari 2. Chastota lug'atlari 3. To'plamlar va amallar 4. Tuzilma tanlash mezonlari 5. Lug'at va to'plam comprehension 6. Tipik xatolar	2	Klaster, mini-mashq	TN3
M9	Funksiyalar: parametrlar, qaytariladigan qiymat, ko'rinish sohasi, lambda funksiyalar. Rekursiya 1. Funksiya yaratish va chaqirish 2. Parametrlar va qaytariladigan qiymat 3. Ko'rinish sohasi 4. lambda funksiyalar 5. Rekursiya g'oyasi 6. Rekursiya va sikl solishtirish	2	Muammoli ma'ruza, kuzatuv jadvali	TN2
M10	Modullar va paketlar. Standart kutubxona (math, random, datetime). pip paket menejeri 1. Modul va paket tushunchasi 2. import usullari 3. math, random, datetime 4. Virtual muhit 5. pip menejeri 6. O'z modulini yaratish	2	Blits-so'rov, amaliy namoyish	TN2
M11	Fayllar bilan ishlash. CSV va JSON formatlari. Istisnolarni qayta ishlash (try/except) 1. Matn fayllarini o'qish va yozish 2. with operatori 3. CSV formati 4. JSON formati 5. try/except va istisno turlari 6. Kiritilgan ma'lumotni tekshirish	2	Insert, keys-stadi	TN2
M12	Ob'yektga yo'naltirilgan dasturlash: sinflar, ob'yektlar, atributlar va metodlar 1. OYD g'oyasi va afzalliklari 2. Sinf va ob'yekt 3. Atributlar va metodlar 4. __init__ va self 5. Maxsus metodlar 6. Hayotiy misollar bilan modellash	2	Klaster, muammoli ma'ruza	TN4

№	Mavzu nomi va asosiy savollari	Soat	Metod	TN
M13	OYD tamoyillari: inkapsulyatsiya, vorislik, polimorfizm 1. Inkapsulyatsiya 2. Vorislik va super() 3. Polimorfizm 4. Abstraksiya 5. Kompozitsiya va vorislik 6. OYD ni maktabda o'qitish masalalari	2	Keys-stadi, Venn diagrammasi	TN4
M14	Algoritmlar murakkabligi (O-notatsiya). Saralash va qidirish algoritmlari 1. Algoritm samaradorligi 2. O-notatsiya 3. Chiziqli va ikkilik qidiruv 4. Oddiy saralash usullari 5. Tez saralash va birlashtirib saralash 6. Eksperiment bilan solishtirish	2	Muammoli ma'ruza, vizuallashtirish	TN5
M15	Ma'lumotlar tuzilmalari: stek, navbat, bog'langan ro'yxat, daraxtlar va graflar asoslari 1. Stek va navbat 2. Bog'langan ro'yxat 3. Daraxt tuzilmasi 4. Graf va ularni tasvirlash 5. BFS va DFS g'oyasi 6. Tuzilma tanlash mezonlari	2	Klaster, vizuallashtirish	TN5
M16	C++ dasturlash tiliga kirish: dastur tuzilishi, ma'lumot turlari, boshqaruv operatorlari 1. C++ ning xususiyatlari 2. Dastur tuzilishi 3. Statik turlar 4. Kiritish-chiqarish (cin, cout) 5. Shart va sikl operatorlari 6. Python bilan taqqoslash	2	Qiyosiy jadval, mini-mashq	TN6
M17	C++ da massivlar, funksiyalar va STL konteynerlari (vector, map). Olimpiada masalalarini yechishda C++ 1. Massivlar 2. Funksiyalar va parametrlar 3. vector va map 4. sort va algoritmlar 5. Tezkor kiritish-chiqarish 6. Olimpiada masalasi namunasi	2	Muammoli ma'ruza, keys-stadi	TN6
M18	Grafik interfeysli dasturlar: Tkinter kutubxonasi. Turtle grafikasi orqali dasturlashni o'qitish 1. Hodisaga asoslangan dasturlash 2. Tkinter vidjetlari 3. Joylashtirish va hodisalar 4. Turtle grafikasi 5. Turtle bilan o'qitish metodikasi 6. Loyiha namunasi	2	Amaliy namoyish, insert	TN8
M19	Ma'lumotlarni qayta ishlash va vizuallashtirish: NumPy, pandas va matplotlib asoslari 1. NumPy massivlari 2. pandas DataFrame 3. Ma'lumotlarni o'qish va filtrlash 4. matplotlib diagrammalari 5. Ma'lumotlar tahlili jarayoni 6. Maktab uchun misollar	2	Amaliy namoyish, klaster	TN8

№	Mavzu nomi va asosiy savollari	Soat	Metod	TN
M20	Dasturiy ta'minot ishlab chiqish texnologiyalari: Git va GitHub, testlash, kod sifati. Sun'iy intellekt yordamchilaridan dasturlashda foydalanish 1. Versiyalarni boshqarish 2. Git asosiy buyruqlari 3. GitHub va hamkorlik 4. Testlash va debugging 5. Kod sifati va uslub 6. SI yordamchilari va akademik halollik	2	Keys-stadi, debat	TN6

3. Amaliy mashg'ulotlarning mavzuli rejasi

№	Amaliy mashg'ulot mavzusi va natijasi	Soat	Shakli	TN
A1	Python va VS Code/Thonny muhitini o'rnatish, birinchi dasturni yozish. Natija: O'rnatilgan muhit va ishlaydigan birinchi dastur	2	Individual	TN1
A2	O'zgaruvchilar, ma'lumot turlari va arifmetik amallarga doir masalalar. Natija: Arifmetik masalalar yechimlari (8–10 ta dastur)	2	Individual	TN7
A3	Kiritish-chiqarish va turlarni o'zgartirishga doir masalalar. Natija: Kiritish-chiqarish dasturlari va tur xatolari tahlili	2	Individual	TN7
A4	Shartli operatorlarga doir masalalar. Natija: Shartli operatorli 8 ta dastur	2	Individual	TN7
A5	Murakkab shartlar va ichma-ich shartli operatorlar. Natija: Murakkab shartli dasturlar va test jadvali	2	Individual	TN7
A6	for sikli: yig'indi, ko'paytma va sanash masalalari. Natija: for siklli 8 ta dastur	2	Individual	TN7
A7	while sikli va iteratsion jarayonlar. Natija: while siklli dasturlar va kuzatuv jadvali	2	Individual	TN7
A8	Ichma-ich sikllar va naqshlar chizishga doir masalalar. Natija: Naqsh chizuvchi dasturlar to'plami	2	Individual	TN7
A9	Sonlar nazariyasiga doir masalalar: tub sonlar, EKUB, EKUK, raqamlar yig'indisi. Natija: Sonlar nazariyasi bo'yicha dasturlar	2	Juftlik	TN9
A10	Satrlar ustida amallar va satr metodlari. Natija: Satr metodlari bo'yicha dasturlar	2	Individual	TN7
A11	Satrlarni qayta ishlashga doir masalalar: palindrom, so'zlarni sanash. Natija: Satr masalalari yechimlari	2	Juftlik	TN7
A12	Ro'yxatlar bilan ishlash: yaratish, qo'shish, o'chirish, kesimlar. Natija: Ro'yxat amallari bo'yicha dasturlar	2	Individual	TN7

№	Amaliy mashg'ulot mavzusi va natijasi	Soat	Shakli	TN
A13	Ro'yxatlarda qidirish, eng katta va eng kichik elementni topish, saralash. Natija: Qidirish va saralash dasturlari	2	Individual	TN7
A14	List comprehension va ikki o'lchovli ro'yxatlar (matritsalar). Natija: Matritsa va comprehension dasturlari	2	Individual	TN7
A15	Kortejlar va to'plamlarga doir masalalar. Natija: Kortej va to'plam masalalari	2	Individual	TN7
A16	Lug'atlar bilan ishlash: chastota lug'ati, telefon kitobi. Natija: Lug'atli dasturlar	2	Juftlik	TN7
A17	Funksiyalar yaratish va chaqirish. Natija: Funksiyalar to'plami	2	Individual	TN7
A18	Rekursiv funksiyalar: faktorial, Fibonachchi sonlari, Xanoy minorasi. Natija: Rekursiv funksiyalar va ularning tahlili	2	Individual	TN7
A19	Standart kutubxona modullaridan foydalanish (math, random, datetime). Natija: Modullardan foydalanuvchi dasturlar	2	Individual	TN8
A20	Fayllarga yozish va fayllardan o'qish. Natija: Fayl bilan ishlovchi dasturlar	2	Individual	TN8
A21	CSV va JSON fayllari bilan ishlash. Natija: CSV va JSON bilan ishlovchi dasturlar	2	Individual	TN8
A22	Istisnolarni qayta ishlash. Kiritilgan ma'lumotlarni tekshirish. Natija: Xatolarga chidamli dasturlar	2	Individual	TN8
A23	Sinflar va ob'yektlar yaratish. Natija: Sinflar va ob'yektlar bilan dasturlar	2	Individual	TN8
A24	Vorislik va polimorfizmga doir masalalar. Natija: Vorislik ierarxiyasi dasturlari	2	Juftlik	TN8
A25	Pufakchali, tanlash va qo'shish usulida saralash algoritmlarini dasturlash. Natija: Uch saralash algoritmi dasturi	2	Individual	TN7
A26	Chiziqli va ikkilik qidiruv algoritmlari. Natija: Qidiruv algoritmlari dasturi	2	Individual	TN7
A27	Stek va navbatni dasturlash. Natija: Stek va navbat sinflari	2	Juftlik	TN7
A28	Graflarda kenglik va chuqurlik bo'yicha qidiruv (BFS, DFS) asoslari. Natija: BFS va DFS dasturlari	2	Juftlik	TN7
A29	C++ da birinchi dastur, kiritish-chiqarish va shartli operatorlar. Natija: C++ dasturlari to'plami	2	Individual	TN6
A30	C++ da sikllar va massivlar. Natija: C++ da sikl va massiv dasturlari	2	Individual	TN6

№	Amaliy mashg'ulot mavzusi va natijasi	Soat	Shakli	TN
A31	C++ da funksiyalar va STL (vector, sort). Natija: C++ da funksiya va STL dasturlari	2	Individual	TN6
A32	Turtle kutubxonasi yordamida grafik dasturlar yaratish. Natija: Turtle grafik dasturlari	2	Individual	TN8
A33	Tkinter yordamida oddiy grafik interfeysli dastur (kalkulyator) yaratish. Natija: Tkinter kalkulyatori	2	Individual	TN8
A34	pandas va matplotlib yordamida o'quvchilar baholari jadvalini tahlil qilish. Natija: Tahlil hisoboti va diagrammalar	2	Individual	TN8
A35	Git va GitHub bilan ishlash: repozitoriy yaratish, commit, push. Natija: GitHub dagi repozitoriy	2	Individual	TN10

4. Laboratoriya mashg'ulotlarining mavzuli rejasi

№	Laboratoriya ishi mavzusi va natijasi	Soat	Shakli	TN
L1	Chiziqli va tarmoqlanuvchi algoritmlarga doir dasturlar majmuasini yaratish. Natija: Chiziqli va tarmoqlanuvchi dasturlar majmuasi	2	Individual	TN8
L2	Takrorlanuvchi jarayonlarga doir dasturlar majmuasini yaratish. Natija: Siklli dasturlar majmuasi	2	Individual	TN8
L3	random moduli yordamida "Sonni top" o'yinini yaratish. Natija: Ishlaydigan o'yin dasturi	2	Individual	TN8
L4	Matn statistikasini hisoblovchi dastur yaratish. Natija: Matn statistikasi dasturi	2	Individual	TN8
L5	Matritsalar ustida amallar bajaruvchi dastur yaratish. Natija: Matritsa amallari dasturi	2	Individual	TN8
L6	Lug'atlar asosida "Lug'at-tarjimon" dasturini yaratish. Natija: Lug'at-tarjimon dasturi	2	Individual	TN8
L7	Funksiyalar kutubxonasini (modul) yaratish va undan foydalanish. Natija: O'z moduli va uni qo'llovchi dastur	2	Individual	TN8
L8	Rekursiv algoritmlar: kombinatorika masalalarini dasturlash. Natija: Kombinatorika dasturlari	2	Individual	TN8
L9	Fayl asosidagi "O'quvchilar ro'yxati" dasturini yaratish (qo'shish, qidirish, saralash). Natija: Fayl asosidagi ro'yxat dasturi	2	Individual	TN8
L10	JSON asosida test (viktorina) dasturini yaratish. Natija: JSON asosidagi viktorina	2	Individual	TN8
L11	OYD asosida "Kutubxona" tizimi modelini yaratish. Natija: Kutubxona tizimi modeli	2	Individual	TN8

№	Laboratoriya ishi mavzusi va natijasi	Soat	Shakli	TN
L12	Saralash algoritmlarining samaradorligini tajribada solishtirish. Natija: Vaqt o'lchash natijalari jadvali	2	Individual	TN8
L13	Ma'lumotlar tuzilmalaridan foydalanib masalalar yechish. Natija: Tuzilmalar asosidagi yechimlar	2	Individual	TN8
L14	Onlayn masalalar tizimlarida (Codeforces va boshqalar) olimpiada masalalarini yechish. Natija: Yechilgan masalalar hisoboti	2	Individual	TN9
L15	C++ tilida olimpiada tipidagi masalalarni yechish. Natija: C++ yechimlari to'plami	2	Individual	TN9
L16	Turtle yordamida fraktallar va geometrik naqshlar chizish. Natija: Fraktal dasturlar	2	Individual	TN8
L17	Tkinter da grafik interfeysli o'quv dasturini yaratish. Natija: O'quv dasturi (GUI)	2	Individual	TN8
L18	Ma'lumotlarni vizuallashtirish: diagrammalar qurish. Natija: Diagrammalar to'plami	2	Individual	TN8
L19	Kodni testlash (unittest) va xatolarni tuzatish (debugging). Natija: Testlar to'plami va tuzatilgan kod	2	Individual	TN10
L20	Yakuniy loyiha: maktab uchun foydali dastur (test tizimi, dars jadvali generatori va h.k.) yaratish, GitHub ga joylashtirish va himoya qilish. Natija: Maktab uchun dastur, GitHub repozitoriysi va himoya	2	Individual, himoya	TN10

5. Mustaqil ta'lim topshiriqlarining taqsimoti

№	Topshiriq	Bog'liq mavzu	Shakli	Topshirish muddati
1	Zamonaviy dasturlash tillarini qiyosiy tahlil qilish va taqdimot	M1, M16	Individual	A5 mashg'ulotigacha
2	O'quvchilar uchun darajalangan Python masalalari to'plami	M3–M9, A2–A9	Individual	A10 mashg'ulotigacha
3	Saralash algoritmlarini vizual tushuntiruvchi o'quv material	M14, A25, L12	Individual	L12 mashg'ulotigacha
4	Onlayn masalalar tizimida 20 ta masalani yechish va hisobot	M14, M17, L14, L15	Individual	L15 mashg'ulotigacha
5	OYD asosida kichik o'quv loyihasi	M12, M13, A23, L11	Kichik guruh (3–4 kishi)	L11 mashg'ulotida
6	Ma'lumotlar tahlili bo'yicha mini-loyiha va GitHub	M19, M20, A34, A35	Kichik guruh (3–4 kishi)	L20 mashg'ulotigacha

6. Nazorat turlari va baholash tartibi

Tinglovchilar bilimi joriy va yakuniy nazorat orqali 5 baholik tizimda baholanadi.

Joriy nazorat amaliy va laboratoriya mashg'ulotlarida bajarilgan dasturlarni himoya qilish, ma'ruzalardagi blits-so'rovlar hamda mustaqil ta'lim topshiriqlarini himoya qilish orqali amalga oshiriladi. Har bir amaliy va laboratoriya ishi 5 baholik shkalada baholanadi: dastur to'g'ri ishlashi, test ma'lumotlarida tekshirilganligi, kod tuzilmasi va o'qilishi, tushuntirish va nazorat savollariga javob inobatga olinadi. Joriy baho barcha amaliy, laboratoriya va mustaqil ish baholarining o'rtacha arifmetigi sifatida aniqlanadi.

Yakuniy nazorat yozma ish shaklida o'tkaziladi. Yozma ish varianti 3 ta topshiriqdan iborat: ikkita nazariy savol va bitta amaliy topshiriq (masalan, berilgan masalaning algoritmini tuzish va Python yoki C++ da dasturini yozish, kod fragmentidagi xatoni topish va tuzatish, sinf ierarxiyasini loyihalash).

Yakuniy nazoratga barcha amaliy va laboratoriya ishlarini topshirgan va joriy nazoratdan kamida "3" baho olgan tinglovchilar qo'yiladi. Fan bo'yicha umumiy baho joriy nazorat (40%) va yakuniy nazorat (60%) natijalarining o'lchangan o'rtachasi sifatida, butun songa yaxlitlab aniqlanadi.

Nazorat turi	Shakli	Ulushi
Joriy nazorat	Amaliy va laboratoriya ishlari himoyasi, blits-so'rov, mustaqil ta'lim	40%
Yakuniy nazorat	Yozma ish (3 topshiriq, 80 daqiqa)	60%

7. Fanni o'qitish uchun zarur moddiy-texnik va dasturiy ta'minot

- kompyuter sinfi (har bir tinglovchiga bitta kompyuter), proyektor yoki interaktiv doska;
- Internet tarmog'iga ulanish (paketlarni o'rnatish, GitHub va onlayn tizimlar uchun);
- Python 3 (Thonny, VS Code yoki PyCharm Community), Jupyter Notebook;
- pip orqali o'rnatiladigan kutubxonalar: NumPy, pandas, matplotlib;
- C++ kompilyatori (g++ yoki mos IDE) yoki onlayn kompilyator;
- Git va GitHub hisobi;
- olimpiada masalalarini onlayn tekshiruvchi tizim (Codeforces yoki shunga o'xshash platforma);
- algoritmlar vizuallashtirish vositasi (VisuAlgo).

III. FANNI O‘QITISHDA FOYDALANILADIGAN INTERFAOL TA’LIM METODLARI

Fanni o‘qitishda tinglovchilarning faolligini oshirish, nazariy bilimlarni amaliyot bilan bog‘lash va kasbiy-metodik kompetensiyalarni shakllantirish maqsadida quyidagi interfaol metodlardan foydalaniladi. Har bir metod ikki vazifani bajaradi: birinchidan, ma’ruza va amaliy mashg‘ulotda tinglovchining o‘zi shu metod bilan ishlab ko‘radi; ikkinchidan, uni maktab darsiga ko‘chirishning aniq namunasini oladi. Shu sababli har bir metod uchun “Fanga tatbiq etish namunasi” va “Maktab darsiga ko‘chirish” qismlari alohida berilgan.

1. “Blits-so‘rov” metodi

Maqsadi: qisqa vaqt ichida o‘tilgan mavzu bo‘yicha tayanch tushunchalarni takrorlash va tinglovchilarning tayyorgarlik darajasini aniqlash.

O‘tkazish tartibi:

- o‘qituvchi 8–10 ta qisqa savol tayyorlaydi;
- savollar tez sur‘atda navbat bilan beriladi, javob uchun 10–15 soniya ajratiladi;
- noto‘g‘ri javoblar darhol boshqa tinglovchilar tomonidan to‘g‘rilanadi;
- yakunda o‘qituvchi umumiy xulosa qiladi va keyingi mavzu bilan bog‘laydi.

Fanga tatbiq etish namunasi (M5 boshida): “if operatori qanday yoziladi?”, “Mantiqiy ifoda qanday qiymat qaytaradi?”, “range(5) qaysi sonlarni beradi?”, “Indent nima uchun kerak?”, “Cheksiz sikl nima?” Javoblarga qarab o‘qituvchi ma’ruzaning qaysi qismiga ko‘proq vaqt ajratishni belgilaydi.

Maktab darsiga ko‘chirish: 8-sinfda “Sikllar” mavzusidan oldin: “O‘zgaruvchi nima?”, “Shart qanday tekshiriladi?”, “Dastur satrlari qanday tartibda bajariladi?”

2. “Klaster” metodi

Maqsadi: mavzuga oid tushunchalar o‘rtasidagi bog‘lanishlarni grafik shaklda aks ettirish orqali tizimli fikrlashni rivojlantirish.

O‘tkazish tartibi: doska yoki qog‘oz markaziga asosiy tushuncha yoziladi (masalan, “Funksiya”). Atrofiga u bilan bog‘liq tushunchalar yoziladi va chiziqlar bilan bog‘lanadi. Keyin har bir tarmoq yangi tushunchalar bilan to‘ldiriladi. Yakunda guruhlar klasterlarini taqqoslaydi.

Qo‘llanilishi: M1, M7, M12 va M15 mavzularida. Elektron variantda klasterlarni Miro yoki Canva kabi bepul onlayn doskalarda yaratish mumkin.

Maktab darsiga ko‘chirish: 8-sinfda “Ro‘yxat” tushunchasi atrofida klaster: indeks, kesim, metodlar, sikl bilan o‘tish, ro‘yxat ichida ro‘yxat.

3. “Insert” metodi

Maqsadi: matn bilan ishlashda faol o‘qish va tanqidiy fikrlashni shakllantirish.

Tinglovchilar ma’ruza matnining bir qismini o‘qiydi va quyidagi belgilar bilan belgilaydi:

Belgi	Ma’nosi
V	Bu menga ma’lum (oldin bilgan edim)
+	Bu men uchun yangi ma’lumot
–	Bu mening tasavvurimga zid (qarama-qarshi)
?	Bu menga tushunarsiz yoki qo‘shimcha ma’lumot kerak

Natijalar jadvalga yoziladi va “?” belgisi qo‘yilgan joylar birgalikda muhokama qilinadi. M1 mavzusining “Tillarning qiyosiy tahlili” qismi va M20 mavzusining “Git va GitHub” qismi shu metod bilan o‘rganiladi.

4. “Venn diagrammasi” metodi

Maqsadi: ikki yoki uch tushunchani taqqoslash, ularning umumiy va farqli jihatlarini aniqlash.

Fanga tatbiq etish namunalari: “Kompilyator va interpretator” (M1), “Ro‘yxat va kortej” (M7), “Ro‘yxat va lug‘at” (M8), “Python va C++” (M16), “Vorislilik va kompozitsiya” (M13).

Maktab darsiga ko‘chirish: “for va while sikli”, “Funksiya va metod”, “Stek va navbat”, “Ro‘yxat va massiv”.

5. “Aqliy hujum” (Brainstorming) metodi

Maqsadi: muammo bo‘yicha qisqa vaqt ichida imkon qadar ko‘p g‘oya to‘plash.

Qoidalar: g‘oyalar tanqid qilinmaydi; miqdor sifatdan muhim; boshqalarning g‘oyasini rivojlantirish rag‘batlantiriladi; eng g‘ayrioddiy g‘oyalar ham yoziladi. Keyingi bosqichda g‘oyalar guruhlanadi va eng amaliylari tanlanadi.

Namunalar: “Dasturdagi xatoni tez topishning qanday usullari bor?” (M20), “Maktab uchun qanday foydali dasturlar yaratish mumkin?” (A35, L20).

Maktab darsiga ko‘chirish: “Dasturlashni o‘rganishda nima qiyin?”, “Python bilan qanday o‘yinlar yaratish mumkin?”

6. “Keys-stadi” metodi

Maqsadi: real yoki real holatga yaqin pedagogik vaziyatlarni tahlil qilish orqali muammoni aniqlash va yechim ishlab chiqish ko‘nikmasini shakllantirish.

Bosqichlari:

- keys matni bilan individual tanishish (5–7 daqiqa);
- muammoni aniqlash va asosiy savollarni shakllantirish;
- kichik guruhlarda yechim variantlarini ishlab chiqish (10–15 daqiqa);
- yechimlarni taqdim etish va himoya qilish;
- o‘qituvchi tomonidan umumlashtirish va baholash.

Fan bo‘yicha keyslar banki majmuaning IX bo‘limida berilgan.

7. “Zigzag” (Jigsaw) metodi

Maqsadi: katta hajmdagi materialni qisqa vaqtda o‘zlashtirish, har bir tinglovchining mas’uliyatini oshirish.

O‘tkazish tartibi:

- guruh 4 kishilik “uy guruhlari”ga bo‘linadi; har bir a‘zo materialning bitta qismini oladi (masalan, M14 da: 1) chiziqli va ikkilik qidiruv, 2) pufakchali saralash, 3) tanlash usulida saralash, 4) qo‘shish usulida saralash);
- bir xil qismni olganlar “ekspert guruhi”da yig‘ilib, 10–12 daqiqa davomida materialni o‘rganadi va qanday tushuntirishni rejalashtiradi;
- har kim uy guruhiga qaytib, o‘z qismini sheriklariga tushuntiradi;
- yakunda qisqa test yoki savol-javob o‘tkaziladi.

Maktab darsiga ko‘chirish: “Takrorlash operatorlari” mavzusida har bir ekspert guruhi bitta mavzu (for, while, break/continue, ichma-ich sikllar) bo‘yicha mutaxassis bo‘ladi va uy guruhiga o‘rgatadi.

8. “Baliq skeleti” (Fishbone) metodi

Maqsadi: muammoning sabablarini tizimli aniqlash va tasniflash.

O‘tkazish tartibi: baliq boshiga muammo yoziladi (masalan, “O‘quvchilar dasturda shart operatorini noto‘g‘ri yozadi”). Yuqori suyaklarga sabablar, pastki suyaklarga ularni tasdiqlovchi dalillar yoziladi. Dumiga yechim yoki xulosa yoziladi.

Qo‘llanilishi: M5, M9 mavzularida va A18 hamda 3-mustaqil topshiriq muhokamasida.

9. “Fe’llar narvoni” metodi

Maqsadi: Blum taksonomiyasi darajalarini va ularga mos harakat fe’llarini amaliy o‘zlashtirish, o‘lchanadigan ta’lim natijalarini va topshiriqlarni yozish.

O‘tkazish tartibi: doskaga olti pog‘onali narvon chiziladi (eslash, tushunish, qo‘llash, tahlil, baholash, yaratish). Guruhlar har bir pog‘onaga mos fe’l va “Sikllar” mavzusidan bitta topshiriq yozadi.

Qo'llanilishi: M5 va A6, A7 mashg'ulotlarida.

10. Juftlikda dasturlash (Pair Programming)

Maqsadi: hamkorlikda masala yechish, kodni og'zaki izohlash va o'zaro o'rganishni rivojlantirish.

Rollar: "Haydovchi" (driver) klaviatura va sichqonchani boshqaradi, kodni yozadi; "Yo'l ko'rsatuvchi" (navigator) kodni kuzatadi, xatolarni topadi va keyingi qadamni aytadi. Har 5–7 daqiqada rollar almashadi.

Qo'llanilishi: A8, A11, A16 va L5 mashg'ulotlarida; maktabda Python darslarida.

11. "Kodni o'qi va bashorat qil" (PRIMM) metodi

Maqsadi: dasturni yozishdan oldin uni o'qish va tushunish ko'nikmasini shakllantirish.

PRIMM besh bosqichdan iborat: Predict (bashorat qil) – o'quvchi dastur natijasini taxmin qiladi; Run (ishga tushir) – dastur bajariladi va bashorat tekshiriladi; Investigate (tahlil qil) – kod qatorma-qator tushuntiriladi; Modify (o'zgartir) – kod ozgina o'zgartiriladi; Make (yarat) – shunga o'xshash yangi dastur yoziladi.

Qo'llanilishi: M3–M9 mavzularida; A2–A8 mashg'ulotlarida Python kodi misolida.

12. "Sветофор" metodi

Maqsadi: dars davomida o'quvchilarning tushunish darajasi haqida tezkor teskari aloqa olish (formativ baholash).

O'tkazish tartibi: har bir tinglovchida qizil, sariq va yashil kartochka bo'ladi. O'qituvchi savol beradi yoki mavzuni tushuntirib bo'lgach, tinglovchilar kartochka ko'taradi: yashil – tushundim, sariq – qisman tushundim, qizil – tushunmadim. Natijaga qarab o'qituvchi tushuntirishni takrorlaydi yoki davom etadi.

Qo'llanilishi: M4, M9, A4–A7 mashg'ulotlarida; maktabda har qanday darsning mustahkamlash bosqichida.

13. SWOT-tahlil

Maqsadi: hodisa, texnologiya yoki shaxsiy faoliyatni kuchli (Strengths), zaif (Weaknesses) tomonlari, imkoniyatlari (Opportunities) va xavflari (Threats) bo'yicha tahlil qilish.

Qo'llanilishi: M1 da Python va C++ tillarining SWOT-tahlili; M20 da SI yordamchilaridan dasturlashda foydalanish bo'yicha guruhli tahlil.

14. Munozara (debat)

Maqsadi: dalillarga asoslangan nuqtayi nazarni shakllantirish va himoya qilish.

O‘tkazish tartibi: tezis e’lon qilinadi (masalan, “Informatika darsida o‘quvchilarga dasturlash topshiriqlarini bajarishda SI chat-botlaridan foydalanishga ruxsat berish kerak”). Guruh ikkiga bo‘linadi: “yoqlovchilar” va “inkor etuvchilar”. Har bir tomon argumentlarini taqdim etadi, savol-javob o‘tkaziladi, yakunda hakamlar xulosa qiladi.

Qo‘llanilishi: M20 mavzusida; shuningdek M1 da “Maktabda birinchi dasturlash tili qaysi bo‘lishi kerak?” mavzusida.

15. Mikrodars va uning tahlili

Maqsadi: dasturlash mavzusidagi dars fragmentini real sharoitga yaqin holatda sinab ko‘rish va kasbiy refleksiyaning rivojlantirish.

O‘tkazish tartibi: tinglovchi 10–15 daqiqalik dars fragmentini o‘tkazadi, guruhning qolgan a’zolari “o‘quvchi” rolini bajaradi. So‘ng kuzatuv varaqasi asosida tahlil qilinadi: nima yaxshi chiqdi, nima o‘zgartirilishi kerak, qanday dalillar bor. Dars o‘tkazgan tinglovchi o‘z-o‘zini tahlil qiladi.

Qo‘llanilishi: M18 (Turtle bilan o‘qitish) va L20 himoyasida.

16. Loyiha metodi

Maqsadi: egallangan bilim va ko‘nikmalarni yaxlit mahsulot – maktab uchun foydali dasturiy loyiha shaklida namoyon etish.

Bosqichlari: muammo va g‘oyani tanlash, maqsad va kutilayotgan natijani aniqlash, rejalashtirish va rollarni taqsimlash, bajarish, mahsulotni tayyorlash, taqdim etish va himoya qilish, refleksiya.

Qo‘llanilishi: 5- va 6-mustaqil topshiriq, L20 yakuniy loyiha.

IV. NAZARIY MATERIALLAR (MA'RUZA MATNLARI)

1-MAVZU. DASTURLASH TILLARI TASNIFI VA RIVOJLANISH TARIXI. KOMPILYATSIYA VA INTERPRETATSIYA. ZAMONAVIY DASTURLASH TILLARI (PYTHON, C++, JAVA, JAVASCRIPT)NING QIYOSIY TAHLILI

Reja:

1. Dasturlash tili tushunchasi va tasnifi.
2. Dasturlash tillarining rivojlanish tarixi.
3. Kompilyatsiya, interpretatsiya va aralash modellar.
4. Zamonaviy tillarning qiyosiy tahlili: Python, C++, Java, JavaScript.
5. Maktabda birinchi dasturlash tilini tanlash mezonlari.

Tayanch iboralar:

dasturlash tili, mashina kodi, assembler, yuqori darajali til, kompilyator, interpretator, bayt-kod, virtual mashina, statik va dinamik tiplash, paradigma, protsedurali dasturlash, ob'yektga yo'naltirilgan dasturlash, funksional dasturlash, sintaksis, semantika.

1. Dasturlash tili tushunchasi va tasnifi

Dasturlash tili – kompyuterga bajarilishi kerak bo'lgan amallarni aniq tavsiflash uchun mo'ljallangan rasmiy til. Tabiiy tildan farqli ravishda uning sintaksisi (yozuv qoidalari) va semantikasi (ma'nosi) qat'iy belgilangan: har bir to'g'ri yozilgan konstruktsiya bir ma'noga ega.

Dasturlash tillari bir necha belgi bo'yicha tasniflanadi:

Tasnif belgisi	Guruhlar	Misollar
Abstraksiya darajasi	Mashina kodi, assembler, yuqori darajali tillar	Mashina kodi, x86 assembler, Python, C++
Bajarilish usuli	Kompilyatsiya, interpretatsiya, aralash	C++, Python, Java
Tiplash	Statik va dinamik; kuchli va kuchsiz	C++ (statik), Python (dinamik, kuchli)
Paradigma	Protsedurali, ob'yektga yo'naltirilgan, funksional, deklarativ	C, Java, Haskell, SQL
Qo'llanish sohasi	Umumiy maqsadli va maxsus (domenga xos) tillar	Python, C++ / SQL, HTML

Paradigma – dastur yozish uslubi va fikrlash modeli. Zamonaviy tillarning ko'pchiligi ko'p paradigmali: Python da protsedurali, ob'yektga yo'naltirilgan va funksional uslubda yozish mumkin.

2. Dasturlash tillarining rivojlanish tarixi

Dasturlash tillari bir necha bosqichdan o'tgan:

- **Mashina kodi (1940-yillar):** dastur nollar va birlar ketma-ketligi; har bir protsessor uchun alohida.
- **Assembler (1950-yillar):** buyruqlar qisqa nomlar bilan yoziladi; hamon protsessorga bog'liq.
- **Birinchi yuqori darajali tillar (1950–60-yillar):** Fortran (ilmiy hisoblar), COBOL (biznes), Lisp, ALGOL; ular algoritm tilini inson tushunadigan shaklga yaqinlashtirdi.
- **Tuzilmali va tizimli dasturlash (1970-yillar):** Pascal, C; tuzilmali dasturlash – go to ni kamaytirib, tarmoqlanish va sikllardan foydalanish.
- **Ob'yektga yo'naltirilgan dasturlash (1980–90-yillar):** C++, Java; murakkab dasturlarni ob'yektlar bilan modellash.
- **Skript va dinamik tillar (1990-yillardan):** Python, JavaScript, Ruby; tez yozish va o'qilishi muhim.
- **Zamonaviy davr:** ko'p paradigmali va xavfsizlikka yo'naltirilgan tillar (Go, Rust, Kotlin, Swift), ma'lumotlar tahlili va SI uchun Python ning keng tarqalishi.

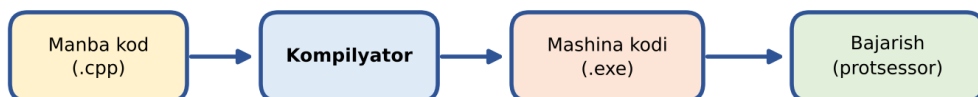
Tarixiy nuqtayi nazar o'quvchilarga muhim g'oyani ko'rsatadi: tillar insonning mashinaga “yaqinlashishi” emas, balki dasturchining murakkablikni boshqarish usullarining rivojlanishidir.

3. Kompilyatsiya, interpretatsiya va aralash modellar

Yuqori darajali tilda yozilgan dastur protsessor tushunadigan mashina kodiga aylantirilishi kerak. Bunga ikki asosiy yo'l bor (1-rasm).

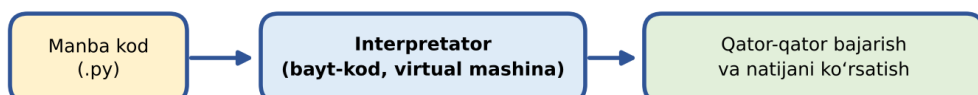
Dastur bajarilishining ikki yo'li

Kompilyatsiya (C++)



tarjima bir marta, butun dastur bo'yicha; keyin tez ishlaydi

Interpretatsiya (Python)



tarjima va bajarish ketma-ket; o'zgartirib, darhol sinash qulay

1-rasm. Dasturning bajarilish usullari: kompilyatsiya va interpretatsiya

Kompilyator butun dasturni bir marta mashina kodiga tarjima qiladi; natija (masalan, .exe fayl) mustaqil ishlaydi va odatda tez bajariladi. Xatolar tarjima bosqichida topiladi. **Interpretator** dasturni qator-qator o‘qib, darhol bajaradi; o‘zgartirib, darhol sinab ko‘rish qulay, lekin bajarilish sekinroq bo‘lishi mumkin.

Amalda ko‘pgina tillar **aralash** model qo‘llaydi: Python manba kodi avval bayt-kodga o‘giriladi, so‘ng virtual mashina uni bajaradi; Java ham bayt-kodga kompilyatsiya qilinib, JVM da ishlaydi.

Belgi	Kompilyatsiya	Interpretatsiya
Tarjima vaqti	Bajarishdan oldin, bir marta	Bajarish jarayonida
Tezlik	Odatda yuqori	Odatda past
Xatolarni topish	Tarjimada ko‘p xato oldindan topiladi	Xato dastur shu qatorga yetganda topiladi
Sinash qulayligi	Har o‘zgarishdan keyin qayta kompilyatsiya	Darhol bajarish, interaktiv rejim
O‘rganish	Qo‘shimcha bosqich	O‘quvchi uchun qulay

4. Zamonaviy tillarning qiyosiy tahlili

Belgi	Python	C++	Java	JavaScript
Tiplash	Dinamik	Statik	Statik	Dinamik
Bajarilish	Interpretator (bayt-kod)	Kompilyator	Bayt-kod, JVM	Brauzer/Node.js dvigateli
Sintaksis	Qisqa, o‘qilishi oson	Murakkab, boshqaruv ko‘p	Batafsil	O‘rtacha
Tezlik	Past-o‘rta	Yuqori	Yuqori	O‘rta
Asosiy sohalar	SI, tahlil, ta’lim, skriptlar	Tizim dasturlari, o‘yinlar, olimpiada	Korporativ tizimlar, Android	Veb-dasturlash

Bir xil masala turli tillarda qanchalik farq qilishini ko‘rish uchun “ikki sonning yig‘indisi” dasturini taqqoslash foydali:

```
# Python
a = int(input())
b = int(input())
print(a + b)

// C++
#include <iostream>
int main() {
    int a, b;
    std::cin >> a >> b;
    std::cout << a + b << std::endl;
    return 0;
}
```

Python da dastur uch-to'rt qator, C++ da esa sarlavha fayli, main funksiyasi va o'zgaruvchi turlari talab qilinadi. Bu farq o'quvchi uchun muhim: Python yangi boshlovchiga algoritimga e'tibor qaratish imkonini beradi, C++ esa kompyuter xotirasi va turlar haqida chuqurroq tushuncha beradi.

5. Maktabda birinchi dasturlash tilini tanlash mezonlari

Birinchi tilni tanlashda quyidagilar hisobga olinadi: sintaksisning soddaligi va o'qilishi, xatolarni tushunish qulayligi, bepul va oson o'rnatiladigan muhit, kutubxonalar va real loyihalar imkoniyati, o'qituvchi va o'quv resurslarining mavjudligi, keyingi bosqichlarga (olimpiada, universitet) o'tish imkoniyati. Shu mezonlar bo'yicha Python ko'pgina ta'lim dasturlarida (CS50, MIT 6.0001 va boshqalar) birinchi til sifatida qo'llanadi. C++ esa olimpiada va tizimli dasturlash uchun muhim ikkinchi til sifatida o'rganiladi (M16, M17). Maktab sharoitida ko'pincha vizual muhit (Scratch) – Python – C++ yo'nalishi tavsiya etiladi.

Metodik maslahat: “Eng yaxshi til” haqidagi bahs o'quvchilarni chalg'itadi. Maqsad tilni o'rganish emas, dasturlash tafakkurini shakllantirish ekanini ta'kidlang: o'zgaruvchi, shart, sikl, funksiya tushunchalari barcha tillarda o'xshash.

Mavzu bo'yicha xulosa

Dasturlash tillari abstraksiya darajasi, bajarilish usuli, tiplash va paradigma bo'yicha tasniflanadi. Tarixan mashina kodidan yuqori darajali ko'p paradigmali tillarga o'tish murakkablikni boshqarish vositalarining rivoji bilan bog'liq. Kompilyatsiya tezlik va oldindan tekshiruv, interpretatsiya esa qulaylik beradi; ko'p tillar aralash modelni qo'llaydi. Python, C++, Java va JavaScript turli sohalarga yo'naltirilgan; maktabda birinchi til sifatida Python, olimpiada uchun C++ keng qo'llanadi.

Nazorat savollari

1. Dasturlash tili nima va uning sintaksisi bilan semantikasi qanday farqlanadi?
2. Dasturlash tillari qanday belgilar bo'yicha tasniflanadi?
3. Programmalash paradigmasi nima? Misollar keltiring.
4. Dasturlash tillarining asosiy rivojlanish bosqichlarini ayting.
5. Kompilyator va interpretatorning farqini tushuntiring.
6. Python dasturi qanday bajariladi (bayt-kod, virtual mashina)?
7. Statik va dinamik tiplashni taqqoslang.
8. Python va C++ ning asosiy farqlari qaysilar?
9. Maktabda birinchi tilni tanlashda qanday mezonlar hisobga olinadi?
10. Nima uchun ko'plab ta'lim dasturlarida Python birinchi til sifatida tanlanadi?

Adabiyotlar: [1], [3]

2-MAVZU. PYTHON DASTURLASH MUHITI: O‘RNATISH, IDE (VS CODE, PYCHARM, THONNY), JUPYTER NOTEBOOK. BIRINCHI DASTUR

Reja:

1. Python tilining xususiyatlari va qo‘llanish sohalari.
2. Python ni o‘rnatish va versiyalar.
3. Dasturlash muhitlari: Thonny, VS Code, PyCharm.
4. Jupyter Notebook va interaktiv rejim.
5. Birinchi dastur va xato xabarlarini o‘qish.
6. Maktab uchun muhit tanlash.

Tayanch iboralar:

Python, interpretator, REPL, IDE, Thonny, VS Code, PyCharm, Jupyter Notebook, kod katagi, virtual muhit, skript, izoh, indentatsiya, xato xabari (traceback), debugger.

1. Python tilining xususiyatlari va qo‘llanish sohalari

Python – yuqori darajali, umumiy maqsadli, interpretatsiya qilinadigan til bo‘lib, o‘qilishi oson va ixcham sintaksisga ega. Uning xususiyatlari: bloklar bo‘shliq (indentatsiya) bilan ajratiladi; dinamik, lekin kuchli tiplash; boy standart kutubxona; ko‘p paradigmali; katta hamjamiyat va ko‘plab uchinchi tomon kutubxonalari.

Qo‘llanish sohalari: ta’lim, skriptlar va avtomatlashtirish, veb-dasturlash, ma’lumotlar tahlili va vizuallashtirish, sun’iy intellekt va mashinali o‘qitish, ilmiy hisoblar, o‘yinlar prototipi. Aynan shu ko‘p qirralilik Python ni ta’limda ham, amaliyotda ham mashhur qilgan.

2. Python ni o‘rnatish va versiyalar

Python rasmiy sayt (python.org) dan yuklab olinadi; Windows da o‘rnatuvchida “Add Python to PATH” belgisini qo‘yish muhim, aks holda buyruq satridan python buyrug‘i ishlamaydi. Linux va macOS da Python ko‘pincha oldindan o‘rnatilgan bo‘ladi. Tilning asosiy qatori Python 3; Python 2 allaqachon qo‘llab-quvvatlanmaydi. Joriy versiyani buyruq satridan tekshirish mumkin:

```
python --version  
python -c "print('Salom, dunyo!')"
```

Ba’zi tizimlarda buyruq python3 ko‘rinishida bo‘ladi. Majmuadagi kod namunalari Python 3 ning zamonaviy versiyalarida ishlashga mo‘ljallangan; match operatori kabi yangi imkoniyatlar uchun 3.10 va undan yuqori versiya kerak.

3. Dasturlash muhitlari

Muhit	Xususiyati	Afzalligi	Kamchiligi
Thonny	Boshlovchilar uchun oddiy IDE, Python bilan birga keladi	Sodda interfeys, qadamma-qadam bajarish, o'zgaruvchilar oynasi	Katta loyihalar uchun imkoniyatlari cheklangan
VS Code	Yengil, kengaytmalar bilan boyitiladigan muharrir	Ko'p til, Git integratsiyasi, kengaytmalar	Sozlash kerak
PyCharm (Community)	Python uchun maxsus professional IDE	Kuchli tahlil, debugger, refaktoring	Resurslarni ko'p talab qiladi
IDLE	Python bilan keladigan oddiy muhit	Qo'shimcha o'rnatish talab etilmaydi	Imkoniyati kam

IDE (integrated development environment) tarkibida muharrir, ishga tushirish tugmasi, xatolarni qidirish vositasi (debugger) va fayl boshqaruvi bo'ladi. Maktab uchun Thonny juda qulay: o'zgaruvchilar qiymatini ko'rsatadi va dasturni qadam-baqadam bajarishga imkon beradi – bu o'quvchida dastur “ichida nima bo'layotganini” tushunishni shakllantiradi.

4. Jupyter Notebook va interaktiv rejim

Python interaktiv rejimi (REPL) dagi har bir buyruq darhol bajariladi: ifodani kiritib, natijani ko'rish mumkin. Jupyter Notebook esa kod, matn, formulalar va grafiklarni bir hujjatda birlashtiradi: kod “katak”larga bo'linadi, har bir katak alohida bajariladi. Ma'lumotlar tahlili, o'quv materiallari va tajribalar uchun qulay. Ishga tushirish: `pip install notebook`, so'ng jupyter notebook buyrug'i (yoki onlayn xizmatlar orqali).

5. Birinchi dastur va xato xabarlarini o'qish

Birinchi dastur ikki qatordan iborat bo'lishi mumkin:

```
ism = input("Ismingiz nima? ")
print("Salom,", ism)
```

Dasturni ishga tushirishda xato bo'lsa, Python xato xabarini (traceback) beradi. Uni o'qishni o'rganish dasturlashning eng muhim ko'nikmalaridan biri:

```
print("Salom"
      File "dastur.py", line 1
        print("Salom"
              ^
SyntaxError: '(' was never closed
```

Xabarda xato turi (SyntaxError), fayl, qator raqami va tushuntirish ko'rsatilgan. Tipik boshlang'ich xatolar: qavs yoki qo'shtirnoqni yopmaslik, katta-

kichik harfni adashtirish (Print o‘rniga print), ortiqcha bo‘shliq (indentatsiya) va noto‘g‘ri nom.

Metodik maslahat: o‘quvchilar xato xabarini ko‘pincha o‘qimaydi va darhol o‘qituvchini chaqiradi. “Avval xabarni o‘qi: qaysi qator? qanday xato?” qoidasini kiriting. Bu keyinchalik mustaqil tuzatish ko‘nikmasini beradi.

6. Maktab uchun muhit tanlash

Maktab sharoitida muhit tanlashda: kompyuter resurslari, Internet mavjudligi, o‘rnatish osonligi, o‘quvchining uyda ham foydalana olishi hisobga olinadi. Oflayn: Thonny (eng sodda), IDLE. Onlayn: brauzerdagi Python muharrirlari (o‘rnatish talab qilinmaydi, lekin Internet kerak). Ko‘p maktablarda Thonny asosiy muhit, ma’lumotlar tahlili uchun esa Jupyter qo‘shimcha qo‘llanadi. Tanlangan muhit butun o‘quv yili davomida o‘zgarmagani ma’qul, aks holda vaqt muhitga o‘rganishga ketadi.

Mavzu bo‘yicha xulosa

Python o‘qilishi oson, ko‘p sohali tildir. U Python.org dan o‘rnatiladi; maktab uchun Thonny, keyinchalik VS Code yoki PyCharm, tahlil uchun Jupyter Notebook mos. Birinchi dasturlar va xato xabarlarini o‘qish ko‘nikmasi asosiy dasturlash madaniyatini boshlaydi. Muhit tanlashda resurs va qulaylik hisobga olinadi.

Nazorat savollari

1. Python tilining asosiy xususiyatlarini sanang.
2. Python qaysi sohalarda qo‘llanadi?
3. Windows da Python o‘rnatishda nimalarga e’tibor berish kerak?
4. IDE nima va u tarkibida nimalar bo‘ladi?
5. Thonny, VS Code va PyCharm ni taqqoslang.
6. REPL va skript rejimlarining farqi nimada?
7. Jupyter Notebook nima va u qachon qulay?
8. Xato xabarini (traceback) qanday o‘qiladi?
9. Boshlovchilar uchun tipik xatolarni sanang.
10. Maktab uchun dasturlash muhitini tanlash mezonlari qaysilar?

Adabiyotlar: [1], [2], [9]

3-MAVZU. O‘ZGARUVCHILAR, MA’LUMOT TURLARI, OPERATORLAR VA IFODALAR. KIRITISH VA CHIQUV

Reja:

1. O‘zgaruvchi tushunchasi va nom berish qoidalari.
2. Asosiy ma’lumot turlari: int, float, str, bool.
3. Operatorlar va ifodalar, ustuvorlik.

4. Turlarni o'zgartirish.
5. Kiritish va chiqarish.
6. Tipik xatolar va o'qitish metodikasi.

Tayanch iboralar:

o'zgaruvchi, nom, qiymat berish, int, float, str, bool, type(), arifmetik operator, taqqoslash operatori, mantiqiy operator, ustuvorlik, butun bo'lish (/), qoldiq (%), turni o'zgartirish, input(), print(), f-string.

1. O'zgaruvchi tushunchasi va nom berish qoidalar

O'zgaruvchi – nom berilgan xotira joyi bo'lib, unda qiymat saqlanadi. Python da o'zgaruvchi tur e'lon qilmasdan, qiymat berish bilan yaratiladi:

```
yosh = 17
ism = "Ali"
narx = 12.5
tayyor = True
yosh = yosh + 1    # qiymat o'zgardi
```

Nom berish qoidalar: harf yoki pastki chiziq bilan boshlanadi, harf, raqam va pastki chiziqdan iborat bo'ladi, katta-kichik harf farqlanadi (Yosh va yosh – har xil), xizmat so'zlari (if, for, class va boshqalar) nom bo'la olmaydi. Mazmunli nomlar tavsiya etiladi: x1 o'rniga talaba_yoshi.

Metodik maslahat: o'zgaruvchini “qutichaga qo'yilgan qiymat” analogiyasi bilan tushuntirish qulay, lekin Python da aniqrog'i: nom – qiymatga “yorliq”. Boshlang'ich sinflarda qutichaga o'xshatish yetarli.

2. Asosiy ma'lumot turlari

Tur	Nomi	Misol	Izoh
int	Butun son	7, -15, 0	Cheklanmagan uzunlikdagi butun sonlar
float	Haqiqiy son	3.14, -0.5, 2e3	Taxminiy aniqlikda saqlanadi
str	Satr	“salom”, 'a'	Belgilar ketma-ketligi
bool	Mantiqiy	True, False	Shartlarda ishlatiladi

Qiymat turini type() funksiyasi aniqlaydi: type(5) natijasi int, type("5") esa str. Muhim nuqta: float lar kompyuterda taxminiy saqlanadi, shu sababli 0.1 + 0.2 ifodasi aniq 0.3 emas, taxminan 0.30000000000000004 beradi. Pul hisoblarida bu muhim; shuning uchun ayrim hollarda sonlar butun (tiyin) ko'rinishida saqlanadi yoki round() qo'llanadi.

3. Operatorlar va ifodalar

Arifmetik operatorlar: + (qo‘shish), - (ayirish), * (ko‘paytirish), / (bo‘lish, natija float), // (butun bo‘lish), % (qoldiq), ** (darajaga ko‘tarish).

```
print(17 / 5)      # 3.4
print(17 // 5)     # 3
print(17 % 5)      # 2
print(2 ** 10)     # 1024
```

Taqqoslash operatorlari (==, !=, <, >, <=, >=) bool qiymat qaytaradi; mantiqiy operatorlar: and, or, not. Ustuvorlik: avval qavslar, so‘ng **, keyin unar minus, *, /, //, %, so‘ng +, -, keyin taqqoslash, not, and, or. Shubha bo‘lsa, qavs ishlatiladi. Qiymat berish operatori (=) taqqoslash operatori (==) bilan aralashtirilmasligi kerak.

Qisqartirilgan qiymat berish: x += 3 ifodasi x = x + 3 ga teng. Shu kabi -=, *=, //=, %= ham bor.

4. Turlarni o‘zgartirish

Turni o‘zgartirish funksiyalari: int(), float(), str(), bool().

```
a = int("25")      # 25
b = float("3.5")   # 3.5
c = str(100)       # "100"
d = int(7.9)       # 7 (qirqiladi, yaxlitlanmaydi)
```

Noto‘g‘ri qiymatni o‘zgartirishga urinish xatoga olib keladi: int(“salom”) ValueError beradi (bu xatolarni qayta ishlash M11 da ko‘riladi).

5. Kiritish va chiqarish

input() funksiyasi foydalanuvchidan satr qabul qiladi. Son kerak bo‘lsa, o‘zgartirish shart:

```
a = int(input("Birinchi son: "))
b = int(input("Ikkinchi son: "))
print("Yig'indi:", a + b)
print(f"{a} + {b} = {a + b}")
```

print() bir necha qiymatni vergul bilan qabul qiladi; sep va end parametrlari ajratuvchi va satr oxirini boshqaradi: print(1, 2, 3, sep=" ") natijasi 1-2-3. f-string (f"...") qiymatlarni matn ichiga qulay joylashtiradi (M6 da batafsil).

Bir qatordan bir nechta son o‘qishning odatiy usuli olimpiada masalalarida uchraydi: a, b = map(int, input().split()).

6. Tipik xatolar va o‘qitish metodikasi

- input() natijasini songa o‘zgartirmaslik: “5” + “3” natijasi “53” (satr birlashishi), 8 emas;
- = va == ni aralashtirish;
- // va / farqini bilmaslik;
- o‘zgaruvchi nomida bo‘shliq yoki xizmat so‘zidan foydalanish;

- qiymat berilmagan o'zgaruvchidan foydalanish (NameError);
- float ning aniq emasligini hisobga olmaslik.

Dars metodikasi: avval “quti” modeli, so'ng kichik topshiriqlar (ism va yosh so'rab xabar chiqarish, to'rtburchak yuzasi, qoldikli bo'lish). PRIMM metodi (A2–A3) samarali: o'quvchi avval kod natijasini taxmin qiladi, so'ng tekshiradi.

Mavzu bo'yicha xulosa

O'zgaruvchi nom berilgan qiymat saqlovchi vositadir. Asosiy turlar – int, float, str, bool; ularning har biri o'z amallariga ega. Operatorlar ifodalar tuzadi, ustuvorlik va turlarni o'zgartirish natijani belgilaydi. input() satr qaytarishi va float ning taxminiyligi eng ko'p uchraydigan xato manbalaridir.

Nazorat savollari

1. O'zgaruvchi nima va unga qanday nom beriladi?
2. Python ning asosiy ma'lumot turlarini sanang.
3. / va // operatorlarining farqi nima?
4. 17 % 5 va 2 ** 5 ifodalari natijasi nima?
5. Operatorlar ustuvorligini tushuntiring.
6. int(7.9) natijasi nima va nima uchun?
7. input() qanday turdagi qiymat qaytaradi?
8. 0.1 + 0.2 ifodasi nima uchun aniq 0.3 bo'lmaydi?
9. = va == ning farqini tushuntiring.
10. Mavzuni o'qitishdagi tipik xatolar qaysilar?

Adabiyotlar: [1], [2]

4-MAVZU. SHARTLI OPERATORLAR: IF, ELIF, ELSE, MATCH. MANTIQUIY IFODALAR

Reja:

1. Mantiqiy ifodalar va taqqoslash.
2. if operatori va to'liq tarmoqlanish.
3. elif zanjiri va ichma-ich shartlar.
4. match operatori.
5. Chekka holatlarni tekshirish.
6. Tipik xatolar va o'qitish metodikasi.

Tayanch iboralar:

shart, mantiqiy ifoda, if, elif, else, indentatsiya, taqqoslash zanjiri, and, or, not, truthy/falsy, match, case, chekka holat, test ma'lumotlari.

1. Mantiqiy ifodalar va taqqoslash

Shartli operatorlar dastur oqimini shartga qarab tarmoqlaydi. Shart – natijasi bool (True yoki False) bo‘lgan ifoda. Python da taqqoslash zanjiri mumkin: $0 < x < 10$ ifodasi x ning 0 va 10 orasida ekanligini tekshiradi. Mantiqiy operatorlarning ishlash jadvali:

A	B	A and B	A or B	not A
True	True	True	True	False
True	False	False	True	False
False	True	False	True	True
False	False	False	False	True

Python da shartda har qanday qiymat ishlatilishi mumkin: 0, bo‘sh satr “”, bo‘sh ro‘yxat [] va None yolg‘on (falsy), qolganlari rost (truthy) deb hisoblanadi. Boshlovchilarga esa dastlab aniq bool ifodalar bilan ishlash tavsiya etiladi.

2. if operatori va to‘liq tarmoqlanish

```
yosh = int(input("Yoshingiz: "))
if yosh >= 18:
    print("Kattalar")
else:
    print("Voyaga yetmagan")
```

Shart oxirida ikki nuqta (:) qo‘yiladi, bajariladigan blok esa bir xil chekinish (odatda 4 ta bo‘shliq) bilan yoziladi. Chekinish Python da sintaksisning bir qismi: noto‘g‘ri chekinish IndentationError ga olib keladi. Bu o‘quvchilarni dastur tuzilmasini vizual ko‘rishga o‘rgatadi.

3. elif zanjiri va ichma-ich shartlar

Ikkidan ortiq variant bo‘lsa, elif ishlatiladi:

```
ball = int(input("Ball: "))
if ball >= 86:
    baho = 5
elif ball >= 71:
    baho = 4
elif ball >= 56:
    baho = 3
else:
    baho = 2
print("Baho:", baho)
```

Shartlar yuqoridan pastga tekshiriladi va birinchi rost shartning bloki bajarilib, qolganlari o‘tkazib yuboriladi. Shuning uchun shartlar tartibi muhim: agar yuqorida ball >= 56 yozilsa, 90 ball ham “3” olib qoladi. Ichma-ich shartlar murakkab mantiq uchun qo‘llanadi, lekin juda chuqur ichma-ichlik kodni o‘qish qiyinlashtiradi; ko‘pincha and bilan soddalashtirish mumkin:

```
if yosh >= 18 and bor_guvohnoma:
    print("Haydashga ruxsat")
```

4. match operatori

Python 3.10 dan boshlab match operatori qiymatni bir nechta namuna bilan solishtirish imkonini beradi:

```
kun = int(input("Hafta kuni (1-7): "))
match kun:
    case 1 | 2 | 3 | 4 | 5:
        print("Ish kuni")
    case 6 | 7:
        print("Dam olish kuni")
    case _:
        print("Noto'g'ri raqam")
```

case _ – “boshqa barcha holatlar”. match uzun elif zanjirlarini ixchamroq qiladi, lekin faqat Python 3.10 va undan yuqori versiyalarda ishlaydi; muhit versiyasini tekshirish kerak.

5. Chekka holatlarni tekshirish

Shartli dasturni sinashda chegaraviy qiymatlar muhim: yuqoridagi baholash dasturida 55, 56, 70, 71, 85, 86 ballar tekshiriladi. Chekka holatlarni o‘ylash dasturchining muhim odati. Qo‘lda test jadvali tuzish tavsiya etiladi:

Kirish	Kutilgan natija	Tekshiriladigan holat
55	2	Quyi chegaradan pastroq
56	3	Quyi chegara
85	4	Yuqori chegaradan oldin
86	5	Yuqori chegara

6. Tipik xatolar va o‘qitish metodikasi

- if x = 5: – taqqoslash o‘rniga qiymat berish (SyntaxError);
- ikki nuqta (:) ni unutish yoki chekinishni buzish;
- shartlar tartibini noto‘g‘ri qo‘yish (kengroq shart birinchi turib, qolganlarini yopib qo‘yadi);
- taqqoslashda tur xatosi: “5” > 3 (satr va son);
- x == 1 or 2 deb yozish (to‘g‘risi x == 1 or x == 2 yoki x in (1, 2)).

Metodika: “Svetofor” va “Aqliy hujum” metodlari bilan “Agar yomg‘ir yog‘sa, soyabon ol” kabi kundalik shartlardan boshlash; so‘ng blok-sxemadan kodga o‘tish. A4–A5 mashg‘ulotlarida darajalangan masalalar: bitta shart, if-else, elif zanjiri, murakkab shart.

Mavzu bo'yicha xulosa

Shartli operator if-elif-else dastur oqimini mantiqiy ifoda natijasiga qarab tarmoqlaydi. Chekinish blok chegarasini belgilaydi; shartlar tartibi natijaga ta'sir qiladi. match ko'p variantli tanlovni ixcham ifodalaydi. Dasturni chekka holatlar bilan sinash va tipik xatolarni oldindan bilish o'qitishning asosiy talabi.

Nazorat savollari

1. Mantiqiy ifoda nima? Misollar keltiring.
2. and, or, not operatorlarining ishlash jadvalini tuzing.
3. Python da qaysi qiymatlar "yolg'on" hisoblanadi?
4. if-else tuzilmasini misol bilan tushuntiring.
5. Python da chekinishning ahamiyati qanday?
6. elif zanjirida shartlar tartibi nima uchun muhim?
7. match operatori qachon qulay va qaysi versiyadan boshlab bor?
8. Chekka holat nima va nima uchun tekshiriladi?
9. $x == 1$ or 2 yozuvi nima uchun noto'g'ri?
10. Shartli operatorni o'qitishda qanday metodlar qo'llanadi?

Adabiyotlar: [2], [4]

5-MAVZU. TAKRORLASH OPERATORLARI: FOR, WHILE, BREAK, CONTINUE

Reja:

1. Takrorlash tuzilmasi va sikllar turlari.
2. for sikli va range().
3. while sikli.
4. break va continue operatorlari.
5. Ichma-ich sikllar.
6. Kuzatuv jadvali va tipik xatolar.

Tayanch iboralar:

sikl, takrorlash (iteratsiya), for, while, range, hisoblagich, akkumulyator, break, continue, ichma-ich sikl, cheksiz sikl, kuzatuv jadvali (trace), else (siklda).

1. Takrorlash tuzilmasi va sikllar turlari

Sikl – bir xil amallar ketma-ketligini bir necha marta bajarish tuzilmasi. Sikllar ikki asosiy turga bo'linadi: takrorlar soni oldindan ma'lum bo'lganda (for) va shartga bog'liq bo'lganda (while). Sikl tanasi – takrorlanuvchi buyruqlar bloki; iteratsiya – siklning bitta bajarilishi.

Maktab uchun sikl g'oyasi unplugged faoliyatdan boshlanadi ("10 marta qarsak chal"), keyin blok-sxema va kodga o'tiladi.

2. for sikli va range()

```
for i in range(5):
    print(i)          # 0 1 2 3 4

for i in range(1, 6):
    print(i)          # 1 2 3 4 5

for i in range(10, 0, -2):
    print(i)          # 10 8 6 4 2
```

range(n) 0 dan n - 1 gacha, range(a, b) a dan b - 1 gacha, range(a, b, qadam) esa berilgan qadam bilan sonlar ketma-ketligini beradi; **oxirgi qiymat kirmaydi** – eng ko'p uchraydigan chalkashlik. for sikli nafaqat sonlar, balki satr va ro'yxat elementlari ustida ham ishlaydi (M6, M7).

Akkumulyator namunasi – 1 dan n gacha sonlar yig'indisi:

```
n = int(input())
yigindi = 0
for i in range(1, n + 1):
    yigindi += i
print(yigindi)
```

Bu yerda yigindi o'zgaruvchisi boshlang'ich qiymat (0) bilan yaratiladi va har iteratsiyada o'zgaradi. Ko'paytma hisoblashda boshlang'ich qiymat 1 bo'ladi.

3. while sikli

while sikli shart rost bo'lguncha takrorlanadi:

```
n = int(input("Son: "))
soni = 0
while n > 0:
    n //= 10
    soni += 1
print("Raqamlar soni:", soni)
```

while da sikl o'zgaruvchisini dasturchining o'zi o'zgartirishi shart; aks holda **cheksiz sikl** paydo bo'ladi. Foydalanuvchi to'g'ri qiymat kiritguncha so'rash while ning tipik qo'llanishi:

```
while True:
    x = int(input("Musbat son kiriting: "))
    if x > 0:
        break
```

for va while tanlovi: takrorlar soni ma'lum yoki ketma-ketlik ustida yurish bo'lsa – for; shart bajarilguncha takrorlash kerak bo'lsa – while.

4. break va continue operatorlari

break siklni darhol tugatadi, continue esa joriy iteratsiyaning qolgan qismini o'tkazib, keyingisiga o'tadi:

```
for i in range(1, 11):
    if i % 3 == 0:
        continue      # 3, 6, 9 o'tkazib yuboriladi
    if i == 8:
        break          # 8 da to'xtaydi
    print(i)           # 1 2 4 5 7
```

Siklga else qismi ham qo'shilishi mumkin: u break ishlamasdan sikl tugagandagina bajariladi (masalan, tub sonni tekshirishda foydali). Boshlovchilar uchun break va continue ni me'yorida ishlatish tavsiya etiladi: ortiqcha qo'llash mantiqni murakkablashtiradi.

5. Ichma-ich sikllar

Ichma-ich siklda tashqi sikl har bir qadamda ichki siklni to'liq bajaradi. Klassik misol – naqsh va jadvallar:

```
for i in range(1, 5):
    for j in range(i):
        print("*", end=" ")
    print()
```

Natija “*” lar uchburchagi: 1, 2, 3, 4 ta yulduzcha. Ko'paytirish jadvali ham ikkita ichma-ich sikl bilan hosil qilinadi. Umumiy iteratsiyalar soni tashqi va ichki takrorlar ko'paytmasiga teng ekanini ta'kidlash M14 dagi murakkablik tushunchasiga zamin yaratadi.

6. Kuzatuv jadvali va tipik xatolar

Kuzatuv jadvali (trace) – dastur bajarilishi davomida o'zgaruvchilar qiymatlarini qadam-baqadam yozib borish usuli. Masalan, yigindi dasturi $n = 4$ uchun:

Iteratsiya	i	yigindi	Izoh
Boshida	—	0	Boshlang'ich qiymat
1	1	1	$0 + 1$
2	2	3	$1 + 2$
3	3	6	$3 + 3$
4	4	10	Natija: 10

Tipik xatolar:

- range(1, n) yozib, n ning o'zi kirmasligini unutish;
- akkumulyatorni sikl ichida nollash;
- while da sikl o'zgaruvchisini o'zgartirishni unutib, cheksiz sikl hosil qilish;

- ichma-ich siklda noto‘g‘ri chekinish;
- hisoblagichga boshlang‘ich qiymat bermaslik.

Metodika: kuzatuv jadvali va Thonny ning qadamma-qadam rejimi o‘quvchiga siklni “ko‘rish”ga imkon beradi. “Fe’llar narvoni” metodi bilan sikllar bo‘yicha topshiriqlar daraja bo‘yicha tuziladi (A6–A8).

Mavzu bo‘yicha xulosa

Sikllar takrorlanuvchi amallarni ixcham yozish vositasidir. for takrorlar soni ma’lum bo‘lganda, while shartga bog‘liq holda qo‘llanadi. break va continue siklning bajarilishini boshqaradi. Ichma-ich sikllar jadval va naqshlarni hosil qiladi. Kuzatuv jadvali sikl mantig‘ini tushunish va xatolarni topishning asosiy vositasi.

Nazorat savollari

1. Sikl nima va u nima uchun kerak?
2. for va while sikllarining farqi nimada va qaysi birini qachon tanlaysiz?
3. range(2, 11, 3) qanday ketma-ketlikni beradi?
4. Akkumulyator nima? 1 dan n gacha yig‘indi dasturini yozing.
5. Cheksiz sikl qachon paydo bo‘ladi?
6. break va continue ning farqini tushuntiring.
7. Ichma-ich sikl yordamida uchburchak naqshi chizing.
8. Kuzatuv jadvali nima beradi?
9. Sikllarni o‘rgatishda qanday tipik xatolar uchraydi?
10. Unplugged faoliyatdan koddagi siklga o‘tish metodikasini tushuntiring.

Adabiyotlar: [2], [4]

6-MAVZU. SATRLAR VA ULAR USTIDA AMALLAR. SATR METODLARI, FORMATLASH (F-STRING)

Reja:

1. Satr tushunchasi, indekslash va uzunlik.
2. Kesimlar (slice).
3. Satr metodlari.
4. f-string va formatlash.
5. Satrlarni qayta ishlash masalalari.
6. Unicode va kodlash haqida tushuncha.

Tayanch iboralar:

satr (str), indeks, manfiy indeks, kesim, o‘zgarmas (immutable), len(), upper(), lower(), strip(), split(), join(), replace(), find(), count(), f-string, Unicode, ord(), chr().

1. Satr tushunchasi, indekslash va uzunlik

Satr (str) – belgilar ketma-ketligi bo‘lib, bir yoki ikkita qo‘shirnoq ichida yoziladi. Satrning har bir belgisi indeksga ega; sanash **0 dan** boshlanadi. Manfiy indekslar oxiridan sanaydi: -1 oxirgi belgi.

```
s = "Python"
print(s[0])      # P
print(s[-1])     # n
print(len(s))    # 6
```

Satrlar **o‘zgarmas** (immutable): `s[0] = "J"` yozuvi xatoga olib keladi. Satrni “o‘zgartirish” aslida yangi satr yaratishdir: `s = "J" + s[1:]`.

2. Kesimlar (slice)

Kesim satrning bir qismini beradi: `s[boshi:oxiri:qadam]`. Boshi kiradi, oxiri kirmaydi.

```
s = "Informatika"
print(s[0:4])      # Info
print(s[:6])       # Inform
print(s[6:])       # atika
print(s[::-1])     # akitamrofniI (teskari)
print(s[::2])      # Ifraia
```

Kesimlar ro‘yxat va kortejlarda ham xuddi shunday ishlaydi (M7), shuning uchun bu mavzuni mustahkam o‘zlashtirish muhim.

3. Satr metodlari

Metod – ob‘yektga tegishli funksiya bo‘lib, nuqta orqali chaqiriladi. Satr metodlari yangi satr qaytaradi, asl satr o‘zgarmaydi.

Metod	Vazifasi	Misol
<code>upper()</code> , <code>lower()</code>	Harflarni katta/kichik qilish	<code>"Ali".upper()</code> natijasi <code>"ALI"</code>
<code>strip()</code>	Chetdagi bo‘shliqlarni olib tashlash	<code>" salom ".strip()</code> natijasi <code>"salom"</code>
<code>split()</code>	Satrni bo‘laklarga ajratish (ro‘yxat)	<code>"a b c".split()</code> natijasi <code>['a', 'b', 'c']</code>
<code>join()</code>	Ro‘yxatni satrga birlashtirish	<code>"-".join(["1", "2"])</code> natijasi <code>"1-2"</code>
<code>replace(a, b)</code>	Qismni almashtirish	<code>"bola".replace("o", "a")</code> natijasi <code>"bala"</code>
<code>find(x)</code>	Birinchi topilgan indeks (topilmasa -1)	<code>"salom".find("l")</code> natijasi <code>2</code>
<code>count(x)</code>	Necha marta uchrashini sanash	<code>"banan".count("a")</code> natijasi <code>2</code>
<code>startswith()</code> , <code>endswith()</code>	Boshlanishi/tugashini tekshirish	<code>"test.py".endswith(".py")</code> natijasi <code>True</code>
<code>isdigit()</code> , <code>isalpha()</code>	Belgilar turini tekshirish	<code>"123".isdigit()</code> natijasi <code>True</code>

Metodlarni zanjir qilib yozish mumkin: `input().strip().lower()`. Kiritilgan ma’lumotni tozalash uchun bu odatiy usul.

4. f-string va formatlash

f-string (Python 3.6 dan) qiymatlarni satr ichiga qulay joylashtiradi:

```
ism = "Ali"
ball = 87.456
print(f"{ism} {ball:.1f} ball oldi")      # Ali 87.5 ball oldi
print(f"{5 * 3 = }")                     # 5 * 3 = 15
print(f"{255:b} {255:x} {7:03d}")        # 11111111 ff 007
print(f"{'markaz':^12}|{'chap':<8}|{'o'ng':>8}")
```

Formatlash belgilari: .2f – verguldan keyin 2 xona, d – butun son, 03d – 3 xonali, nol bilan to‘ldirilgan, ^ < > – tekislash, b va x – ikkilik va o‘n oltilik tizim. Eski usullar (% va format()) ham uchraydi, lekin f-string o‘qilishi oson bo‘lgani uchun tavsiya etiladi.

5. Satrlarni qayta ishlash masalalari

Namunaviy masalalar: palindromni tekshirish, so‘zlar sonini sanash, unli harflarni sanash, sezar shifri.

```
def palindrom(s):
    s = s.lower().replace(" ", "")
    return s == s[::-1]

matn = input()
unlilar = sum(1 for h in matn.lower() if h in "aeiou")
print(len(matn.split()), unlilar, palindrom(matn))
```

Bu masalalar for sikli, shart va kesimlarni birlashtiradi. Matnni qayta ishlash o‘quvchilar uchun qiziqarli, chunki natija real hayotga yaqin (shifrlash, matn tahlili, so‘z o‘yinlari).

6. Unicode va kodlash haqida tushuncha

Kompyuterda har bir belgi raqam bilan kodlanadi. Python satrlari Unicode asosida ishlaydi, shuning uchun o‘zbek lotin va kirill harflari, shuningdek emoji ham muammosiz qo‘llanadi. ord(“A”) belgi kodini (65), chr(65) esa kod bo‘yicha belgini beradi. Fayl bilan ishlashda kodlashni aniq ko‘rsatish (encoding=”utf-8”) tavsiya etiladi (M11). Bu mavzu informatikaning “axborotni kodlash” bo‘limi bilan bog‘lanadi.

Metodik maslahat: o‘zbek tilidagi o‘‘ va g‘ harflari ba‘zan turli belgilar (apostrof turlari) bilan yoziladi; matn tahlili masalalarida bu muammoni namoyish qilish o‘quvchilarga real ma’lumotlarning “iflosligi” haqida tushuncha beradi.

Mavzu bo‘yicha xulosa

Satr – o‘zgarmas belgilar ketma-ketligi; indeks 0 dan boshlanadi, kesimda oxirgi chegara kirmaydi. Satr metodlari yangi satr qaytaradi. f-string formatlashning eng qulay usuli. Satrlar bilan ishlash sikl, shart va kesimlarni birlashtiradi hamda Unicode kodlash bilan bog‘liq.

Nazorat savollari

1. Satr nima? Uning o‘zgarmasligi nimani anglatadi?
2. “Informatika”[2:7] va [::-1] natijalarini toping.
3. Manfiy indekslar qanday ishlaydi?
4. split() va join() metodlarini misol bilan tushuntiring.
5. strip(), lower() va replace() qanday maqsadlarda ishlatiladi?
6. find() va count() ning farqi nimada?
7. f-string da ikki xona aniqlikda formatlash qanday yoziladi?
8. Palindromni tekshirish dasturini yozing.
9. Unicode nima va ord() hamda chr() nimani bajaradi?
10. Satrlar mavzusini o‘rgatishda qanday tipik xatolar uchraydi?

Adabiyotlar: [1], [2]

7-MAVZU. RO‘YXATLAR VA KORTEJLAR. RO‘YXAT METODLARI, KESIMLAR, LIST COMPREHENSION

Reja:

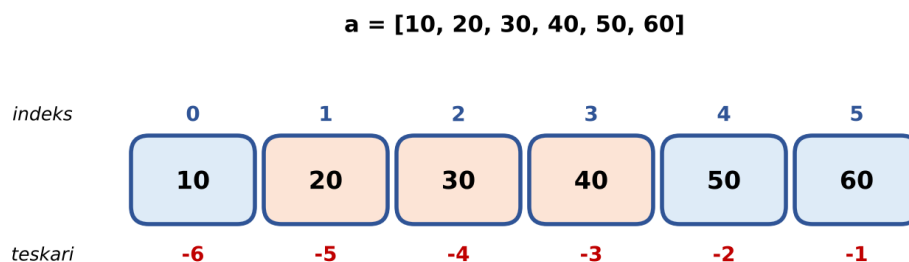
1. Ro‘yxat yaratish va indekslash.
2. Ro‘yxat metodlari.
3. Kesimlar va ro‘yxatni nusxalash.
4. List comprehension.
5. Kortejlar.
6. Ikki o‘lchovli ro‘yxatlar.

Tayanch iboralar:

ro‘yxat (list), element, indeks, append(), insert(), remove(), pop(), sort(), reverse(), kesim, nusxalash, list comprehension, kortej (tuple), o‘zgarmaslik, ikki o‘lchovli ro‘yxat (matritsa), unpacking.

1. Ro‘yxat yaratish va indekslash

Ro‘yxat (list) – tartiblangan, o‘zgaruvchan elementlar to‘plami; turli turdagi elementlarni saqlashi mumkin. Elementlarga indeks bo‘yicha murojaat qilinadi (2-rasm).



a[1:4] -> [20, 30, 40] (oxirgi indeks kirmaydi)

a[-1] = 60, a[::2] = [10, 30, 50], a[::-1] - teskari tartib

2-rasm. Ro'yxat indeklari va kesim: musbat va manfiy indekslar

```
sonlar = [5, 2, 9, 1]
bos = []
aralash = [1, "a", 3.5, True]
sonlar[1] = 20          # o'zgartirish mumkin
print(sonlar[-1])       # 1
print(len(sonlar))      # 4
for x in sonlar:
    print(x)
```

Python da ro'yxat boshqa tillardagi massivlarga o'xshash, lekin hajmi dinamik o'zgaradi. Ro'yxat bo'yicha for sikli har bir elementni ketma-ket beradi; indeks ham kerak bo'lsa, enumerate(sonlar) ishlatiladi.

2. Ro'yxat metodlari

Metod	Vazifasi	Misol
append(x)	Oxiriga element qo'shadi	a.append(5)
insert(i, x)	i-o'ringa qo'shadi	a.insert(0, 7)
remove(x)	Birinchi x ni o'chiradi	a.remove(5)
pop(i)	i-elementni olib tashlab qaytaradi (standart: oxirgisi)	a.pop()
sort()	Joyida tartiblaydi	a.sort(reverse=True)
reverse()	Teskari tartiblaydi	a.reverse()
index(x), count(x)	Joylashuv va uchrash soni	a.index(9)
extend(b)	Ikkinchi ro'yxatni qo'shadi	a.extend([1, 2])

Funksiyalar: len(), sum(), min(), max(), sorted() (yangi tartiblangan ro'yxat qaytaradi, asl o'zgarmaydi). Muhim farq: sort() ro'yxatni **joyida** o'zgartiradi va None qaytaradi; sorted() esa yangisini qaytaradi.

3. Kesimlar va ro'yxatni nusxalash

Kesimlar satrlardagi kabi ishlaydi: `a[1:3]`, `a[::-1]`. Ro'yxatlarda kesimga qiymat berish mumkin. Eng muhim nuqta: **`b = a`** ro'yxatni nusxalamaydi, ikkala nom bitta ob'yektga ishora qiladi:

```
a = [1, 2, 3]
b = a          # bitta ob'yekt!
b.append(4)
print(a)       # [1, 2, 3, 4]

c = a.copy()   # yoki a[:] yoki list(a)
c.append(5)
print(a)       # [1, 2, 3, 4] o'zgarmadi
```

Bu boshlovchilar uchun eng ko'p chalkashlik manbalaridan biri. Ichma-ich ro'yxatlarda `copy()` sayoz nusxa beradi; chuqur nusxa uchun `copy.deepcopy()` ishlatiladi.

4. List comprehension

List comprehension ro'yxatni bir qatorda yaratadi:

```
kvadratlar = [x * x for x in range(1, 6)]      # [1, 4, 9, 16, 25]
juftlar = [x for x in range(20) if x % 2 == 0]  # shart bilan
uzunlik = [len(s) for s in ["a", "bb", "ccc"]]  # [1, 2, 3]
```

Umumiy shakl: `[ifoda for element in ketma-ketlik if shart]`. Bu sikl va `append()` ning ixcham ekvivalenti. Ammo juda murakkab comprehension o'qishni qiyinlashtiradi – bunday hollarda oddiy sikl afzal. Boshlovchilarga avval oddiy sikl bilan yozib, keyin comprehension ga o'tish tavsiya etiladi.

5. Kortejlar

Kortej (tuple) – ro'yxatga o'xshash, lekin **o'zgarmas** ketma-ketlik; yumaloq qavs bilan yoziladi. Kortejlar o'zgarmas ma'lumotlar uchun (koordinata, sana) va funksiyadan bir nechta qiymat qaytarishda qo'llanadi:

```
nuqta = (3, 4)
x, y = nuqta          # ochish (unpacking)

def min_max(a):
    return min(a), max(a)

kichik, katta = min_max([4, 9, 1])
```

Kortej o'zgarmas bo'lgani uchun lug'at kaliti bo'la oladi (M8).

6. Ikki o'lchovli ro'yxatlar

Jadval va matritsalar ro'yxatlar ro'yxati sifatida ifodalanadi:

```

m = [[1, 2, 3],
      [4, 5, 6],
      [7, 8, 9]]
print(m[1][2])           # 6
for qator in m:
    print(*qator)
diagonal = sum(m[i][i] for i in range(3))
n = 3
togri = [[0] * n for _ in range(n)] # to'g'ri
noto = [[0] * n] * n               # XATO: qatorlar bitta ob'yekt

```

Eng tipik xato: `[[0] * n] * n` usuli bir xil ro'yxatni n marta takrorlaydi, shu sababli bitta elementni o'zgartirsak, butun ustun o'zgaradi. Muammo ro'yxat nusxalash mavzusi bilan bog'liq.

Metodik maslahat: nusxalash va bitta ob'yektga ikki nom muammosini Python Tutor (pythontutor.com) kabi vizualizatorlarda ko'rsatish samarali: o'quvchi xotira "xaritasini" ko'radi.

Mavzu bo'yicha xulosa

Ro'yxat – o'zgaruvchan tartiblangan to'plam; kesimlar, metodlar va list comprehension uni qulay qayta ishlaydi. `b = a` nusxa emas, bitta ob'yektga ikkinchi nom. Kortej o'zgarimas, ikki o'lchovli ro'yxatlar esa jadvallarni ifodalaydi; ularni yaratishda ro'yxatni takrorlash xatosiga e'tibor berish kerak.

Nazorat savollari

1. Ro'yxat nima va uning elementlariga qanday murojaat qilinadi?
2. `append()`, `insert()`, `remove()` va `pop()` metodlarining farqini tushuntiring.
3. `sort()` va `sorted()` ning farqi nimada?
4. `b = a` va `b = a.copy()` orasidagi farqni tushuntiring.
5. `a[1:4]` va `a[::-1]` kesimlarining natijasini aniqlang.
6. List comprehension nima? `[x * x for x in range(1, 6)]` natijasi qanday?
7. Kortej ro'yxatdan nimasi bilan farq qiladi va qachon ishlatiladi?
8. Unpacking nima? Misol keltiring.
9. Ikki o'lchovli ro'yxat qanday yaratiladi va `[[0] * n] * n` nima uchun xato?
10. Ro'yxatlar mavzusini o'qitishda qanday vizual vositalardan foydalaniladi?

Adabiyotlar: [1], [4]

8-MAVZU. LUG'ATLAR VA TO'PLAMLAR

Reja:

1. Lug'at tuzilishi va asosiy amallar.
2. Lug'at metodlari va yurish.
3. Chastota lug'atlari.

4. To‘plamlar va to‘plam amallari.
5. Tuzilma tanlash mezonlari.
6. Comprehension va tipik xatolar.

Tayanch iboralar:

lug‘at (dict), kalit, qiymat, get(), items(), keys(), values(), in operatori, chastota, to‘plam (set), birlashma, kesishma, ayirma, takrorlarni olib tashlash, hash, comprehension.

1. Lug‘at tuzilishi va asosiy amallar

Lug‘at (dict) – “kalit : qiymat” juftliklari to‘plami. Kalit bo‘yicha qiymatni tez topish imkonini beradi. Kalitlar noyob va o‘zgarmas turda (satr, son, kortej) bo‘ladi.

```
talaba = {"ism": "Ali", "yosh": 17, "baho": 5}
print(talaba["ism"])           # Ali
talaba["kurs"] = 1             # yangi kalit qo'shish
talaba["yosh"] = 18            # qiymatni o'zgartirish
del talaba["baho"]             # o'chirish
print("ism" in talaba)         # True (kalit bormi)
print(talaba.get("manzil", "yo'q")) # xatosiz, standart qiymat
```

Mavjud bo‘lmagan kalitga talaba[“manzil”] orqali murojaat KeyError beradi; get() esa bunday xatoni oldini oladi.

2. Lug‘at metodlari va yurish

Metod	Vazifasi	Izoh
keys()	Kalitlar ko‘rinishi	for k in d.keys()
values()	Qiymatlar ko‘rinishi	sum(d.values())
items()	(kalit, qiymat) juftliklari	for k, v in d.items()
get(k, std)	Xavfsiz olish	Kalit bo‘lmasa std qaytadi
pop(k)	O‘chirib, qiymatni qaytaradi	KeyError bo‘lishi mumkin
update(d2)	Boshqa lug‘atni qo‘shadi	Takrorlansa, yangilanadi
setdefault(k, std)	Kalit bo‘lmasa qo‘yadi	Guruhlashda qulay

```
narx = {"non": 3000, "sut": 12000, "tuxum": 15000}
for nom, summa in narx.items():
    print(f"{nom}: {summa} so'm")
```

Python 3.7 dan boshlab lug‘at kiritilish tartibini saqlaydi.

3. Chastota lug‘atlari

Lug‘atning eng klassik qo‘llanishi – elementlarni sanash:

```

matn = "bir ikki bir uch ikki bir"
chastota = {}
for soz in matn.split():
    chastota[soz] = chastota.get(soz, 0) + 1
print(chastota)          # {'bir': 3, 'ikki': 2, 'uch': 1}

eng_kop = max(chastota, key=chastota.get)
print(eng_kop)          # bir

```

Xuddi shu vazifani `collections.Counter` ham bajaradi (M10 da standart kutubxona ko‘riladi). Chastota lug‘ati matn tahlili, ovoz berish natijalarini sanash, harflar statistikasi kabi masalalarda qo‘llanadi.

4. To‘plamlar va to‘plam amallari

To‘plam (set) – takrorlanmaydigan, tartibsiz elementlar majmuasi. Matematikadagi to‘plamlar kabi birlashma, kesishma va ayirma amallari bor.

```

a = {1, 2, 3, 4}
b = {3, 4, 5}
print(a | b)      # birlashma {1, 2, 3, 4, 5}
print(a & b)      # kesishma {3, 4}
print(a - b)      # ayirma {1, 2}
print(a ^ b)      # simmetrik ayirma {1, 2, 5}

takrorsiz = set([1, 2, 2, 3, 3, 3])  # {1, 2, 3}
bos = set()                        # {} bo'sh lug'at bo'ladi!

```

To‘plamdan foydalanishning asosiy afzalligi – tegishlilikni tekshirish ($x \in s$) ro‘yxatdagiga qaraganda ancha tez ishlashi va takrorlarni olib tashlashning qulayligi. Ro‘yxatdan takrorlarni olib tashlash: `set(ro‘yxat)`, lekin tartib saqlanmaydi.

5. Tuzilma tanlash mezonlari

Tuzilma	Tartib	O‘zgaruvchan	Takror	Qachon ishlatiladi
Ro‘yxat (list)	Bor	Ha	Mumkin	Ketma-ketlik, tartiblash
Kortej (tuple)	Bor	Yo‘q	Mumkin	O‘zgarmas guruh, kalit
Lug‘at (dict)	Kiritilish tartibi	Ha	Kalit noyob	Kalit bo‘yicha qidirish
To‘plam (set)	Yo‘q	Ha	Yo‘q	Noyoblik, to‘plam amallari

“Qaysi tuzilma?” savoli algoritmik fikrlashning bir qismi: ma’lumot qanday ishlatilishiga qarab tanlanadi.

6. Comprehension va tipik xatolar

Lug‘at va to‘plam uchun ham comprehension mavjud:

```
kvadrat = {x: x * x for x in range(1, 6)}      # {1: 1, 2: 4, ...}
harflar = {h for h in "informatika"}          # noyob harflar
```

Tipik xatolar: mavjud bo'lmagan kalitga murojaat qilish (KeyError); o'zgaruvchan ob'yekt (ro'yxat) ni kalit qilishga urinish (TypeError); {} bilan bo'sh to'plam yaratish; lug'at bo'yicha yurganda uning o'zini o'zgartirish; to'plamda tartib kutish.

Metodik maslahat: lug'atni "ma'lumotnoma" (telefon daftari) bilan, to'plamni esa "noyob elementlar savati" bilan solishtirish o'quvchilarga tushunarli.

Mavzu bo'yicha xulosa

Lug'at kalit bo'yicha qiymatni tez topadi; chastota lug'ati sanash masalalarini hal qiladi. To'plam noyoblik va to'plam amallari uchun qulay. Ro'yxat, kortej, lug'at va to'plam o'rtasidan tanlash ma'lumot ishlatilishiga bog'liq.

Nazorat savollari

1. Lug'at nima va u ro'yxatdan nimasi bilan farq qiladi?
2. Lug'at kaliti qanday talablarga javob berishi kerak?
3. d[k] va d.get(k) o'rtasidagi farq nima?
4. items() metodidan foydalanib lug'atni qanday aylanib chiqamiz?
5. So'zlar chastotasini hisoblash dasturini yozing.
6. To'plam nima va qachon ishlatiladi?
7. Ikki to'plamning birlashmasi, kesishmasi va ayirmasini toping: {1, 2, 3} va {2, 3, 4}.
8. Nima uchun {} bilan bo'sh to'plam yaratib bo'lmaydi?
9. Qaysi tuzilmani qachon tanlash kerak? Misollar bilan tushuntiring.
10. Lug'at comprehension ga misol keltiring.

Adabiyotlar: [1], [4]

9-MAVZU. FUNKSIYALAR: PARAMETRLAR, QAYTARILADIGAN QIYMAT, KO'RINISH SOHASI, LAMBDA FUNKSIYALAR. REKURSIYA

Reja:

1. Funksiya yaratish va chaqirish.
2. Parametrlar va qaytariladigan qiymat.
3. Ko'rinish sohasi.
4. lambda funksiyalar.
5. Rekursiya g'oyasi.
6. Rekursiya va sikl solishtirish.

Tayanch iboralar:

funksiya, def, parametr, argument, return, standart qiymat, nomli argument, lokal va global o'zgaruvchi, ko'rinish sohasi, docstring, lambda, rekursiya, bazaviy holat, chaqiruvlar steki, RecursionError.

1. Funksiya yaratish va chaqirish

Funksiya – nom berilgan, qayta ishlatiladigan kod bloki. U dasturni bo'laklarga ajratadi, takrorlanishni kamaytiradi va o'qishni osonlashtiradi (dekompozitsiya).

```
def salom(ism):  
    """Foydalanuvchiga salom beradi."""  
    print("Salom, ", ism)  
  
salom("Ali")  
salom("Vali")
```

Funksiya def kalit so'zi, nomi, qavs ichida parametrlari va ikki nuqtadan keyin chekinish bilan yozilgan tanadan iborat. Tanadagi birinchi satr-qo'shtirnoq (docstring) funksiya vazifasini tushuntiradi. Funksiya faqat chaqirilganda bajariladi.

2. Parametrlar va qaytariladigan qiymat

```
def yuza(a, b=1):  
    return a * b  
  
print(yuza(3, 4))          # 12  
print(yuza(5))             # 5 (b standart qiymatda)  
print(yuza(b=2, a=6))      # 12 (nomli argumentlar)
```

Parametr – funksiya tavsifidagi nom, argument – chaqirishda berilgan qiymat. return qiymat qaytaradi va funktsiyani tugatadi; return bo'lmasa, funksiya None qaytaradi. Bir nechta qiymatni kortej sifatida qaytarish mumkin. Muhim farq: print() ekranga chiqaradi, return esa qiymatni dasturning boshqa qismiga uzatadi – boshlovchilar buni tez-tez aralashtiradi.

O'zgaruvchan argumentlar soni uchun *args ishlatiladi:

```
def yigindi(*sonlar):  
    return sum(sonlar)  
  
print(yigindi(1, 2, 3, 4)) # 10
```

Standart qiymat sifatida o'zgaruvchan ob'yekt (masalan, bo'sh ro'yxat) ishlatish xavfli: def f(a=[]) – ro'yxat barcha chaqiruvlarda umumiy bo'lib qoladi. To'g'risi: a=None, tanada esa if a is None: a = [].

3. Ko'rinish sohasi

Funksiya ichida yaratilgan o'zgaruvchi **lokal** bo'lib, faqat shu funksiya ichida ko'rinadi. Funksiyadan tashqarida yaratilgani **global**.

```
x = 10
def f():
    x = 5          # lokal x, globalga tegmaydi
    return x
print(f(), x)      # 5 10
```

global kalit soʻzi bilan globalni oʻzgartirish mumkin, lekin bu odatda yomon uslub hisoblanadi; maʼlumotlar parametrlar va qaytariladigan qiymat orqali uzatiladi. LEGB qoidasi: Python nomni lokal, oʻrab turuvchi, global, soʻngra oʻrnatilgan (built-in) sohalarda qidiradi.

4. lambda funksiyalar

lambda – nomsiz, bir ifodali qisqa funksiya: lambda x: x * x. Odatda tartiblash kaliti sifatida ishlatiladi:

```
kv = lambda x: x * x
print(kv(7))          # 49
talabalar = [("Ali", 85), ("Vali", 92), ("Gul", 78)]
talabalar.sort(key=lambda t: t[1], reverse=True)
print(talabalar)
```

Murakkab mantiq uchun odatiy def afzal. lambda map(), filter() va sorted() bilan qulay.

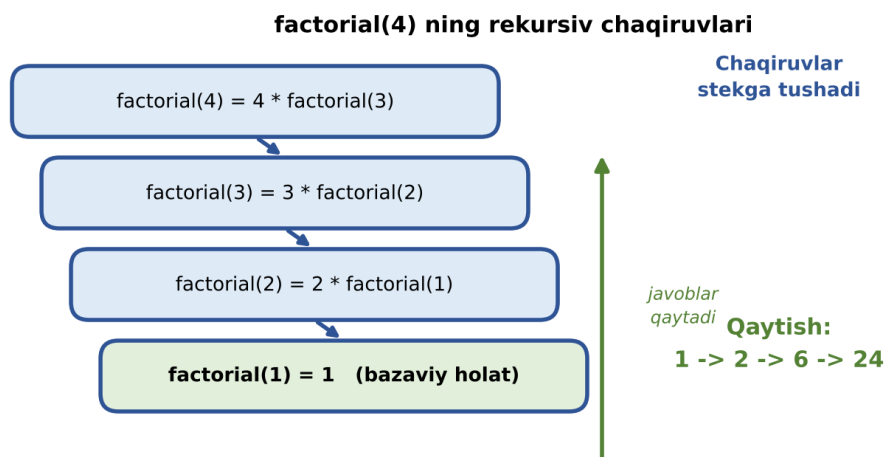
5. Rekursiya gʻoyasi

Rekursiya – funksiyaning oʻzini chaqirishi. Har bir rekursiv funksiya ikki qism boʻladi: **bazaviy holat** (toʻxtash sharti) va **rekursiv qadam** (masalani kichikroq qismiga keltirish).

```
def faktorial(n):
    if n == 0:          # bazaviy holat
        return 1
    return n * faktorial(n - 1)

print(faktorial(4))    # 24
```

Chaqiruvlar jarayonini stek bilan tushuntirish qulay (3-rasm; soddalik uchun rasmda bazaviy holat $n = 1$ da koʻrsatilgan, kodda esa $n == 0$): faktorial(4) -> faktorial(3) -> ... -> bazaviy holat; soʻng natijalar teskari tartibda qaytib koʻpaytiriladi.



3-rasm. *factorial(4)* chaqiruvlar steki

Bazaviy holat bo‘lmasa yoki unga yetib bo‘lmasa, `RecursionError` (chaqiruvlar chuqurligi chegarasidan oshish) yuz beradi.

6. Rekursiya va sikl solishtirish

Fibonachchi sonlari rekursiyaning klassik misoli:

```
def fib(n):
    if n < 2:
        return n
    return fib(n - 1) + fib(n - 2)

def fib_sikl(n):
    a, b = 0, 1
    for _ in range(n):
        a, b = b, a + b
    return a
```

`fib(n)` soddaligi bilan chiroyli, lekin bir xil qiymatlarni qayta-qayta hisoblagani uchun n ortishi bilan juda sekinlashadi (taxminan 2^n amal); `fib_sikl` esa chiziqli vaqtda ishlaydi. Bu M14 dagi murakkablik mavzusiga mos misol. Rekursiya daraxt va graf bilan ishlashda, “bo‘lib tashla” algoritmlarida (birlashtirib saralash, ikkilik qidiruv) tabiiy yechim beradi.

Metodik maslahat: rekursiyani tushuntirishda “matryoshka” va “ko‘zgudagi ko‘zgu” kabi analogiyalar, shuningdek kuzatuv jadvali va stek chizmasi foydali. Avval bazaviy holatni yozishni o‘rgating.

Mavzu bo‘yicha xulosa

Funksiya dasturni qismlarga ajratadi; parametrlar kirish, return esa natija uchun xizmat qiladi. Lokal va global sohalarni farqlash xatolarni oldini oladi. lambda qisqa nomsiz funksiya. Rekursiya bazaviy holat va rekursiv qadamdan iborat; chaqiruvlar stekda saqlanadi; sikl bilan solishtirish samaradorlik tushunchasiga olib boradi.

Nazorat savollari

1. Funksiya nima va uni nima uchun ishlatamiz?
2. Parametr va argumentning farqi nimada?
3. print va return o'rtasidagi farqni tushuntiring.
4. Standart qiymatli parametrlar qanday ishlaydi? Ro'yxat standart qiymati nima uchun xavfli?
5. Lokal va global o'zgaruvchilar nima?
6. *args nima uchun ishlatiladi?
7. lambda funksiya nima? sort(key=...) ga misol keltiring.
8. Rekursiya nima va bazaviy holat nima uchun kerak?
9. faktorial(4) chaqiruvlar jarayonini tushuntiring.
10. Fibonachchi uchun rekursiv va siklli yechimlarni taqqoslang.

Adabiyotlar: [2], [4], [15]

10-MAVZU. MODULLAR VA PAKETLAR. STANDART KUTUBXONA (MATH, RANDOM, DATETIME). PIP PAKET MENEJERI

Reja:

1. Modul va paket tushunchasi.
2. import usullari.
3. math, random, datetime modullari.
4. Qo'shimcha standart modullar.
5. pip va virtual muhit.
6. O'z modulini yaratish.

Tayanch iboralar:

modul, paket, import, from ... import, as, standart kutubxona, math, random, datetime, collections, pip, PyPI, virtual muhit (venv), requirements.txt, __name__.

1. Modul va paket tushunchasi

Modul – Python kodi yozilgan .py fayl; undagi funksiya va o'zgaruvchilarni boshqa dasturda qayta ishlatish mumkin. Paket – bir nechta modul joylashgan papka. Python “batareyalar bilan birga” falsafasiga amal qiladi: boy **standart kutubxona** o'rnatish bilan birga keladi. Uni docs.python.org saytidagi hujjatlardan o'rganish mumkin.

2. import usullari

```
import math
print(math.sqrt(16))

from math import pi, sqrt
print(pi)

import random as rnd
print(rnd.randint(1, 6))
```

Usullar: import modul (nom modul.nom shaklida), from modul import nom (to‘g‘ridan-to‘g‘ri), import modul as qisqa_nom. from modul import * usuli nomlarni to‘qnashtirishi mumkin, shuning uchun tavsiya etilmaydi.

3. math, random, datetime modullari

Modul	Imkoniyatlari	Misol
math	Matematik funksiyalar va konstantalar	math.sqrt(2), math.floor(3.7), math.pi, math.gcd(12, 18)
random	Tasodifiy sonlar va tanlash	random.randint(1, 10), random.choice(ro‘yxat), random.shuffle(a)
datetime	Sana va vaqt bilan ishlash	datetime.date.today(), datetime.timedelta(days=7)

```
import random, datetime
son = random.randint(1, 100) # topishmoq o'yini uchun
tanlov = random.choice(["tosh", "qaychi", "qog'oz"])
bugun = datetime.date.today()
print(bugun.year, bugun.strftime("%d.%m.%Y"))
```

random yordamida “Sonni top” va “Tosh-qaychi-qog‘oz” kabi o‘yinlar yaratish o‘quvchilarni juda jalb etadi. random.seed(n) natijalarni takrorlanuvchan qiladi – testlashda foydali.

4. Qo‘shimcha standart modullar

Boshqa foydali standart modullar: collections (Counter, deque, defaultdict), statistics (o‘rta, mediana), os va pathlib (fayl yo‘llari), time (vaqt o‘lchash), json va csv (M11), itertools (kombinatorika). Masalan, so‘zlar chastotasi:

```
from collections import Counter
print(Counter("informatika").most_common(3))
```

5. pip va virtual muhit

Standart kutubxonada bo‘lmagan paketlar PyPI (Python Package Index) omboridan pip yordamida o‘rnatiladi:

```
pip install requests
pip list
pip freeze > requirements.txt
pip install -r requirements.txt
```

Har bir loyiha uchun alohida **virtual muhit** (venv) yaratish tavsiya etiladi – shunda loyihalar paketlari bir-biriga xalaqit bermaydi:

```
python -m venv .venv
.venv\Scripts\activate      # Windows
source .venv/bin/activate   # Linux va macOS
```

requirements.txt fayli loyiha bog‘liqliklarini saqlaydi. Paket nomi va versiyalari o‘zgarishi mumkin, shuning uchun o‘rnatishdan oldin rasmiy hujjatni tekshirish lozim. Begona paketlarni o‘rnatishda ishonchli manbadan foydalanish va nomini aniq yozish xavfsizlik uchun muhim.

6. O‘z modulini yaratish

Har qanday .py fayl modul bo‘la oladi. Masalan, geometriya.py faylida:

```
# geometriya.py
import math

def doira_yuzi(r):
    return math.pi * r ** 2

if __name__ == "__main__":
    print(doira_yuzi(2))      # faqat to'g'ridan-to'g'ri ishga tushirilganda
```

Boshqa faylda: import geometriya; geometriya.doira_yuzi(3). if __name__ == “__main__”: sharti modul import qilinganda test kodining ishga tushishini oldini oladi. Fayl nomi standart modullar nomi (masalan, random.py) bilan bir xil bo‘lmasligi kerak – aks holda import xatosi yoki chalkashlik yuzaga keladi.

Metodik maslahat: o‘quvchilarga “o‘z kutubxonangni yarat” topshirig‘i dasturni qismlarga bo‘lish va qayta ishlatish madaniyatini shakllantiradi.

Mavzu bo‘yicha xulosa

Modul – qayta ishlatiladigan Python fayli, paket – modullar papkasi. import ning bir necha shakli bor. math, random, datetime va boshqa standart modullar ko‘p vazifani tayyor hal qiladi; tashqi paketlar pip bilan, virtual muhitda o‘rnatiladi. O‘z modulini yaratish kodni tuzilmalashtiradi.

Nazorat savollari

1. Modul va paket nima?
2. import ning qanday shakllari bor? Farqini tushuntiring.
3. from modul import * nima uchun tavsiya etilmaydi?
4. math modulining qaysi funksiyalarini bilasiz?
5. random modulida qanday funksiyalar bor? “Sonni top” o‘yinini tuzing.

6. datetime moduli nima uchun ishlatiladi?
7. collections.Counter nimani hal qiladi?
8. pip nima va u bilan paket qanday o'rnatiladi?
9. Virtual muhit nima uchun kerak va u qanday yaratiladi?
10. if __name__ == "__main__": nima vazifani bajaradi?

Adabiyotlar: [1], [9]

11-MAVZU. FAYLLAR BILAN ISHLASH. CSV VA JSON FORMATLARI. ISTISNOLARNI QAYTA ISHLASH (TRY/EXCEPT)

Reja:

1. Matn fayllarini o'qish va yozish.
2. with operatori va fayl rejimlari.
3. CSV formati.
4. JSON formati.
5. try/except va istisno turlari.
6. Kiritilgan ma'lumotni tekshirish.

Tayanch iboralar:

fayl, open(), rejim (r, w, a), with, encoding, read(), readline(), write(), CSV, JSON, csv moduli, json moduli, istisno (exception), try, except, else, finally, raise, ValueError, FileNotFoundError.

1. Matn fayllarini o'qish va yozish

Dastur tugagach o'zgaruvchilardagi ma'lumotlar yo'qoladi; doimiy saqlash uchun fayllardan foydalaniladi. Fayl open() funksiyasi bilan ochiladi:

Rejim	Ma'nosi	Izoh
"r"	O'qish (standart)	Fayl mavjud bo'lmasa FileNotFoundError
"w"	Yozish	Mavjud faylni o'chirib qayta yaratadi
"a"	Qo'shib yozish	Fayl oxiriga qo'shadi
"r+"	O'qish va yozish	Mavjud fayl uchun

2. with operatori

with fayl ishi tugagach uni avtomatik yopadi, xato bo'lsa ham:

```

with open("ismlar.txt", "w", encoding="utf-8") as f:
    f.write("Ali\n")
    f.write("Vali\n")

with open("ismlar.txt", "r", encoding="utf-8") as f:
    for qator in f:
        print(qator.strip())

with open("ismlar.txt", encoding="utf-8") as f:
    hammasi = f.read()          # butun matn

```

O‘qish usullari: `read()` – butun fayl, `readline()` – bitta qator, `readlines()` – qatorlar ro‘yxati, faylni qator-qator aylanish – eng tejamkor usul. O‘zbek harflari uchun `encoding="utf-8"` ni ko‘rsatish xato oldini oladi. Yozishda “\n” qo‘lda qo‘shiladi; “w” rejimi mavjud ma’lumotni o‘chirishini o‘quvchilarga alohida ta’kidlang.

3. CSV formati

CSV (comma-separated values) – jadval ma’lumotlarini matn ko‘rinishida saqlash formati; har bir qator – yozuv, ustunlar vergul (yoki nuqtali vergul) bilan ajratiladi.

```

import csv

with open("baholar.csv", "w", newline="", encoding="utf-8") as f:
    w = csv.writer(f)
    w.writerow(["ism", "ball"])
    w.writerow(["Ali", 85])
    w.writerow(["Vali", 92])

with open("baholar.csv", newline="", encoding="utf-8") as f:
    for qator in csv.DictReader(f):
        print(qator["ism"], int(qator["ball"]))

```

`csv.DictReader` har bir qatorni lug‘at sifatida beradi. CSV dan o‘qilgan barcha qiymatlar satr bo‘ladi – zarur bo‘lsa songa o‘zgartirish kerak. CSV fayllarni Excel va pandas (M19) ham o‘qiy oladi.

4. JSON formati

JSON (JavaScript Object Notation) – ichma-ich tuzilmali ma’lumotlarni (lug‘at va ro‘yxat) matn shaklida saqlash va almashish formati; veb-xizmatlar va sozlamalarda keng qo‘llanadi.

```
import json

data = {"ism": "Ali", "baholar": [5, 4, 5], "faol": True}
with open("talaba.json", "w", encoding="utf-8") as f:
    json.dump(data, f, ensure_ascii=False, indent=2)

with open("talaba.json", encoding="utf-8") as f:
    d = json.load(f)
    print(d["baholar"])
```

`ensure_ascii=False` o'zbek harflarini o'qiladigan holda saqlaydi. Python lug'ati va JSON bir-biriga juda o'xshaydi, lekin JSON da `True` o'rniga `true` yoziladi, kalitlar faqat satr bo'ladi.

5. try/except va istisno turlari

Dastur ishlashi paytidagi xatolar **istisno** (exception) deb ataladi. Ularni qayta ishlamasak, dastur to'xtaydi. try/except xatoni ushlab, dasturni to'g'ri davom ettirishga imkon beradi:

```
try:
    n = int(input("Son kiriting: "))
    print(10 / n)
except ValueError:
    print("Bu son emas!")
except ZeroDivisionError:
    print("Nolga bo'lib bo'lmaydi!")
else:
    print("Xatosiz bajarildi")
finally:
    print("Bu har doim bajariladi")
```

Keng uchraydigan istisnolar: `ValueError` (noto'g'ri qiymat), `TypeError` (noto'g'ri tur), `ZeroDivisionError`, `IndexError` (indeks chegaradan chiqdi), `KeyError` (lug'atda kalit yo'q), `FileNotFoundError`. Faqat kutilgan xatolarni ushlash kerak; yalang except: barcha xatolarni, shu jumladan dasturchi xatolarini ham "yutib yuboradi". Istisnoni o'zimiz ham `raise` bilan chaqirishimiz mumkin: `raise ValueError("Yosh manfiy bo'lmasligi kerak")`.

6. Kiritilgan ma'lumotni tekshirish

Real dasturlar foydalanuvchining noto'g'ri kiritishiga tayyor bo'lishi kerak:

```
def son_ol(xabar):
    while True:
        try:
            return int(input(xabar))
        except ValueError:
            print("Iltimos, butun son kiriting.")

yosh = son_ol("Yoshingiz: ")
```

Fayl bilan ishlaganda ham xato ehtimoli bor: fayl mavjud bo'lmasligi yoki ruxsat yo'qligi mumkin, shuning uchun `open()` try ichiga olinadi. Tekshiruv ikki yo'l

bilan amalga oshadi: oldindan shart bilan (if) va xato yuz berganidan keyin (try/except); Python da odatda ikkinchisi tabiiy deb hisoblanadi.

Metodik maslahat: dasturlarni “buzishga urinish” mashqi foydali: o‘quvchilar bir-birining dasturiga noto‘g‘ri qiymatlar kiritib, uning ishonchliligini tekshiradi.

Mavzu bo‘yicha xulosa

Fayllar ma’lumotni dastur tugagandan keyin ham saqlaydi; with fayl yopilishini kafolatlaydi. CSV jadvallar, JSON ichma-ich tuzilmalar uchun mos. try/except xatolarni boshqaradi, kiritilgan ma’lumotni tekshirish dastur ishonchliligini oshiradi.

Nazorat savollari

1. Fayl rejimlari “r”, “w” va “a” ning farqi nimada?
2. with operatorining afzalligi nima?
3. Faylni o‘qishning qanday usullari bor?
4. CSV formati nima va csv moduli bilan qanday ishlanadi?
5. JSON nima? json.dump va json.load nima qiladi?
6. Istisno (exception) nima?
7. try, except, else va finally ning vazifasini tushuntiring.
8. Qanday asosiy istisno turlarini bilasiz?
9. Nima uchun yalang except: dan qochish kerak?
10. Foydalanuvchi kiritgan butun sonni tekshiruvchi funksiyani yozing.

Adabiyotlar: [1], [2]

12-MAVZU. OB’YEKTGA YO‘NALTIRILGAN DASTURLASH: SINFLAR, OB’YEKTLAR, ATRIBUTLAR VA METODLAR

Reja:

1. OYD g‘oyasi va afzalliklari.
2. Sinf va ob’yekt.
3. Atributlar va metodlar.
4. __init__ va self.
5. Maxsus metodlar.
6. Hayotiy misollar bilan modellash.

Tayanch iboralar:

ob’yektga yo‘naltirilgan dasturlash (OYD), sinf (class), ob’yekt (instance), atribut, metod, konstruktor, __init__, self, __str__, __repr__, sinf atributi, ob’yekt atributi, modellash.

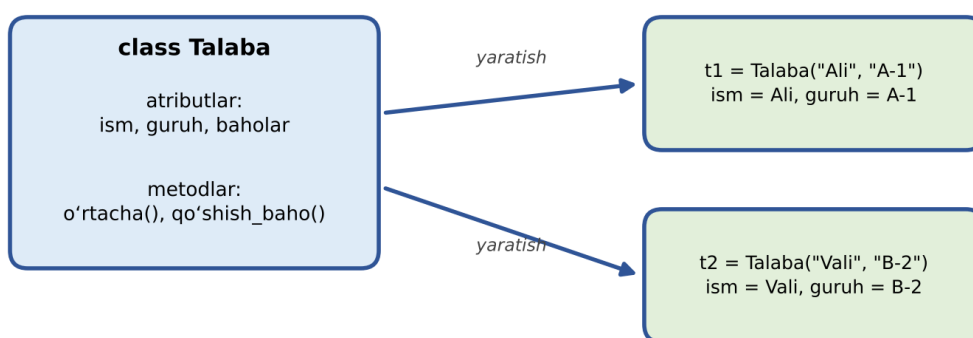
1. OYD g'oyasi va afzalliklari

Protsedurali dasturlashda dastur funksiyalar ketma-ketligi, ma'lumotlar esa alohida bo'ladi. OYD da ma'lumot (atributlar) va ular ustida amallar (metodlar) bitta butunlikka – **ob'yektga** birlashtiriladi. Dunyodagi narsalar (talaba, kitob, hisob raqam) ob'ektlar sifatida modellashiriladi.

Afzalliklari: dasturni real dunyoga yaqin tuzish, kodni qayta ishlatish, murakkab dasturlarni qismlarga ajratish va boshqarish osonligi.

2. Sinf va ob'yekt

Sinf – ob'ektlar uchun andoza (qolip); **ob'yekt** – sinfning aniq nusxasi (4-rasm). Masalan, “Talaba” – sinf, “Ali” va “Vali” – uning ob'ektlari.



Sinf – qolip (loyiha), ob'yekt – shu qolip asosida yaratilgan aniq nusxa

4-rasm. Sinf va ob'yekt: sinf – andoza, ob'yektlar – uning nusxalari

```
class Talaba:
    def __init__(self, ism, yosh):
        self.ism = ism
        self.yosh = yosh

    def tanishtir(self):
        return f"Men {self.ism}, {self.yosh} yoshdaman"

t1 = Talaba("Ali", 17)
t2 = Talaba("Vali", 16)
print(t1.tanishtir())
print(t2.ism)
```

Sinf nomi odatda katta harf bilan boshlanadi (PascalCase). Sinfdan ob'yekt yaratish – Talaba(“Ali”, 17) – konstruktorni chaqiradi.

3. Atributlar va metodlar

Atribut – ob'yektga tegishli ma'lumot, metod – ob'yekt ustida amal bajaruvchi funksiya. Ob'yekt atributlari har bir ob'yektda alohida; **sinf atributi** barcha ob'ektlar uchun umumiy:

```

class Hisob:
    foiz = 0.05                                # sinf atributi (umumiy)

    def __init__(self, egasi, balans=0):
        self.egasi = egasi                    # ob'yekt atributi
        self.balans = balans

    def qosh(self, summa):
        self.balans += summa

    def yech(self, summa):
        if summa > self.balans:
            print("Mablag' yetarli emas")
        else:
            self.balans -= summa

h = Hisob("Ali", 1000)
h.qosh(500)
h.yech(2000)
print(h.balans)                                # 1500

```

4. __init__ va self

`__init__` – ob'yekt yaratilganda avtomatik chaqiriladigan maxsus metod (konstruktor); u atributlarning boshlang'ich qiymatini beradi. **self** – metod chaqirilayotgan ob'yektning o'ziga ishora; metod ta'rifida birinchi parametr sifatida yoziladi, chaqirishda esa ko'rsatilmaydi: `t1.tanishtir()` aslida `Talaba.tanishtir(t1)` bilan teng. Boshlovchilar `self` ni tez-tez tushirib qoldiradi yoki atribut o'rniga oddiy lokal o'zgaruvchi yaratadi (`self.ism` o'rniga `ism`).

5. Maxsus metodlar

Ikkita pastki chiziq bilan o'ralgan “dunder” metodlar ob'yektning standart amallar bilan xulqini belgilaydi:

Metod	Qachon chaqiriladi	Misol
<code>__init__</code>	Ob'yekt yaratilganda	<code>Talaba("Ali", 17)</code>
<code>__str__</code>	<code>print()</code> va <code>str()</code> da	<code>print(t1)</code>
<code>__repr__</code>	Dasturchi uchun tasvir	Interaktiv rejimda
<code>__len__</code>	<code>len(ob'yekt)</code> da	<code>len(guruh)</code>
<code>__eq__</code>	<code>==</code> taqqoslashda	<code>t1 == t2</code>
<code>__add__</code>	<code>+</code> operatorida	<code>vektor1 + vektor2</code>

```

class Vektor:
    def __init__(self, x, y):
        self.x, self.y = x, y
    def __add__(self, other):
        return Vektor(self.x + other.x, self.y + other.y)
    def __str__(self):
        return f"({self.x}; {self.y})"

print(Vektor(1, 2) + Vektor(3, 4))            # (4; 6)

```

6. Hayotiy misollar bilan modellash

OYD ni o'rgatishda ob'yektlarni o'quvchi bilishi mumkin bo'lgan sohalardan tanlash samarali: kutubxona (Kitob, Kitobxon), maktab (O'quvchi, Sinf), o'yin (Qahramon, Dushman), do'kon (Mahsulot, Savat). Modellash bosqichlari: 1) ob'yektlarni aniqlash; 2) ularning atributlarini belgilash; 3) metodlarni belgilash; 4) kodlash va sinash. Bu bosqichlarni qog'ozda sinf diagrammasi (UML ning soddalashgan ko'rinishi) bilan boshlash mumkin.

Metodik maslahat: “Sinf – pechene qolipi, ob'yekt – pechene” kabi analogiya o'quvchilarda g'oyani yaxshi shakllantiradi. Avval bitta sinf, so'ng ob'yektlar ro'yxati (masalan, o'quvchilar ro'yxati) bilan ishlashga o'ting.

Mavzu bo'yicha xulosa

OYD ma'lumot va amallarni ob'yektda birlashtiradi. Sinf – andoza, ob'yekt – uning nusxasi. `__init__` boshlang'ich qiymat beradi, `self` joriy ob'yektga ishora qiladi. Maxsus metodlar ob'yekt xulqini standart amallar bilan moslashtiradi. Modellash hayotiy misollardan boshlanadi.

Nazorat savollari

1. OYD ning g'oyasi nimada va uning afzalliklari qanday?
2. Sinf va ob'yektning farqi nimada?
3. Atribut va metod nima?
4. `__init__` qanday vazifani bajaradi?
5. `self` nima va nima uchun kerak?
6. Ob'yekt atributi va sinf atributining farqi nimada?
7. `__str__` metodi nima uchun ishlatiladi?
8. `__add__` metodiga misol keltiring.
9. Kutubxona uchun Kitob sinfini loyihalang.
10. OYD ni o'rgatishda qanday analogiyalardan foydalaniladi?

Adabiyotlar: [1], [5]

13-MAVZU. OYD TAMOYILLARI: INKAPSULYATSIYA, VORISLIK, POLIMORFIZM

Reja:

1. Inkapsulyatsiya.
2. Vorislik va `super()`.
3. Polimorfizm.
4. Abstraksiya.
5. Kompozitsiya va vorislik.
6. OYD ni maktabda o'qitish masalalari.

Tayanch iboralar:

inkapsulyatsiya, ochiq va yopiq atribut, property, vorislik, asosiy (ota) sinf, hosila (bola) sinf, super(), polimorfizm, metodni qayta aniqlash (override), abstraksiya, kompozitsiya, “is-a” va “has-a” munosabati.

1. Inkapsulyatsiya

Inkapsulyatsiya – ma’lumotlar va ular ustida amallarni birlashtirish hamda ob’yekt ichki tuzilishiga tashqaridan to‘g‘ridan-to‘g‘ri murojaatni cheklash. Python da qat’iy yopiq atributlar yo‘q; kelishuv bo‘yicha pastki chiziq (`_nom`) “ichki foydalanish uchun”, ikkita pastki chiziq (`__nom`) esa nom o‘zgartirilishi (name mangling) orqali yopiqlikka yaqin himoya beradi.

```
class Hisob:
    def __init__(self, balans=0):
        self._balans = balans

    @property
    def balans(self):
        return self._balans

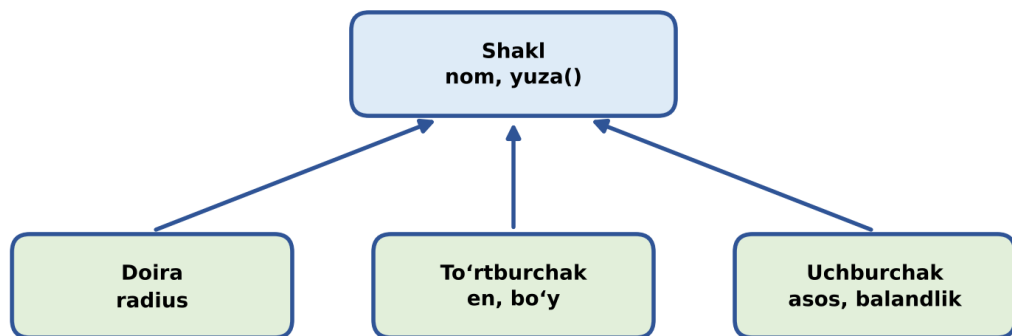
    def qosh(self, summa):
        if summa <= 0:
            raise ValueError("Summa musbat bo'lishi kerak")
        self._balans += summa

h = Hisob(100)
h.qosh(50)
print(h.balans)           # 150
```

`@property` atributni metod kabi himoyalab, o‘qish uchun oddiy atribut ko‘rinishida beradi. Foyda: noto‘g‘ri qiymat o‘rnatilishini oldini olish (masalan, manfiy balans).

2. Vorislik va super()

Vorislik – yangi sinf (hosila) mavjud sinf (asosiy) xususiyatlarini oladi va ularni kengaytiradi (5-rasm). “is-a” (“... bu ...”) munosabatida qo‘llanadi: Talaba – bu Shaxs.



Har bir avlod yuza() metodini o'zicha bajaradi - polimorfizm

strelka: "... dan meros oladi" (vorislik)

5-rasm. Vorislik ierarxiyasi

```

class Shaxs:
    def __init__(self, ism):
        self.ism = ism
    def tanishtir(self):
        return f"Men {self.ism}"

class Talaba(Shaxs):
    def __init__(self, ism, kurs):
        super().init(ism) # ota sinf konstruktori
        self.kurs = kurs
    def tanishtir(self):
        return super().tanishtir() + f", {self.kurs}-kurs talabasiman"

print(Talaba("Ali", 2).tanishtir())
  
```

`super()` ota sinf metodlariga murojaat qilish imkonini beradi. Python ko'plab vorislikni (`class C(A, B)`) ham qo'llaydi, lekin u murakkab; maktab darajasida bir bosqichli vorislik yetarli.

3. Polimorfizm

Polimorfizm – bir xil nomdagi metodni turli ob'yektlar o'ziga xos tarzda bajarishi:

```

class Hayvon:
    def ovoz(self):
        return "..."

class Mushuk(Hayvon):
    def ovoz(self):
        return "Miyov"

class It(Hayvon):
    def ovoz(self):
        return "Vov"

for h in [Mushuk(), It(), Hayvon()]:
    print(h.ovozi())
  
```

Dastur har bir ob'yekt turini bilmasdan, faqat ovoz() metodini chaqiradi. Python da “o’rdak tipizatsiyasi” (duck typing) tamoyili bor: ob'yektning turi emas, uning kerakli metodga egaligi muhim.

4. Abstraksiya

Abstraksiya – murakkab tizimning muhim jihatlarini ajratib, ikkinchi darajali tafsilotlarni yashirish. Foydalanuvchi mashinani “gaz, tormoz, rul” orqali boshqaradi, dvigatel ichini bilmaydi. Dasturda abstraksiya sinf interfeysi (ochiq metodlar) orqali ifodalanadi. Python da abc moduli yordamida abstrakt sinflar yaratish mumkin: ular to’g’ridan-to’g’ri ob’yekt yaratmaydi, hosila sinflar belgilangan metodlarni amalga oshirishi shart bo’ladi.

5. Kompozitsiya va vorislik

Kompozitsiya – ob’yekt boshqa ob’yektlarni o’z ichiga oladi (“has-a” munosabati): Avtomobil – Dvigatelga ega; Sinf – O’quvchilarga ega.

Belgi	Vorislik (“is-a”)	Kompozitsiya (“has-a”)
Munosabat	Talaba bu Shaxs	Avtomobil dvigatelga ega
Bog’lanish	Kuchli, ierarxiya	Erkinroq
Qachon	Haqiqiy “turi” munosabati bor bo’lsa	Qismlardan butun yig’ilsa

Umumiy tavsiya: shubha bo’lsa, vorislikdan ko’ra kompozitsiyani afzal ko’ring – u o’zgartirish uchun moslashuvchanroq. Vorislikni faqat haqiqiy “is-a” munosabatida ishlatish.

6. OYD ni maktabda o’qitish masalalari

OYD o’quvchilar uchun mavhum tuyuladi; shuning uchun: (1) konkret, tanish ob’yektlar bilan boshlash; (2) kichik sinflar (2–4 atribut, 2–3 metod); (3) sinf diagrammalarini qog’ozda chizish; (4) ob’yektlarni “rolli o’yin” orqali ko’rsatish (har bir o’quvchi – ob’yekt, atributlari yozilgan kartochka bilan); (5) avval OYD siz, so’ng OYD li yechimni solishtirish – afzallik ko’rinadi. Tipik xatolar: self ni unutish; ota sinf konstruktorini chaqirmaslik; vorislikni kerak bo’lmagan joyda ishlatish; sinf va ob’yektni aralashtirish. Murakkab mavzular (ko’plab vorislik, metaklasslar) maktab darajasidan tashqari.

Mavzu bo’yicha xulosa

OYD ning to’rt asosiy tamoyili: inkapsulyatsiya (ma’lumotni himoyalash), vorislik (qayta ishlatish va kengaytirish), polimorfizm (bir interfeys – turli xulq), abstraksiya (muhimini ajratish). “is-a” munosabati vorislik, “has-a” – kompozitsiya uchun. Maktabda OYD konkret misollar va kichik sinflar bilan o’rgatiladi.

Nazorat savollari

1. Inkapsulyatsiya nima va Python da u qanday amalga oshiriladi?
2. @property dekoratori nima uchun ishlatiladi?
3. Vorislik nima? “is-a” munosabatiga misol keltiring.
4. super() nima uchun kerak?
5. Polimorfizm nima? Hayvonlar misolida tushuntiring.
6. Metodni qayta aniqlash (override) nima?
7. Abstraksiya nimani anglatadi?
8. Kompozitsiya va vorislikning farqi nimada?
9. Qachon vorislik, qachon kompozitsiya tanlanadi?
10. OYD ni maktabda o‘rgatishda qanday qiyinchiliklar bor?

Adabiyotlar: [1], [5]

14-MAVZU. ALGORITMLAR MURAKKABLIGI (O-NOTATSIYA). SARALASH VA QIDIRISH ALGORITMLARI

Reja:

1. Algoritm samaradorligi.
2. O-notatsiya.
3. Chiziqli va ikkilik qidiruv.
4. Oddiy saralash usullari.
5. Tez saralash va birlashtirib saralash.
6. Eksperiment bilan solishtirish.

Tayanch iboralar:

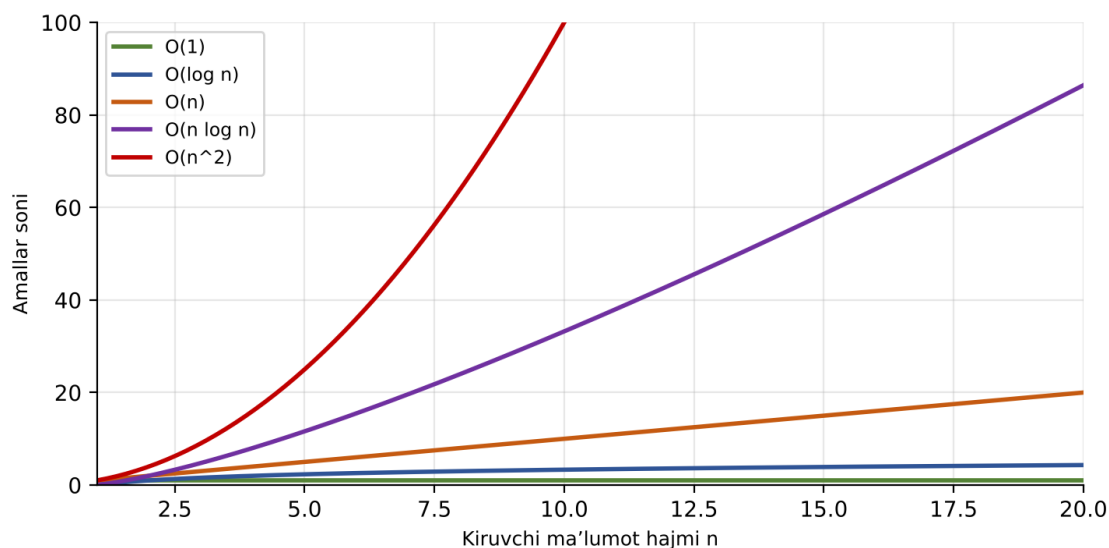
vaqt murakkabligi, xotira murakkabligi, O-notatsiya, chiziqli qidiruv, ikkilik qidiruv, pufakchali saralash, tanlash orqali saralash, qo‘shish orqali saralash, tez saralash (quicksort), birlashtirib saralash (mergesort), barqarorlik.

1. Algoritm samaradorligi

Bir masalani turli algoritmlar bilan yechish mumkin, lekin ular tezligi va xotira talabi bilan farq qiladi. Samaradorlik kirish ma’lumotlari hajmi (n) ortganda amallar soni qanday o‘sishi bilan baholanadi. Soniyalarda emas, **amallar soni** o‘lchanadi, chunki soniya kompyuterga bog‘liq.

2. O-notatsiya

O-notatsiya (Big O) algoritm bajaradigan amallar sonining n ga bog‘liq o‘shish tartibini ifodalaydi, doimiy ko‘paytuvchilar va kichik qo‘shiluvchilar tashlab yuboriladi (6-rasm).



6-rasm. Asosiy o‘rinish tartiblari: $O(1)$, $O(\log n)$, $O(n)$, $O(n \log n)$, $O(n^2)$

Belgi	Nomi	Misol
$O(1)$	Doimiy	Ro‘yxat elementiga indeks bo‘yicha murojaat
$O(\log n)$	Logarifmik	Ikkilik qidiruv
$O(n)$	Chiziqli	Ro‘yxatni bir marta ko‘rib chiqish, maksimumni topish
$O(n \log n)$	Chiziqli-logarifmik	Samarali saralash (birlashtirib, tez saralash)
$O(n^2)$	Kvadratik	Ichma-ich ikki sikl, oddiy saralash usullari
$O(2^n)$	Eksponensial	Fibonachchining sodda rekursiv hisobi

Masalan, $n = 1000$ da: $O(n)$ – 1000 ga yaqin amal, $O(n^2)$ – million, $O(\log n)$ – taxminan 10. n ortgani sari farq tubdan kattalashadi. Bir sikl – odatda $O(n)$, ichma-ich ikki sikl – $O(n^2)$.

3. Chiziqli va ikkilik qidiruv

Chiziqli qidiruv elementlarni birin-ketin ko‘radi: $O(n)$. **Ikkilik qidiruv** faqat **tartiblangan** ro‘yxatda ishlaydi: o‘rtadagi elementni tekshirib, qidiruv oralig‘ini har safar yarmiga qisqartiradi: $O(\log n)$.

```
def ikkilik_qidiruv(a, x):
    past, yuqori = 0, len(a) - 1
    while past <= yuqori:
        o = (past + yuqori) // 2
        if a[o] == x:
            return o
        elif a[o] < x:
            past = o + 1
        else:
            yuqori = o - 1
    return -1

print(ikkilik_qidiruv([1, 3, 5, 7, 9, 11], 7))    # 3
```

“Son o‘ylab ko‘r” o‘yini ikkilik qidiruv g‘oyasini ko‘rsatadi: 1 dan 1000 gacha sonni taxminan 10 ta savolda topish mumkin. Unplugged faoliyat sifatida juda qulay.

4. Oddiy saralash usullari

Pufakchali saralash qo‘shni elementlarni solishtirib, kerak bo‘lsa almashtiradi; tanlash orqali saralash har safar eng kichik elementni topib, o‘z o‘rniga qo‘yadi; qo‘shish orqali saralash har bir elementni tartiblangan qismdagi o‘z o‘rniga kiritadi. Uchalasi ham odatda $O(n^2)$.

```
def pufakchali(a):
    n = len(a)
    for i in range(n - 1):
        almashdi = False
        for j in range(n - 1 - i):
            if a[j] > a[j + 1]:
                a[j], a[j + 1] = a[j + 1], a[j]
                almashdi = True
        if not almashdi:
            break
    return a

print(pufakchali([5, 2, 9, 1, 6]))
```

Python da $a[j]$, $a[j+1] = a[j+1]$, $a[j]$ almashtirish uchun vaqtinchalik o‘zgaruvchi shart emas. Bu usullar ta’limiy qiymatga ega: algoritmik fikrlash, sikl va shartlarni mustahkamlaydi. Amaliyotda esa tayyor `sorted()` va `sort()` ishlatiladi.

5. Tez saralash va birlashtirib saralash

Bu algoritmlar “bo‘lib tashla va hukmronlik qil” g‘oyasiga asoslangan va rekursiv: masalani kichikroq qismlarga bo‘lib, ularni yechib, natijalarni birlashtiradi. Birlashtirib saralash (mergesort) ro‘yxatni teng ikkiga bo‘ladi, har ikki yarmini saralab, tartiblangan holda birlashtiradi; murakkabligi har doim $O(n \log n)$. Tez saralash (quicksort) tayanch elementni tanlab, elementlarni kichik va katta guruhlariga ajratadi; o‘rtacha $O(n \log n)$, eng yomon holatda $O(n^2)$.

```
def tez_saralash(a):
    if len(a) <= 1:
        return a
    t = a[len(a) // 2]
    kichik = [x for x in a if x < t]
    teng = [x for x in a if x == t]
    katta = [x for x in a if x > t]
    return tez_saralash(kichik) + teng + tez_saralash(katta)
```

Python ning o‘rnatilgan `sorted()` funksiyasi Timsort algoritmidan foydalanadi (birlashtirish va qo‘shish usullari aralashmasi), eng yomon holatda $O(n \log n)$.

6. Eksperiment bilan solishtirish

Nazariy xulosani eksperiment bilan tasdiqlash ta'sirchan: time modulida vaqtni o'lchab, turli hajmdagi ma'lumotlarda algoritmlarni solishtirish mumkin.

```
import time, random

a = [random.randint(1, 10**6) for _ in range(3000)]
t0 = time.perf_counter()
pufakchali(a.copy())
t1 = time.perf_counter()
sorted(a)
t2 = time.perf_counter()
print(f"pufakchali: {t1 - t0:.3f} s, sorted: {t2 - t1:.5f} s")
```

Natija kompyuterga bog'liq, lekin farq (yuzlab marta) deyarli har doim ko'rinadi. VisuAlgo (visualgo.net) kabi vositalar algoritmlarni qadamma-qadam animatsiya qiladi (M14 uchun [12]).

Metodik maslahat: O-notatsiyani maktab o'quvchilari uchun qat'iy matematik ta'rif bilan emas, "n ikki baravar ko'paysa, vaqt qancha o'sadi?" savoli bilan tushuntiring.

Mavzu bo'yicha xulosa

Algoritm samaradorligi amallar sonining n ga bog'liq o'sishi bilan baholanadi va O-notatsiyada ifodalanadi. Ikkilik qidiruv tartiblangan ma'lumotda $O(\log n)$ da ishlaydi. Oddiy saralash $O(n^2)$, samarali saralash $O(n \log n)$. Eksperimentlar nazariy xulosani tasdiqlaydi.

Nazorat savollari

1. Algoritm samaradorligi nimaga qarab baholanadi?
2. O-notatsiya nimani ifodalaydi?
3. $O(1)$, $O(n)$, $O(n^2)$ algoritmlarga misollar keltiring.
4. Chiziqli va ikkilik qidiruvning farqi nimada?
5. Ikkilik qidiruv uchun nima talab qilinadi?
6. Pufakchali saralash qanday ishlaydi va uning murakkabligi qanday?
7. Tanlash va qo'shish orqali saralash g'oyasini tushuntiring.
8. "Bo'lib tashla va hukmronlik qil" tamoyili nima?
9. Birlashtirib saralash va tez saralashni taqqoslang.
10. Algoritmlarni eksperiment bilan qanday solishtiramiz?

Adabiyotlar: [6], [12]

15-MAVZU. MA'LUMOTLAR TUZILMALARI: STEK, NAVBAT, BOG'LANGAN RO'YXAT, DARAXTLAR VA GRAFLAR ASOSLARI

Reja:

1. Stek va navbat.
2. Bog'langan ro'yxat.

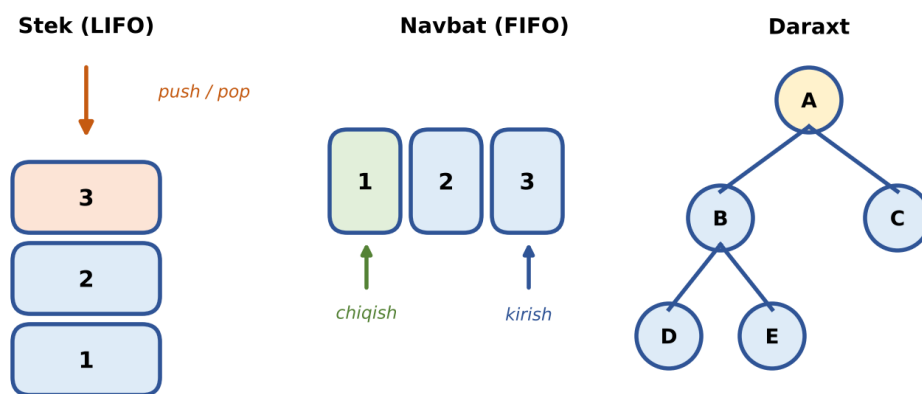
3. Daraxt tuzilmasi.
4. Graf va ularni tasvirlash.
5. BFS va DFS g'oyasi.
6. Tuzilma tanlash mezonlari.

Tayanch iboralar:

ma'lumotlar tuzilmasi, stek (LIFO), navbat (FIFO), push, pop, enqueue, dequeue, deque, bog'langan ro'yxat, tugun, ikkilik daraxt, ildiz, bargi, graf, uchlar va qirralar, qo'shnilik ro'yxati, BFS, DFS.

1. Stek va navbat

Ma'lumotlar tuzilmasi – ma'lumotlarni saqlash va ular bilan ishlash usuli (7-rasm). **Stek** “oxirgi kirgan – birinchi chiqadi” (LIFO) tamoyilida ishlaydi (taxta ustiga taxta qo'yilgandek). **Navbat** “birinchi kirgan – birinchi chiqadi” (FIFO) tamoyilida (do'konda navbat kabi).



Tuzilma tanlash masala talabiga bog'liq: oxirgi kirgan birinchi chiqadimi yoki birinchi kirgan?

7-rasm. Stek, navbat va daraxt tuzilmalari

Stek Python ro'yxati bilan amalga oshiriladi:

```
stek = []
stek.append(1)      # push
stek.append(2)
stek.append(3)
print(stek.pop())   # 3 (oxirgi kirgan)

from collections import deque
navbat = deque()
navbat.append("A")   # enqueue
navbat.append("B")
print(navbat.popleft()) # A (birinchi kirgan)
```

Navbatni oddiy ro'yxat bilan (pop(0)) amalga oshirish samarasiz, chunki hamma elementlar siljiydi; shuning uchun collections.deque ishlatiladi. Stek qo'llanishi: qavslar balansini tekshirish, brauzerda “Orqaga”, rekursiya chaqiruvlari

(M9), “Bekor qilish” (undo) funksiyasi. Navbat: printer navbati, vazifalarni rejalashtirish, BFS algoritmi.

```
def qavslar_togri(s):
    stek = []
    juft = {"(": ")", "[": "]", "{": "}"
    for h in s:
        if h in "([{":
            stek.append(h)
        elif h in juft:
            if not stek or stek.pop() != juft[h]:
                return False
    return not stek

print(qavslar_togri("{[()]}" ), qavslar_togri("([)]"))    # True False
```

2. Bog‘langan ro‘yxat

Bog‘langan ro‘yxat tugunlardan iborat bo‘lib, har bir tugun qiymat va keyingi tugunga havolani saqlaydi. Boshiga element qo‘shish $O(1)$, lekin indeks bo‘yicha murojaat $O(n)$.

```
class Tugun:
    def __init__(self, qiymat):
        self.qiymat = qiymat
        self.keyingi = None

class Royxat:
    def __init__(self):
        self.bosh = None
    def boshiga_qosh(self, q):
        t = Tugun(q)
        t.keyingi = self.bosh
        self.bosh = t
    def chiqar(self):
        t = self.bosh
        while t:
            print(t.qiymat, end=" -> ")
            t = t.keyingi
        print("None")
```

Python ro‘yxati (list) aslida dinamik massiv; bog‘langan ro‘yxat OYD, havolalar va xotira g‘oyalarini tushuntirish uchun muhim ta’limiy tuzilma.

3. Daraxt tuzilmasi

Daraxt – ierarxik tuzilma: ildiz (yuqori tugun), tugunlar, bolalar va barglar (bolasi yo‘q tugunlar). Ikkilik daraxtda har bir tugunning ko‘pi bilan ikkita bolasi bo‘ladi. Daraxtlar fayl tizimi, oila daraxti, HTML tuzilmasi va qaror qabul qilishda uchraydi. Ikkilik qidiruv daraxti (BST) da chap bolaning qiymati ota qiymatidan kichik, o‘ng bolaniki katta; bu qidirishni tezlashtiradi (muvozanatli holda $O(\log n)$). Daraxt bo‘ylab yurish tabiiy ravishda rekursiya bilan yoziladi:

```
class Tugun:
    def __init__(self, q):
        self.q, self.chap, self.ong = q, None, None

    def yurish(t):
        if t:
            yurish(t.chap)
            print(t.q, end=" ")
            yurish(t.ong)
```

4. Graf va ularni tasvirlash

Graf – uchlar (tugunlar) va ularni bog‘lovchi qirralar to‘plami: ijtimoiy tarmoq, shaharlar orasidagi yo‘llar, xarita. Graf **qo‘shnilik ro‘yxati** (Python lug‘ati) yoki qo‘shnilik matritsasi (ikki o‘lchovli ro‘yxat) ko‘rinishida beriladi:

```
graf = {
    "A": ["B", "C"],
    "B": ["A", "D"],
    "C": ["A", "D"],
    "D": ["B", "C"],
}
```

Qo‘shnilik ro‘yxati xotirani tejaydi (siyrak graflar uchun), matritsa esa ikki uch orasida qirra borligini tez tekshiradi.

5. BFS va DFS g‘oyasi

Grafda yurishning ikki asosiy usuli: **BFS** (kenglik bo‘yicha qidiruv) – navbat yordamida, boshlang‘ich uchdan “qatlam-qatlam” yuradi; eng qisqa yo‘lni (qirralar soni bo‘yicha) topadi. **DFS** (chuqurlik bo‘yicha qidiruv) – stek yoki rekursiya yordamida bir yo‘lni oxirigacha borib, qaytadi.

```
from collections import deque

def bfs(graf, bosh):
    korilgan = {bosh}
    navbat = deque([bosh])
    while navbat:
        u = navbat.popleft()
        print(u, end=" ")
        for v in graf[u]:
            if v not in korilgan:
                korilgan.add(v)
                navbat.append(v)

bfs(graf, "A")          # A B C D
```

Bu mavzu maktab o‘quvchilari uchun yuqori sinflarda va olimpiadaga tayyorgarlikda ahamiyatli; asosiy g‘oyalar vizual tarzda (VisuAlgo) ko‘rsatiladi [12].

6. Tuzilma tanlash mezonlari

Tuzilma	Asosiy amal	Murakkablik	Qo'llanishi
Ro'yxat (list)	Indeks bo'yicha murojaat	O(1)	Ketma-ketlik, jadval
Stek	Oxirga qo'shish/olish	O(1)	Qavslar, undo
Navbat (deque)	Oxiriga qo'shish, boshidan olish	O(1)	BFS, vazifa navbati
Lug'at (dict)	Kalit bo'yicha topish	O(1) o'rtacha	Chastota, indekslash
To'plam (set)	Tegishlilik tekshiruvi	O(1) o'rtacha	Noyoblik
Bog'langan ro'yxat	Boshiga qo'shish	O(1)	Ta'limiy, maxsus hollar

Murakkabliklar odatiy (o'rtacha) holat uchun; aniq qiymatlar uchun docs.python.org dagi TimeComplexity sahifasini tekshiring.

Metodik maslahat: stek va navbatni jismoniy modellashtiring (kitoblar uyumi, o'quvchilar navbati) – keyin kodga o'ting.

Mavzu bo'yicha xulosa

Stek (LIFO) va navbat (FIFO) eng muhim chiziqli tuzilmalar; Python da ular list va deque bilan amalga oshiriladi. Bog'langan ro'yxat, daraxt va graf murakkab munosabatlarni ifodalaydi. BFS va DFS graf bo'ylab yurish usullari. Tuzilmani tanlash asosiy amallar samaradorligiga bog'liq.

Nazorat savollari

1. Stek va navbatning farqini LIFO va FIFO tamoyillari orqali tushuntiring.
2. Navbat uchun nima uchun collections.deque tavsiya etiladi?
3. Qavslar balansini stek yordamida tekshirish algoritmini tushuntiring.
4. Bog'langan ro'yxat list dan nimasi bilan farq qiladi?
5. Daraxt tuzilmasi atamalarini (ildiz, barg, bola) tushuntiring.
6. Ikkilik qidiruv daraxti qanday xususiyatga ega?
7. Graf nima va u qanday ifodalanadi?
8. BFS va DFS ning farqi nimada?
9. Qaysi masalalarda stek, qaysilarida navbat ishlatiladi?
10. Tuzilma tanlashda qanday mezonlar hisobga olinadi?

Adabiyotlar: [6], [12]

16-MAVZU. C++ DASTURLASH TILIGA KIRISH: DASTUR TUZILISHI, MA'LUMOT TURLARI, BOSHQARUV OPERATORLARI

Reja:

1. C++ tilining xususiyatlari va qo'llanish sohalari.

2. Dastur tuzilishi va kompilyatsiya.
3. Statik turlar va o'zgaruvchilar.
4. Kiritish-chiqarish: cin va cout.
5. Shart va sikl operatorlari.
6. Python bilan taqqoslash.

Tayanch iboralar:

C++, kompilyator (g++), #include, main(), namespace, int, long long, double, char, bool, string, cin, cout, endl, if, switch, for, while, statik tiplash, nuqtali vergul.

1. C++ tilining xususiyatlari va qo'llanish sohalari

C++ – kompilyatsiya qilinadigan, statik tiplashgan, yuqori samarali umumiy maqsadli til. U tizim dasturlari, o'yin dvigatellari, operatsion tizim komponentlari, ishlashi tez bo'lishi kerak bo'lgan ilovalar va **olimpiada dasturlash** uchun keng qo'llanadi. Python ga nisbatan dastur tezroq ishlaydi va xotira ustidan ko'proq nazorat beriladi, lekin yozish murakkabroq. Maktab o'quvchilari uchun C++ asosan algoritmik olimpiadalarga tayyorgarlikda va kompyuter xotirasi hamda turlar tushunchasini chuqurlashtirishda muhim.

2. Dastur tuzilishi va kompilyatsiya

```
#include <iostream>
using namespace std;

int main() {
    cout << "Salom, dunyo!" << endl;
    return 0;
}
```

#include <iostream> kiritish-chiqarish kutubxonasini ulaydi; main() – dastur boshlanadigan funksiya; { } – blok chegarasi; har bir buyruq nuqtali vergul (;) bilan tugaydi. return 0 dastur muvaffaqiyatli tugaganini bildiradi. Kompilyatsiya va ishga tushirish (g++ kompilyatori misolida):

```
g++ -std=c++17 -O2 dastur.cpp -o dastur
./dastur
```

Muhit sifatida Code::Blocks, VS Code (g++ bilan), CLion yoki onlayn kompilyatorlar ishlatiladi. Kompilyator bayroqlari va versiyasi muhitga qarab farq qilishi mumkin; hujjatlar en.cppreference.com va kompilyator qo'llanmasida.

3. Statik turlar va o'zgaruvchilar

C++ da o'zgaruvchi turi **e'lon qilinadi** va u o'zgarmaydi:

Tur	Hajmi (odatda)	Izoh
int	4 bayt	Butun son, taxminan $\pm 2 \cdot 10^9$ gacha
long long	8 bayt	Katta butun son, taxminan $\pm 9 \cdot 10^{18}$ gacha

Tur	Hajmi (odatda)	Izoh
double	8 bayt	Haqiqiy son
char	1 bayt	Bitta belgi ('a')
bool	1 bayt	true yoki false
string	O'zgaruvchan	Satr (<string> sarlavhasi)

Muhim farq: Python da butun sonlar cheksiz, C++ da int to'lib ketadi (overflow). Olimpiada masalalarida katta sonlar uchun long long ishlatish kerak. Turlar hajmi platformaga bog'liq bo'lishi mumkin.

```
int a = 5;
long long katta = 10000000000LL * 3;
double narx = 12.5;
char belgi = 'A';
bool tayyor = true;
string ism = "Ali";
```

Butun sonlar bo'linishi: $7 / 2$ ifodasi C++ da 3 (butun bo'lish), $7 / 2.0$ esa 3.5; qoldiq uchun % operatori ishlatiladi. Bu Python dagi // va / dan farqli.

4. Kiritish-chiqarish: cin va cout

```
#include <iostream>
using namespace std;

int main() {
    int a, b;
    cin >> a >> b;
    cout << "Yig'indi: " << a + b << endl;
    return 0;
}
```

cin >> o'zgaruvchi qiymat o'qiydi (bo'shliq yoki yangi qator bilan ajratilgan), cout << chiqaradi. endl yangi qator qo'yadi va buferni tozalaydi; katta hajmdagi chiqarishda "\n" tezroq ishlaydi (M17).

5. Shart va sikl operatorlari

Sintaksis Python dan farq qiladi: shart qavs ichida, blok figurali qavslarda, chekinish esa faqat o'qish qulayligi uchun.

```

int x;
cin >> x;
if (x > 0) {
    cout << "musbat";
} else if (x < 0) {
    cout << "manfiy";
} else {
    cout << "nol";
}

for (int i = 1; i <= 5; i++) {
    cout << i << " ";
}

int n = 12, s = 0;
while (n > 0) {
    s += n % 10;
    n /= 10;
}
switch (x) {
    case 1: cout << "bir"; break;
    case 2: cout << "ikki"; break;
    default: cout << "boshqa";
}

```

for ning uch qismi: boshlang‘ich qiymat; davom etish sharti; qadam. Mantiqiy operatorlar: &&, ||, !. switch ichida break ni unutish keyingi case larning ham bajarilishiga olib keladi.

6. Python bilan taqqoslash

Mavzu	Python	C++
Tur e‘loni	Yo‘q (dinamik)	Majburiy (statik)
Blok	Chekinish	{ }
Buyruq oxiri	Qator oxiri	;
Shart	if x > 0:	if (x > 0) { }
Sikl	for i in range(5):	for (int i = 0; i < 5; i++)
Chiqarish	print(a)	cout << a;
Butun son	Cheksiz	Cheklangan (int, long long)
Xatolarni topish	Bajarilish paytida	Ko‘pi kompilyatsiyada

Metodik maslahat: Python dan C++ ga o‘tayotgan o‘quvchiga bir xil masalani ikkala tilda yozdirish samarali: g‘oya o‘sha-o‘sha, faqat yozuv shakli farq qiladi. Bu “til algoritmining yozilish usuli” degan muhim fikrni mustahkamlaydi.

Mavzu bo‘yicha xulosa

C++ tez, statik tiplashgan, kompilyatsiya qilinadigan til; olimpiada dasturlashda keng qo‘llanadi. Dastur main() funksiyasidan boshlanadi, turlar e‘lon qilinadi, kiritish-chiqarish cin va cout bilan bajariladi. Python bilan taqqoslash o‘quvchiga tillar farqini va umumiy algoritmik g‘oyalarni ko‘rsatadi.

Nazorat savollari

1. C++ ning asosiy xususiyatlari nimalardan iborat?
2. C++ dasturining minimal tuzilishini yozing.
3. Dastur qanday kompilyatsiya qilinadi va ishga tushiriladi?
4. C++ ning asosiy ma'lumot turlari qaysilar?
5. int to'lib ketishi nima va nima uchun long long ishlatiladi?
6. C++ da $7 / 2$ va $7 / 2.0$ natijalari nima?
7. cin va cout qanday ishlaydi?
8. for sikli qanday qismlardan iborat?
9. switch operatorida break ning ahamiyati qanday?
10. Python va C++ ni asosiy belgilari bo'yicha taqqoslang.

Adabiyotlar: [7], [10]

17-MAVZU. C++ DA MASSIVLAR, FUNKSIYALAR VA STL KONTEYNERLARI (VECTOR, MAP). OLIMPIADA MASALALARINI YECHISHDA C++

Reja:

1. Massivlar.
2. Funksiyalar va parametrlar.
3. vector va map konteynerlari.
4. sort va STL algoritmlari.
5. Tezkor kiritish-chiqarish.
6. Olimpiada masalasi namunasi.

Tayanch iboralar:

massiv, indeks, funksiya prototipi, havola bilan uzatish (&), STL, vector, map, set, pair, sort(), iterator, push_back(), size(), ios::sync_with_stdio, vaqt chegarasi, xotira chegarasi.

1. Massivlar

Massiv – bir xil turdagi elementlarning ketma-ket joylashgan to'plami; o'lchami oldindan belgilanadi, indeks 0 dan boshlanadi.

```
int a[5] = {3, 1, 4, 1, 5};
int n = 5;
long long yigindi = 0;
for (int i = 0; i < n; i++) {
    yigindi += a[i];
}
cout << yigindi;

int jadval[3][3];           // ikki o'lchovli massiv
```

Python dan farqli ravishda C++ massiv chegarasini avtomatik tekshirmaydi: `a[10]` ga murojaat qilish aniqlanmagan xulq (undefined behavior) beradi – dastur xato natija berishi yoki to‘xtashi mumkin. Shu sababli indeks chegarasini o‘zingiz nazorat qilish kerak. Olimpiada masalalarida massiv o‘lchamini chegaradan biroz katta qilib e‘lon qilish odat.

2. Funksiyalar va parametrlar

```
int kvadrat(int x) {
    return x * x;
}

void almashtir(int &a, int &b) {    // havola orqali
    int t = a; a = b; b = t;
}

int main() {
    int p = 3, q = 5;
    almashtir(p, q);                // p = 5, q = 3
    cout << kvadrat(p);
}
```

Funksiya turi (qaytariladigan qiymat), nomi va parametrlar turlari ko‘rsatiladi; hech narsa qaytarmasa void. Parametr odatda **qiymat bo‘yicha** uzatiladi (nusxa), & bilan esa **havola bo‘yicha**: funksiya asl o‘zgaruvchini o‘zgartiradi. Funksiya ishlatilishidan oldin e‘lon qilingan (prototip) yoki ta’riflangan bo‘lishi kerak.

3. vector va map konteynerlari

STL (Standard Template Library) tayyor ma’lumotlar tuzilmalari va algoritmlar to‘plami. **vector** – Python ro‘yxatiga o‘xshash dinamik massiv; **map** – Python lug‘atiga o‘xshash kalit-qiymat tuzilmasi (kalitlar tartiblangan).

```
#include <vector>
#include <map>
#include <string>

vector<int> v;
v.push_back(5);
v.push_back(2);
cout << v.size() << " " << v[0];
for (int x : v) cout << x << " ";

map<string, int> chastota;
string soz;
while (cin >> soz) {
    chastota[soz]++;
}
for (auto &p : chastota) {
    cout << p.first << " " << p.second << "\n";
}
```

Python	C++ (STL)	Izoh
list	vector	Dinamik massiv
dict	map / unordered_map	map tartiblangan, unordered_map tezroq o'rtacha
set	set / unordered_set	Noyob elementlar
deque	deque	Ikki tomonlama navbat
tuple/juftlik	pair	p.first, p.second

4. sort va STL algoritmlari

```
#include <algorithm>

vector<int> v = {5, 2, 9, 1};
sort(v.begin(), v.end()); // o'sish bo'yicha
sort(v.begin(), v.end(), greater<int>()); // kamayish bo'yicha
reverse(v.begin(), v.end());
int eng_katta = *max_element(v.begin(), v.end());
bool bor = binary_search(v.begin(), v.end(), 5); // tartiblangan bo'lsa
```

sort() odatda $O(n \log n)$ vaqtda ishlaydi. Maxsus tartib uchun taqqoslash funksiyasi yoki lambda beriladi. Bu Python dagi sorted(key=...) ga mos keladi.

5. Tezkor kiritish-chiqarish

Ko'p ma'lumot o'qiladigan olimpiada masalalarida cin/cout ni tezlashtirish zarur:

```
ios::sync_with_stdio(false);
cin.tie(nullptr);
```

Bu ikki satr main() boshiga qo'yiladi; shundan keyin cin va C dagi scanf/printf ni aralashtirib ishlatmaslik, endl o'rniga "\n" yozish tavsiya etiladi.

6. Olimpiada masalasi namunasi

Masala. n ta butun son berilgan. Eng katta va eng kichik sonlar orasidagi farqni toping. Kirish: birinchi qatorda n ($1 \leq n \leq 10^5$), ikkinchi qatorda n ta son (modulda 10^9 dan oshmaydi). Chiqish: bitta son.

```
#include <bits/stdc++.h>
using namespace std;

int main() {
    ios::sync_with_stdio(false);
    cin.tie(nullptr);
    int n;
    cin >> n;
    vector<long long> a(n);
    for (auto &x : a) cin >> x;
    long long mx = *max_element(a.begin(), a.end());
    long long mn = *min_element(a.begin(), a.end());
    cout << mx - mn << "\n";
    return 0;
}
```

Masalani yechish bosqichlari: shartni diqqat bilan o‘qish; chegaralarni tahlil qilish (farq $2 \cdot 10^9$ ga yetishi mumkin – long long kerak); algoritmnı tanlash; kodlash; chekka holatlar ($n = 1$) bilan sinash. Onlayn tizimlar (Codeforces va boshqalar) [13] masalalarnı avtomatik tekshiradi va “Wrong Answer”, “Time Limit Exceeded”, “Runtime Error” kabi hukmlar beradi. bits/stdc++.h sarlavhasi GCC ga xos va barcha kompilyatorlarda mavjud emas.

Metodik maslahat: olimpiada tayyorgarligida har bir masalani avval qog‘ozda yechish, so‘ng kodlash odatini shakllantiring. Eng ko‘p xato – shartni noto‘g‘ri o‘qish va chekka holatlarnı e‘tiborsiz qoldirish.

Mavzu bo‘yicha xulosa

C++ massivlari chegarani tekshirmaydi; funksiyalar qiymat yoki havola orqali parametr oladi. STL vector, map, set va sort kabi tayyor vositalarnı beradi. Tezkor kiritish-chiqarish olimpiadada muhim. Masalani yechishda chegaralar, tur va chekka holatlarnı tahlil qilish talab qilinadi.

Nazorat savollari

1. C++ da massiv qanday e‘lon qilinadi va indekslanadi?
2. Massiv chegarasidan chiqish nimaga olib keladi?
3. Qiymat va havola bo‘yicha parametr uzatishning farqi nimada?
4. vector nima va u Python ro‘yxatidan nimasi bilan farq qiladi?
5. map qanday ishlatiladi? So‘zlar chastotasini hisoblang.
6. sort() funksiyasi qanday ishlatiladi?
7. Nima uchun ios::sync_with_stdio(false) ishlatiladi?
8. Olimpiada masalasini yechishning asosiy bosqichlari qaysilar?
9. “Time Limit Exceeded” hukmi nimani bildiradi va qanday bartaraf etiladi?
10. Python dagi list, dict, set ga C++ dagi qaysi tuzilmalar mos keladi?

Adabiyotlar: [7], [10]

18-MAVZU. GRAFIK INTERFEYSLI DASTURLAR: TKINTER KUTUBXONASI. TURTLE GRAFIKASI ORQALI DASTURLASHNI O‘QITISH

Reja:

1. Hodisaga asoslangan dasturlash.
2. Tkinter va uning vidjetlari.
3. Joylashtirish va hodisalar.
4. Turtle grafikasi.
5. Turtle bilan o‘qitish metodikasi.
6. Loyiha namunasi.

Tayanch iboralar:

grafik interfeys (GUI), hodisa (event), tugma, vidjet (widget), Tk, Label, Button, Entry, Canvas, pack, grid, mainloop, callback, Turtle, forward, left, penup, pendown.

1. Hodisaga asoslangan dasturlash

Shu paytgacha dasturlar yuqoridan pastga bajarilardi (konsol). Grafik dasturlarda esa dastur **hodisalarni** (tugma bosish, matn kiritish, sichqoncha harakati) kutadi va ularga javob beradi. Asosiy g‘oya: dastur oynasi “asosiy sikl” (mainloop) da turadi, foydalanuvchi harakati esa mos funksiyani (callback) chaqiradi.

2. Tkinter va uning vidjetlari

Tkinter – Python bilan birga keladigan standart GUI kutubxonasi (ba’zi Linux tizimlarida alohida o‘rnatish talab qilinishi mumkin). Asosiy vidjetlar:

Vidjet	Vazifasi	Misol
Tk	Asosiy oyna	root = Tk()
Label	Matn yoki rasm ko‘rsatish	Label(root, text=”Salom”)
Button	Tugma	Button(root, text=”OK”, command=f)
Entry	Bir qatorli matn kiritish	Entry(root)
Text	Ko‘p qatorli matn	Text(root)
Canvas	Chizish maydoni	Canvas(root, width=300, height=200)
Checkbutton, Radiobutton	Belgilash va tanlash	-

3. Joylashtirish va hodisalar

Vidjetlar oynada pack() (ketma-ket), grid() (jadval) yoki place() (aniq koordinata) bilan joylashtiriladi. Misol – oddiy kalkulyator qismi:

```
from tkinter import *

def hisobla():
    try:
        a = float(e1.get())
        b = float(e2.get())
        natija.config(text=f"Yig'indi: {a + b}")
    except ValueError:
        natija.config(text="Son kiriting!")

root = Tk()
root.title("Qo'shish")
e1 = Entry(root); e1.grid(row=0, column=0)
e2 = Entry(root); e2.grid(row=0, column=1)
Button(root, text="Hisobla", command=hisobla).grid(row=1, column=0,
columnspan=2)
natija = Label(root, text="")
natija.grid(row=2, column=0, columnspan=2)
root.mainloop()
```

Tugma command parametriga funksiya **nomi** beriladi (qavssiz!); hisobla() deb yozish funksiyani darhol chaqirib yuboradi – keng tarqalgan xato. Klaviatura va sichqoncha hodisalari bind() orqali ulanadi: canvas.bind("<Button-1>", funksiya).

4. Turtle grafikasi

Turtle – “toshbaqa” yordamida chizish kutubxonasi (standart kutubxona tarkibida): toshbaqa buyruqlar bilan harakatlanib, izida chiziq qoldiradi. Boshlovchilar uchun juda ko‘rgazmali, chunki natija darhol ko‘rinadi.

```
import turtle
t = turtle.Turtle()
for _ in range(4):
    t.forward(100)
    t.left(90)

t.penup(); t.goto(-150, 0); t.pendown()
t.color("red")
for i in range(36):
    t.circle(50)
    t.left(10)
turtle.done()
```

Asosiy buyruqlar: forward(n), backward(n), left(a), right(a), penup(), pendown(), goto(x, y), color(), circle(r), begin_fill() va end_fill().

5. Turtle bilan o‘qitish metodikasi

Turtle sikl, funksiya va parametr tushunchalarini o‘rgatishning ajoyib vositasi: kvadrat – sikl; yulduzcha, spiral – parametrli sikl; ko‘pburchak funksiyasi – parametrlar; rekursiv daraxt va fraktallar – rekursiya. Misol:

```
def kopburchak(n, uzunlik):
    for _ in range(n):
        t.forward(uzunlik)
        t.left(360 / n)

def daraxt(uzunlik):
    if uzunlik < 10:
        return
    t.forward(uzunlik)
    t.left(30); daraxt(uzunlik * 0.7)
    t.right(60); daraxt(uzunlik * 0.7)
    t.left(30); t.backward(uzunlik)
```

Metodik afzalliklar: natijaning vizualligi, xatolarning koʻzga koʻrinishi (chizma notoʻgʻri boʻlsa, darhol sezilgan), ijodiy yondashuv imkoniyati (“oʻz naqshingni yarat”). Kamchiligi: katta loyihalar uchun yetarli emas, shuning uchun u oʻqitishning boshlangʻich bosqichida qoʻllanadi.

6. Loyiha namunasi

Mini-loyiha gʻoyalari: “Sonni top” oʻyini (Tkinter), oddiy kalkulyator, vazifalar roʻyxati (to-do) – faylga saqlash bilan, Turtle bilan geometrik sanʼat, sodda “ilon” oʻyini (Canvas). Loyiha bosqichlari: gʻoya, interfeys eskizi, kod, sinash, taqdimot. Katta GUI loyihalarda koʻpincha OYD (sinf sifatida oyna) qoʻllanadi, bu M12 ni mustahkamlaydi.

Metodik maslahat: GUI dasturlarida xatolar koʻpincha “hech narsa boʻlmaydi” koʻrinishida namoyon boʻladi (ayniqsa command bilan). Debugging uchun print() orqali tekshirishni oʻrgating.

Mavzu boʻyicha xulosa

GUI dasturlari hodisalarga javob beradi; Tkinter vidjetlar (Label, Button, Entry, Canvas) va joylashtirish usullarini (pack, grid) beradi. Turtle grafikasi sikl, funksiya va rekursiyani vizual oʻrgatadi. Mini-loyihalar oʻquvchilarda motivatsiya va ijodkorlikni oshiradi.

Nazorat savollari

1. Hodisaga asoslangan dasturlash nima?
2. mainloop() nima vazifani bajaradi?
3. Tkinter ning asosiy vidjetlarini sanang.
4. pack(), grid() va place() ning farqi nimada?
5. Tugma command parametriga funksiya qanday beriladi? Qavssiz yozilishining sababi nima?
6. Turtle kutubxonasining asosiy buyruqlarini sanang.
7. Turtle yordamida kvadrat va yulduzcha chizish dasturini yozing.
8. Turtle sikl va rekursiyani oʻrgatishda qanday yordam beradi?
9. Turtle ning metodik afzallik va kamchiliklari qaysilar?

10. Tkinter da mini-loyiha g'oyasi va bosqichlarini taklif qiling.

Adabiyotlar: [1], [9]

19-MAVZU. MA'LUMOTLARNI QAYTA ISHLASH VA VIZUALLASHTIRISH: NUMPY, PANDAS VA MATPLOTLIB ASOSLARI

Reja:

1. NumPy massivlari.
2. pandas: Series va DataFrame.
3. Ma'lumotlarni o'qish va filtrlash.
4. matplotlib diagrammalari.
5. Ma'lumotlar tahlili jarayoni.
6. Maktab uchun misollar.

Tayanch iboralar:

NumPy, massiv (ndarray), vektorlashtirish, pandas, Series, DataFrame, read_csv, filtrlash, guruhlash (groupby), matplotlib, grafik, diagramma, ma'lumotlar tahlili, tozalash, vizuallashtirish.

1. NumPy massivlari

NumPy – sonli hisoblar uchun kutubxona; asosiy ob'yekti ndarray – bir xil turdagi elementlarning samarali massivi. U pip install numpy bilan o'rnatiladi. Asosiy afzallik – **vektorlashtirish**: amallar butun massivga sikl yozmasdan qo'llanadi.

```
import numpy as np

a = np.array([1, 2, 3, 4, 5])
print(a * 2)           # [ 2  4  6  8 10]
print(a.mean(), a.max()) # 3.0 5
print(a[a > 2])         # [3 4 5] ( mantiqiy filtr)
m = np.arange(12).reshape(3, 4) # 3x4 matritsa
print(m.sum(axis=0))       # ustunlar yig'indisi
```

Ro'yxatga nisbatan NumPy massivi ancha tez ishlaydi va hisob-kitoblar ixcham yoziladi. Ro'yxat [1, 2] * 2 elementlarni takrorlaydi, massiv esa har bir elementni ikkiga ko'paytiradi – bu farqni tushuntirish muhim.

2. pandas: Series va DataFrame

pandas jadval ma'lumotlari (Excel yoki CSV kabi) bilan ishlash kutubxonasi. **Series** – bir o'lchovli belgilangan ma'lumot, **DataFrame** – ikki o'lchovli jadval (qatorlar va nomlangan ustunlar).

```
import pandas as pd

df = pd.DataFrame({
    "ism": ["Ali", "Vali", "Gul", "Nodira"],
    "sinf": [9, 9, 10, 10],
    "ball": [85, 92, 78, 95],
})
print(df.head())
print(df["ball"].mean())
print(df.describe())
```

3. Ma'lumotlarni o'qish va filtrlash

```
df = pd.read_csv("baholar.csv", encoding="utf-8")
print(df.shape, df.columns.tolist())
yuqori = df[df["ball"] >= 90] # filtr
tartib = df.sort_values("ball", ascending=False)
ortacha = df.groupby("sinf")["ball"].mean() # guruhlash
df["baho"] = df["ball"].apply(lambda b: 5 if b >= 86 else 4 if b >= 71 else 3)
df.to_csv("natija.csv", index=False)
```

Real ma'lumotlarda bo'sh qiymatlar va xatolar bo'ladi: `df.isna().sum()` bo'sh qiymatlarni sanaydi, `df.dropna()` yoki `df.fillna(0)` bilan ishlanadi. Ma'lumotni **tozalash** tahlilning katta qismini egallaydi.

4. matplotlib diagrammalari

matplotlib – grafik va diagrammalar chizish kutubxonasi:

```
import matplotlib.pyplot as plt

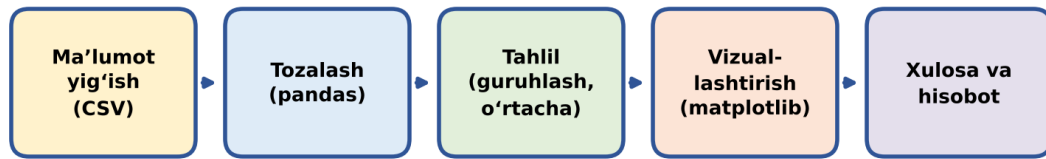
ortacha.plot(kind="bar")
plt.title("Sinflar bo'yicha o'rtacha ball")
plt.xlabel("Sinf")
plt.ylabel("Ball")
plt.savefig("diagramma.png", dpi=150)
plt.show()
```

Diagramma turlari: chiziqli (vaqt bo'yicha o'zgarish), ustunli (toifalarni solishtirish), gistogramma (taqsimot), sochilish (ikki o'zgaruvchi orasidagi bog'liqlik), doiraviy (ulushlar). To'g'ri diagramma tanlash va sarlavha, o'q nomlari, birliklarni ko'rsatish o'quvchilarga ma'lumotlar savodxonligini beradi.

5. Ma'lumotlar tahlili jarayoni

Tahlil bosqichma-bosqich jarayon (8-rasm).

Ma'lumotlar tahlili bosqichlari



Maktab uchun misol: o'quvchilar baholari jadvalidan sinf o'rtachasi va diagramma

8-rasm. Ma'lumotlar tahlili jarayoni

Bosqichlar: 1) savol qo'yish; 2) ma'lumot yig'ish (CSV, so'rovnoma, ochiq ma'lumotlar); 3) tozalash; 4) tahlil (o'rtacha, guruhlash, korrelyatsiya); 5) vizuallashtirish; 6) xulosa va taqdimot. Xulosalar ma'lumotga asoslanishi, taxmin va natijani farqlash, namuna kichik bo'lganda ehtiyotkor bo'lish muhim.

6. Maktab uchun misollar

Maktab o'quvchilari uchun mavzular: sinf o'quvchilari bo'yi va vazni; haftalik ob-havo; maktab bufetidagi sotuvlar; sport natijalari; ijtimoiy tarmoqdan foydalanish vaqti (anonim so'rovnoma); mahalliy ochiq ma'lumotlar. Shaxsiy ma'lumotlarni (ism, telefon) yig'ishda maxfiylikka rioya qilinadi: anonim yoki o'ylab topilgan ma'lumotdan foydalaniladi. Kutubxonalarning versiyalari va imkoniyatlari o'zgarib turadi, shuning uchun rasmiy hujjatlarni (docs.python.org, pandas va matplotlib saytlari) tekshirish tavsiya etiladi [8], [9].

Metodik maslahat: avval 10–15 qatorli kichik ma'lumotni qo'lda tahlil qildirib, so'ng pandas ni ko'rsating: o'quvchi kutubxona nima qilishini tushunadi.

Mavzu bo'yicha xulosa

NumPy massivlarni samarali hisoblaydi, pandas jadval ma'lumotlarini o'qish, tozalash va guruhlashga imkon beradi, matplotlib natijani vizuallashtiradi. Tahlil savol qo'yishdan xulosagacha bo'lgan jarayon; maktabda anonim yoki ochiq ma'lumotlardan foydalaniladi.

Nazorat savollari

1. NumPy ning ro'yxatdan afzalligi nimada?
2. Vektorlashtirish nima?
3. Series va DataFrame ning farqini tushuntiring.
4. CSV faylni pandas bilan qanday o'qiymiz?
5. Ma'lumotni filtrlash va tartiblash qanday amalga oshiriladi?
6. groupby nima qiladi?
7. Qanday diagramma turlarini bilasiz va ular qachon qo'llanadi?

8. Ma'lumotlar tahlilining asosiy bosqichlarini sanang.
9. Ma'lumotni tozalash nima uchun kerak?
10. Maktab uchun tahlil mavzusi tanlashda qanday talablar hisobga olinadi?

Adabiyotlar: [8], [9]

20-MAVZU. DASTURIY TA'MINOT ISHLAB CHIQISH TEKNOLOGIYALARI: GIT VA GITHUB, TESTLASH, KOD SIFATI. SUN'IY INTELLEKT YORDAMCHILARIDAN DASTURLASHDA FOYDALANISH

Reja:

1. Versiyalarni boshqarish.
2. Git asosiy buyruqlari.
3. GitHub va hamkorlik.
4. Testlash va debugging.
5. Kod sifati va uslub.
6. Sun'iy intellekt yordamchilari va akademik halollik.

Tayanch iboralar:

versiyalarni boshqarish, Git, repozitoriy, commit, branch, merge, pull request, GitHub, README, .gitignore, birlik testi (unit test), assert, unittest, pytest, debugging, PEP 8, docstring, refaktoring, SI yordamchisi, akademik halollik.

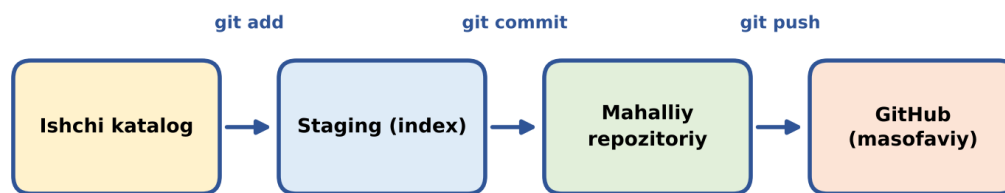
1. Versiyalarni boshqarish

Dasturlash loyihasi vaqt o'tishi bilan o'zgaradi. Versiyalarni boshqarish tizimi (Git) har bir o'zgarishni saqlaydi, oldingi holatga qaytish, bir necha kishi bilan parallel ishlash va o'zgarishlar tarixini ko'rish imkonini beradi. "dastur_final_2_oxirgi_TOZA.py" kabi fayllar o'rniga bitta loyiha va uning tarixi bo'ladi.

2. Git asosiy buyruqlari

Git ish oqimi uchta maydondan iborat: ishchi papka -> tayyorlov (staging) -> repozitoriy (9-rasm).

Git ish oqimi



git pull / git clone - o'zgarishlarni GitHub dan olish

har bir commit - loyiha tarixining nomlangan "surati"

9-rasm. Git ish oqimi: ishchi papka, staging, mahalliy va masofaviy repozitoriy

Buyruq	Vazifasi
git init	Yangi repozitoriy yaratadi
git status	Joriy holatni ko'rsatadi
git add fayl	Faylni staging ga qo'shadi
git commit -m "xabar"	O'zgarishlarni tarixga saqlaydi
git log --oneline	Commit tarixini ko'rsatadi
git branch yangi, git switch yangi	Tarmoq yaratish va o'tish
git merge yangi	Tarmoqni birlashtirish
git clone manzil	Masofaviy repozitoriydan nusxa olish
git push, git pull	Masofaviy repozitoriyga yuborish va undan olish

```

git init
git add dastur.py
git commit -m "Birinchi versiya: salomlashish dasturi"
git switch -c yangi-funksiya
  
```

Commit xabari qisqa va ma'noli bo'lishi kerak ("Xatoni tuzatdim" emas, "Bo'sh kiritishda ValueError ushlanadi"). Tarkibida parol va maxfiy kalitlarni saqlamaslik muhim; keraksiz fayllar .gitignore da ko'rsatiladi.

3. GitHub va hamkorlik

GitHub – Git repozitoriylarni onlayn saqlash va hamkorlikda ishlash platformasi. Asosiy tushunchalar: repozitoriy (loyiha), README.md (loyiha tavsifi), issue (vazifa yoki xato), pull request (o'zgarishni asosiy tarmoqqa qo'shishni taklif qilish va ko'rib chiqish), fork (boshqa loyihadan nusxa olish). Maktab sharoitida GitHub o'quvchi portfoliosi (loyihalar to'plami) va guruh ishlarini tashkil qilish uchun ishlatiladi. Hujjatlar: docs.github.com [11]. Voyaga yetmagan o'quvchilar uchun hisob ochish yosh cheklovlari va maktab siyosatiga bog'liq: o'qituvchi platforma shartlarini tekshirishi hamda maxfiy ma'lumotlarni joylamasligini nazorat qilishi kerak.

Yaxshi README tarkibi: loyiha nomi va maqsadi; o'rnatish va ishga tushirish yo'riqnomasi; foydalanish misoli; muallif(lar); litsenziya.

4. Testlash va debugging

Test – dasturning kutilgan natija berishini tekshiruvchi kod. Eng oddiy shakl – assert:

```
def yuza(a, b):  
    return a * b  
  
assert yuza(3, 4) == 12  
assert yuza(0, 5) == 0
```

Standart unittest moduli yoki pytest kutubxonasi yordamida testlar tizimli yoziladi:

```
# test_yuza.py (pytest)  
from geometriya import yuza  
  
def test_oddiy():  
    assert yuza(3, 4) == 12  
  
def test_nol():  
    assert yuza(0, 5) == 0
```

Testlar chekka holatlarni qamrab olishi kerak: bo'sh kirish, nol, manfiy qiymatlar, juda katta qiymatlar. **Debugging** – xatoni topish va tuzatish: (1) xatoni takrorlash; (2) xato xabarini o'qish; (3) gipoteza qilish; (4) print() yoki debugger (Thonny, VS Code) bilan tekshirish; (5) tuzatish va qayta sinash. “Rezina o'rdak” usuli: muammoni ovoz chiqarib (o'rdakka) tushuntirish ko'pincha yechimni topadi.

5. Kod sifati va uslub

Yaxshi kod – nafaqat to'g'ri ishlaydigan, balki o'qilishi oson kod. Python uslub qo'llanmasi PEP 8: o'zgaruvchi va funksiya nomlari kichik harf va pastki chiziq bilan (hisobla_yuza), sinflar PascalCase, 4 bo'shliqli chekinish, mazmunli nomlar, ortiqcha takrorlarni funksiyaga ajratish (DRY – “Don't Repeat Yourself”), docstring va qisqa izohlar (“nima”dan ko'ra “nima uchun”ni tushuntiruvchi). **Refactoring** – kodning xulqini o'zgartirmasdan tuzilmasini yaxshilash. Kod sharhi (code review) – hamkasb yoki o'qituvchi kodni o'qib, tavsiyalar berish; o'quvchilar bir-biriga sharh yozishni o'rganishi foydali. Linter va formatlovchilar (masalan, ruff, flake8, black) uslubni avtomatik tekshiradi; ularni o'rnatish va sozlash hujjatlarga qarab bajariladi.

6. Sun'iy intellekt yordamchilari va akademik halollik

Sun'iy intellekt (SI) yordamchilari kod yozishda tushuntirish, xatoni topish, test g'oyalari berish va kodni takomillashtirishda yordam bera oladi. Ammo ular

xato, eskirgan yoki xavfsiz bo'lmagan kod berishi mumkin, shuning uchun natija doim tekshiriladi, tushuniladi va sinaladi.

Ma'qul foydalanish	Ma'qul bo'lmagan foydalanish
Tushunmagan xato xabarini tushuntirishni so'rash	Topshiriq yechimini tayyor holda olib, o'zinikidek topshirish
Yozilgan kodni tekshirtirish va takomillashtirish bo'yicha tavsiya olish	Kodni tushunmasdan nusxalash
Test uchun chekka holatlar g'oyasini so'rash	Maxfiy yoki shaxsiy ma'lumotlarni yuborish
Yangi tushunchaga misol so'rash va o'zi mashq qilish	SI javobini tekshirmasdan ishonish

Akademik halollik qoidalari: topshiriqni bajarishda SI dan foydalanish ruxsat etilganmi-yo'qmi, o'qituvchi oldindan aniq belgilaydi; foydalanilgan bo'lsa, qaysi qismda va qanday foydalanilgani ochiq ko'rsatiladi; o'quvchi o'z kodini tushuntirib bera olishi (himoya) kerak. Shaxsiy ma'lumotlar, parollar va kalitlar SI xizmatlariga yuborilmaydi. O'qituvchi uchun muhim vazifa: topshiriq shaklini o'zgartirish (og'zaki himoya, jarayonni ko'rsatish, kodni sharhlash), shunda SI "yechim yozib berish" vositasi emas, o'rganish yordamchisi bo'ladi. Yaxshi pedagogik amaliyot: o'quvchi avval o'zi urinadi, so'ng SI dan "qanday yaxshilash mumkin?" deb so'raydi va farqni tahlil qiladi.

Metodik maslahat: sinf bilan "SI yordamchisini ishlatish qoidalari" shartnomasini birgalikda tuzing: bu qoidalar o'quvchilar tomonidan ishlab chiqilsa, ularga rioya qilish ehtimoli yuqori bo'ladi.

Mavzu bo'yicha xulosa

Git loyiha tarixini boshqaradi, GitHub hamkorlik va portfolio uchun xizmat qiladi. Testlash dastur to'g'riligini tizimli tekshiradi, debugging xatolarni bosqichma-bosqich topadi. PEP 8 va yaxshi odatlar kod sifatini oshiradi. SI yordamchilari kuchli vosita, lekin ularni tanqidiy va akademik halollik qoidalariga rioya qilgan holda ishlatish zarur.

Nazorat savollari

1. Versiyalarni boshqarish nima uchun kerak?
2. Git ning asosiy buyruqlarini sanang va ularning vazifasini tushuntiring.
3. Commit nima va yaxshi commit xabari qanday bo'ladi?
4. Branch va merge nima?
5. GitHub da pull request va issue nima?
6. Yaxshi README qanday qismlardan iborat?
7. Birlik testi nima? assert va pytest ga misol keltiring.
8. Debugging bosqichlarini tushuntiring.

9. PEP 8 ning asosiy talablari qaysilar?

10. SI yordamchisidan ma'qul va ma'qul bo'lmagan foydalanish misollarini keltiring; akademik halollik qoidalari qanday?

Adabiyotlar: [11], [14]

V. AMALIY MASHG'ULOT MATERIALLARI

Amaliy mashg'ulotlar kompyuter sinfida o'tkaziladi. Jami 35 ta amaliy mashg'ulot (70 soat) Python va C++ tillarida dasturlash ko'nikmalarini bosqichma-bosqich shakllantirishga, shuningdek maktab o'qituvchisi uchun zarur bo'lgan masalalar tizimini tuzish va tushuntirish ko'nikmasiga qaratilgan. Har bir mashg'ulot quyidagi tuzilishga ega: mavzu, maqsad, kutilayotgan natijalar (TN), kerakli jihozlar, qisqacha nazariy ma'lumot, ishni bajarish tartibi (kod namunalari bilan), mustaqil bajarish uchun topshiriqlar (10 ta variant) va nazorat savollari.

Amaliy mashg'ulotni o'tkazish tartibi. Mashg'ulot odatda quyidagi bosqichlardan iborat: (1) o'tilgan mavzuni qisqa takrorlash va 3–5 ta blits-savol; (2) namunaviy kodni o'qituvchi bilan birgalikda tahlil qilish (PRIMM usuli: avval natijani taxmin qilish, so'ng ishga tushirish va o'zgartirish); (3) ishni bajarish tartibi bo'yicha qadamlarni mustaqil bajarish; (4) o'z varianti bo'yicha topshiriqni yechish; (5) juftlikda kod sharhi (code review) va xulosa.

Ishni topshirish va himoya qilish. Variant raqami o'qituvchi tomonidan beriladi yoki tinglovchining ro'yxatdagi tartibiga ko'ra belgilanadi. Ish natijasi (dastur fayli, test ma'lumotlari, qisqa izoh) elektron shaklda topshiriladi va nazorat savollariga og'zaki javob bilan himoya qilinadi. Tinglovchi o'z kodining har bir qismini tushuntira olishi shart. Kod namunalari majmuada Python 3 ning zamonaviy versiyalari va C++17 ni nazarda tutib yozilgan; muhit versiyasi va kutubxonalar o'zgargan hollarda rasmiy hujjatlar (docs.python.org, en.cppreference.com) tekshiriladi.

Amaliy ishlar uchun umumiy baholash mezonlari:

Baho	Mezon
5 (a'lo)	Dastur to'g'ri ishlaydi, chekka holatlar hisobga olingan, kod o'qilishi oson (nomlar, chekinish, izohlar); nazorat savollariga asosli javob berilgan
4 (yaxshi)	Dastur asosan to'g'ri ishlaydi; ayrim chekka holatlar hisobga olinmagan yoki uslubda kamchiliklar bor; savollarga javob asosan to'g'ri
3 (qoniqarli)	Dastur qisman ishlaydi yoki o'qituvchi yordami bilan tuzatilgan; kodni tushuntirishda qiyinchilik bor
2 (qoniqarsiz)	Dastur ishlamaydi yoki mustaqil bajarilmagan; kodni tushuntira olmaydi

1-AMALIY MASHG'ULOT. PYTHON VA VS CODE/THONNY MUHITINI O'RNATISH, BIRINCHI DASTURNI YOZISH

Maqsad: Python va Thonny (yoki VS Code) muhitini o'rnatish, sozlash va birinchi dasturni yozib ishga tushirish ko'nikmasini hosil qilish.

Kutilayotgan natijalar (TN1): tinglovchi muhitni o'rnatadi, versiyani tekshiradi, dasturni saqlaydi va ishga tushiradi, xato xabarini o'qib tuzatadi.

Kerakli jihozlar: kompyuter, Internet (oʻrnatish uchun), python.org dan olingan oʻrnatuvchi, Thonny yoki VS Code.

Qisqacha nazariy maʼlumot

Python oʻrnatishda Windows da “Add Python to PATH” belgisi qoʻyiladi; versiya buyruq satridan python –version bilan tekshiriladi. Thonny dasturni qadamma-qadam bajarish va oʻzgaruvchilarni koʻrsatish imkonini beradi. Dastur fayli .py kengaytmasi bilan saqlanadi. Rasmiy yoʻriqnoma docs.python.org saytida; interfeys va versiyalar oʻzgarishi mumkin.

Ishni bajarish tartibi

1-qadam. Oʻrnatish. Python ning joriy barqaror 3-versiyasi va Thonny oʻrnatiladi (yoki mavjud oʻrnatma tekshiriladi).

2-qadam. Versiyani tekshirish. Buyruq satrida versiya tekshiriladi, interaktiv rejimda bir nechta ifoda sinab koʻriladi.

```
python --version
python -c "print(2 + 3 * 4)"
```

3-qadam. Birinchi dastur. Yangi fayl yaratilib, saqlanadi (salom.py) va ishga tushiriladi.

```
ism = input("Ismingiz nima? ")
print("Salom,", ism, "!")
```

4-qadam. Qasddan xato. Dasturdagi qavsni olib tashlab, xato xabari oʻqiladi: qaysi qator, qanday xato turi (SyntaxError).

5-qadam. Tuzatish va debug. Xatoni tuzatib, Thonny da dastur qadamma-qadam (F7) bajariladi; oʻzgaruvchilar oynasi kuzatiladi.

Mustaqil bajarish uchun topshiriqlar

Variant tanlanib, mashgʻulotda koʻrsatilgan tartibda dastur yoziladi. Topshiriq natijasi: oʻrnatilgan muhit va ishlaydigan birinchi dastur.

Variant	Topshiriq mavzusi
1	Ism va yoshni soʻrab, “N yildan keyin ... yoshda boʻlasiz” xabarini chiqaruvchi dastur
2	Oʻz maktabingiz vizit kartasini (5 qator) chiqaruvchi dastur
3	Ikki sonni soʻrab ularning yigʻindisini chiqaruvchi dastur
4	Aylana radiusi boʻyicha uzunligini chiqaruvchi dastur ($\pi = 3.14$)
5	Doʻkondagi narx va miqdor boʻyicha umumiy summani hisoblovchi dastur
6	Harorat Selsiydan Farangeytga oʻtkazuvchi dastur
7	Daqiqani soat va daqiqaga ajratuvchi dastur
8	Ismni uch marta qaytarib, ramka ichida chiqaruvchi dastur
9	Oʻquvchining uchta bahosining oʻrtachasini chiqaruvchi dastur

Variant	Topshiriq mavzusi
10	Kitob sahifalari va kunlik o'qish bo'yicha tugatish muddatini hisoblovchi dastur

Nazorat savollari

1. Python muhitini o'rnatishda nimalarga e'tibor beriladi?
2. IDE nima va Thonny ning afzalligi nimada?
3. Python dasturi fayli qanday kengaytma bilan saqlanadi?
4. Xato xabarida qanday ma'lumotlar bo'ladi?
5. Interaktiv rejim va skript rejimi qanday farqlanadi?

Adabiyotlar: [2], [9]

2-AMALIY MASHG'ULOT. O'ZGARUVCHILAR, MA'LUMOT TURLARI VA ARIFMETIK AMALLARGA DOIR MASALALAR

Maqsad: o'zgaruvchilar, ma'lumot turlari va arifmetik amallarga oid masalalarni yechish ko'nikmasini shakllantirish.

Kutilayotgan natijalar (TN7): tinglovchi 8–10 ta arifmetik dastur yozadi, //, % va ** operatorlarini to'g'ri qo'llaydi, ifodalar ustuvorligini hisobga oladi.

Kerakli jihozlar: kompyuter, Python muhiti.

Qisqacha nazariy ma'lumot

Arifmetik operatorlar: +, -, *, / (natija float), // (butun bo'lish), % (qoldiq), (daraja). Ustuvorlik: **avval**, so'ng *, /, //, %, keyin +, -. Butun sonning raqamlarini ajratish uchun // va % dan foydalaniladi: 345 // 100 = 3, 345 % 10 = 5. float sonlar taxminiy saqlanadi, natijani round() bilan yaxlitlash mumkin.

```
n = 345
yuzlar = n // 100
onlar = n // 10 % 10
birlar = n % 10
print(yuzlar, onlar, birlar, round(3.14159, 2))
```

Ishni bajarish tartibi

1-qadam. Namuna tahlili. Raqamlarni ajratuvchi namunaviy dastur ishga tushirilib, har bir ifoda natijasi qo'lda tekshiriladi.

2-qadam. Oddiy masalalar. To'rtburchak yuzi va perimetri, aylana uzunligi kabi 3 ta masala yechiladi.

3-qadam. Butun bo'lish va qoldiq. Soniyalarni soat, daqiqa va soniyaga ajratuvchi dastur yoziladi.

```
s = int(input("Soniya: "))
print(s // 3600, "soat", s % 3600 // 60, "daqiqa", s % 60, "soniya")
```

4-qadam. Raqamlar bilan ishlash. Uch xonali sonning raqamlar yig'indisi va teskarisi topiladi.

5-qadam. float aniqligi. $0.1 + 0.2$ natijasi ko'rib, `round()` bilan yaxlitlash sinab ko'riladi; kuzatuvlar yozib qo'yiladi.

Mustaqil bajarish uchun topshiriqlar

Variant tanlanib, mashg'ulotda ko'rsatilgan tartibda dastur yoziladi. Topshiriq natijasi: arifmetik masalalar yechimlari (8–10 ta dastur).

Variant	Topshiriq mavzusi
1	Uchburchak uch tomoni bo'yicha perimetri va yarim perimetrini hisoblash
2	Kvadratning yuzi bo'yicha tomonini topish (** 0.5)
3	Uch xonali sonning raqamlari yig'indisi va ko'paytmasi
4	Sutkani soat, daqiqa va soniyaga o'tkazish
5	Kilometrni mil va metrga o'tkazish
6	Ikki sonning o'rta arifmetigi va o'rta geometrigi
7	To'rt xonali sonning raqamlarini alohida chiqarish
8	Narx, chegirma foizi va yakuniy summani hisoblash
9	Konfetlarni n ta bolaga teng bo'lib berish: har biriga nechta va nechta qoladi
10	Shar hajmi va sirti radius bo'yicha

Nazorat savollari

1. / va // operatorlari qanday farqlanadi?
2. % operatori qanday masalalarda ishlatiladi?
3. Uch xonali sonning raqamlarini qanday ajratamiz?
4. Operatorlar ustuvorligi qanday?
5. float sonlarda nima uchun noaniqlik paydo bo'ladi?

Adabiyotlar: [2], [4]

3-AMALIY MASHG'ULOT. KIRITISH-CHIQUARISH VA TURLARNI O'ZGARTIRISHGA DOIR MASALALAR

Maqsad: `input()` va `print()` imkoniyatlarini, turlarni o'zgartirishni o'rganish va tipik tur xatolarini tahlil qilish.

Kutilayotgan natijalar (TN7): tinglovchi kiritilgan ma'lumotni kerakli turga o'zgartiradi, f-string va `sep/end` bilan chiqarishni formatlaydi, tur xatolarini topadi va tuzatadi.

Kerakli jihozlar: kompyuter, Python muhiti.

Qisqacha nazariy ma'lumot

`input()` har doim satr qaytaradi; son kerak bo'lsa, `int()` yoki `float()` qo'llanadi. Bir qatorda bir nechta son: `a, b = map(int, input().split())`. `print()` ning `sep` va `end` parametrlari ajratuvchi va qator oxirini belgilaydi. f-string (`f'{x:.2f}'`) formatlash uchun qulay. Noto'g'ri qiymat (`int("abc")`) `ValueError` beradi.

```
a, b = map(int, input().split())
print(a, b, sep=" | ", end="!\n")
print(f"{a} / {b} = {a / b:.3f}")
```

Ishni bajarish tartibi

1-qadam. PRIMM: taxmin. Berilgan 4 ta qisqa kod parchasi natijasi avval taxmin qilinadi, so'ng ishga tushirib tekshiriladi.

2-qadam. Tur xatosi. `"5" + "3"` va `int("5") + int("3")` farqi kuzatiladi; xatolar jadvaliga yoziladi.

3-qadam. Bir qatordan o'qish. `map(int, input().split())` bilan ikki-uch sonni o'qiydigan dastur yoziladi.

4-qadam. Formatlash. f-string yordamida jadval ko'rinishidagi chiqarish tuziladi (tekislash va aniqlik).

5-qadam. Xatoni qayta ishlash. Noto'g'ri kiritishda dastur qanday to'xtashi ko'riladi (`ValueError`); keyingi mavzuda (`try/except`) tuzatiladi.

Mustaqil bajarish uchun topshiriqlar

Variant tanlanib, mashg'ulotda ko'rsatilgan tartibda dastur yoziladi. Topshiriq natijasi: kiritish-chiqarish dasturlari va tur xatolari tahlili.

Variant	Topshiriq mavzusi
1	Ism, familiya va yoshni bitta qatarga chiroyli formatda chiqarish
2	Ikki sonning yig'indisi, ayirmasi, ko'paytmasi va bo'linmasini jadval shaklida chiqarish
3	Uch sonni bitta qatorda o'qib, o'rtachasini 2 xona aniqlikda chiqarish
4	Tovarning nomi, narxi, miqdorini o'qib chek ko'rinishida chiqarish
5	Sana (kun, oy, yil) ni uch xil formatda (dd.mm.yyyy, yyyy-mm-dd, matn) chiqarish
6	Satr va sonni kiritib, satrni son marta takrorlash
7	Vaqtni soat:daqiqa formatida nol bilan to'ldirib chiqarish (07:05)
8	Ikkita sonni o'qib, ularning qiymatlarini almashtirib chiqarish
9	O'quvchi ismini va 3 fan balini o'qib, qisqa hisobot chiqarish
10	Sonni o'nlik, ikkilik va o'n oltilik sanoq sistemalarida chiqarish

Nazorat savollari

1. `input()` qanday turdagi qiymat qaytaradi?

2. map(int, input().split()) nima qiladi?
3. print() ning sep va end parametrlari nima uchun kerak?
4. f-string da kasr aniqligi qanday beriladi?
5. Tur xatolarining oldini olish usullari qanday?

Adabiyotlar: [2]

4-AMALIY MASHG‘ULOT. SHARTLI OPERATORLARGA DOIR MASALALAR

Maqsad: shartli operatorlar (if, if-else, elif) yordamida tarmoqlanuvchi dasturlar yozish ko‘nikmasini shakllantirish.

Kutilayotgan natijalar (TN7): tinglovchi 8 ta dastur yozadi, shartlarni to‘g‘ri tuzadi va chekka holatlar bilan sinaydi.

Kerakli jihozlar: kompyuter, Python muhiti.

Qisqacha nazariy ma’lumot

Shartli operator if shart: blokni bajaradi; else qolgan holatlarni, elif qo‘shimcha shartlarni qamrab oladi. Shartlar yuqoridan pastga tekshiriladi va birinchi rost shartning bloki bajariladi. Taqqoslash operatorlari: ==, !=, <, >, <=, >=. Blok chekinish bilan ajratiladi.

```
x = int(input())
if x % 2 == 0:
    print("juft")
else:
    print("toq")
```

Ishni bajarish tartibi

1-qadam. Blok-sxema. Berilgan masala uchun blok-sxema chiziladi, so‘ng kodlanadi.

2-qadam. Juft-toq, musbat-manfiy. Oddiy ikki tarmoqli dasturlar yoziladi.

3-qadam. Uch tarmoqli masala. Sonning ishorasi (musbat, manfiy, nol) aniqlanadi.

4-qadam. Baholash dasturi. Ball bo‘yicha 5/4/3/2 baho chiqaruvchi dastur yoziladi.

```
ball = int(input())
if ball >= 86: print(5)
elif ball >= 71: print(4)
elif ball >= 56: print(3)
else: print(2)
```

5-qadam. Chekka holatlar. 55, 56, 70, 71, 85, 86 ballar bilan test jadvali tuziladi va natijalar solishtiriladi.

Mustaqil bajarish uchun topshiriqlar

Variant tanlanib, mashgʻulotda koʻrsatilgan tartibda dastur yoziladi. Topshiriq natijasi: shartli operatorli 8 ta dastur.

Variant	Topshiriq mavzusi
1	Ikki sonning kattasini topish
2	Uch sonning eng kichigini topish
3	Yil kabisa yilmi yoki yoʻqligini aniqlash
4	Sonning 3 ga boʻlinishini tekshirish
5	Uchburchak mavjudligini tomonlari boʻyicha tekshirish
6	Hafta kuni raqami boʻyicha nomini chiqarish
7	Harorat boʻyicha kiyim tavsiyasi (3 daraja)
8	Ikkita sonni taqqoslab “katta”, “kichik” yoki “teng” deb chiqarish
9	Kvadrat tenglama diskriminanti boʻyicha ildizlar sonini aniqlash
10	Chegirma: summa 100 000 dan oshsa 10%, 500 000 dan oshsa 20%

Nazorat savollari

1. if, elif va else ning vazifasi nima?
2. Python da chekinish nima uchun muhim?
3. elif zanjirida shartlar tartibi nima uchun ahamiyatli?
4. Chekka holat nima?
5. = va == ning farqi nimada?

Adabiyotlar: [2], [4]

5-AMALIY MASHGʻULOT. MURAKKAB SHARTLAR VA ICHMA-ICH SHARTLI OPERATORLAR

Maqsad: murakkab shartlar (and, or, not) va ichma-ich shartli operatorlar bilan ishlash koʻnikmasini shakllantirish.

Kutilayotgan natijalar (TN7): tinglovchi murakkab shartli dasturlar yozadi, mantiqiy ifodalarni soddalashtiradi va test jadvali tuzadi.

Kerakli jihozlar: kompyuter, Python muhiti.

Qisqacha nazariy maʼlumot

Murakkab shart and, or, not bilan tuziladi. $0 < x < 10$ koʻrinishidagi taqqoslash zanjiri Python da mumkin. Ichma-ich shartlar murakkab mantiq uchun qoʻllanadi, lekin chuqur ichma-ichlik kodni oʻqishni qiyinlashtiradi; koʻpincha and bilan soddalashtirish mumkin. match operatori (Python 3.10 dan) koʻp variantli tanlovni ixcham yozadi.

```
x = int(input())
if 10 <= x <= 99 and x % 2 == 0:
    print("juft ikki xonali son")
```

Ishni bajarish tartibi

1-qadam. Haqiqiylik jadvali. and, or, not uchun haqiqiylik jadvali daftarda tuziladi va kod bilan tekshiriladi.

2-qadam. Murakkab shart. Sonning ikki xonali va 7 ga karrali ekanligini tekshiruvchi dastur yoziladi.

3-qadam. Ichma-ich shart. Yoshi va guvohnomasiga qarab haydashga ruxsat beruvchi dastur ikki usulda (ichma-ich va and bilan) yoziladi.

4-qadam. match operatori. Hafta kuni raqami bo'yicha ish yoki dam olish kunini aniqlovchi dastur match bilan yoziladi (muhit versiyasi tekshiriladi).

5-qadam. Test jadvali. Har bir dastur uchun kamida 6 ta test kiritish va kutilgan natija jadvali tuziladi.

Mustaqil bajarish uchun topshiriqlar

Variant tanlanib, mashg'ulotda ko'rsatilgan tartibda dastur yoziladi. Topshiriq natijasi: murakkab shartli dasturlar va test jadvali.

Variant	Topshiriq mavzusi
1	Nuqta koordinatalari bo'yicha chorakni aniqlash
2	Nuqta kvadrat ichida ekanligini tekshirish
3	Uchburchak turini (teng tomonli, teng yonli, turli tomonli) aniqlash
4	Oyning kunlari sonini (kabisa yilini hisobga olib) chiqarish
5	Yosh bo'yicha toifa: bola, o'smir, kattalar, keksa
6	Oddiy kalkulyator: ikki son va amal belgisi (+, -, *, /) bo'yicha natija; nolga bo'lishni tekshirish
7	Ikkita to'rtburchakning bir-birini qoplashini tekshirish
8	Parol sharti: uzunligi 8 dan kam emas va raqam bor
9	Vaqt (soat) bo'yicha salomlashish: ertalab, kunduzi, kechqurun
10	Tosh-qaychi-qog'oz o'yinida g'olibni aniqlash

Nazorat savollari

1. and, or, not operatorlari qanday ishlaydi?
2. $0 < x < 10$ yozuvi Python da qanday ishlaydi?
3. $x == 1$ or 2 yozuvi nima uchun noto'g'ri?
4. Ichma-ich shartni qachon and bilan almashtirish mumkin?
5. match operatori qaysi hollarda qulay?

Adabiyotlar: [2], [4]

6-AMALIY MASHG'ULOT. FOR SIKLI: YIG'INDI, KO'PAYTMA VA SANASH MASALALARI

Maqsad: for sikli va range() yordamida yig'indi, ko'paytma va sanash masalalarini yechish ko'nikmasini shakllantirish.

Kutilayotgan natijalar (TN7): tinglovchi for siklli 8 ta dastur yozadi, akkumulyator va hisoblagichdan to'g'ri foydalanadi, range() parametrlarini o'zlashtiradi.

Kerakli jihozlar: kompyuter, Python muhiti.

Qisqacha nazariy ma'lumot

for i in range(a, b, qadam) sikli sonlar ketma-ketligi ustida takrorlaydi; oxirgi qiymat kirmaydi. Akkumulyator (yig'indi uchun 0, ko'paytma uchun 1) siklgacha beriladi, hisoblagich shart bajarilganda 1 ga oshiriladi.

```
n = int(input())
yigindi = 0
juftlar = 0
for i in range(1, n + 1):
    yigindi += i
    if i % 2 == 0:
        juftlar += 1
print(yigindi, juftlar)
```

Ishni bajarish tartibi

1-qadam. range() bilan tanishuv. range(5), range(1, 6), range(10, 0, -3) natijalari ro'yxatga o'tkazib chiqariladi (list(range(...))).

2-qadam. Yig'indi. 1 dan n gacha sonlar yig'indisi va kvadratlari yig'indisi hisoblanadi.

3-qadam. Hisoblagich. Berilgan oraliqda 3 ga karrali sonlar soni sanaladi.

4-qadam. Ko'paytma. n! hisoblanadi (boshlang'ich qiymat 1).

5-qadam. Kuzatuv jadvali. n = 5 uchun yig'indi dasturining kuzatuv jadvali qo'lda to'ldirilib, kod natijasi bilan solishtiriladi.

Mustaqil bajarish uchun topshiriqlar

Variant tanlanib, mashg'ulotda ko'rsatilgan tartibda dastur yoziladi. Topshiriq natijasi: for siklli 8 ta dastur.

Variant	Topshiriq mavzusi
1	1 dan n gacha toq sonlar yig'indisi
2	n ta sonni kiritib, ularning o'rtacha qiymati
3	1 dan 100 gacha 7 ga karrali sonlar yig'indisi va soni
4	n ta sonning eng kattasi va eng kichigi

Variant	Topshiriq mavzusi
5	a dan b gacha sonlarning kublari
6	Berilgan sonning bo'luvchilari soni
7	n ta sondan musbatlari yig'indisi va soni
8	2 ning 0 dan n gacha darajalari jadvali
9	$1 + 1/2 + 1/3 + \dots + 1/n$ yig'indisi
10	Ko'paytirish jadvali (berilgan son uchun 1 dan 10 gacha)

Nazorat savollari

1. range(2, 11, 3) qanday ketma-ketlik beradi?
2. Akkumulyatorga boshlang'ich qiymat nima uchun beriladi?
3. Hisoblagich qanday ishlaydi?
4. for siklining takrorlar soni qanday aniqlanadi?
5. Kuzatuv jadvali nima beradi?

Adabiyotlar: [2], [4]

7-AMALIY MASHG'ULOT. WHILE SIKLI VA ITERATION JARAYONLAR

Maqsad: while sikli va iteration jarayonlar bilan ishlash, cheksiz sikldan qochish ko'nikmasini shakllantirish.

Kutilayotgan natijalar (TN7): tinglovchi while siklli dasturlar yozadi, to'xtash shartini to'g'ri tuzadi, kuzatuv jadvali bilan jarayonni tekshiradi.

Kerakli jihozlar: kompyuter, Python muhiti.

Qisqacha nazariy ma'lumot

while sikli shart rost bo'lguncha takrorlanadi; sikl o'zgaruvchisi tanada o'zgartirilishi shart, aks holda cheksiz sikl paydo bo'ladi. break siklni to'xtatadi, continue joriy iteratsiyani o'tkazib yuboradi. Raqamlar bilan ishlashda $n \neq 10$ va $n \% 10$ odatiy juftlik.

```
n = int(input())
s = 0
while n > 0:
    s += n % 10
    n //= 10
print(s)
```

Ishni bajarish tartibi

1-qadam. Raqamlar yig'indisi. Sonning raqamlari yig'indisi while bilan hisoblanadi; $n = 4075$ uchun kuzatuv jadvali to'ldiriladi.

2-qadam. Foydalanuvchi kiritishini tekshirish. To'g'ri qiymat (musbat son) kiritilguncha so'raydigan sikl yoziladi (while True va break).

3-qadam. Yig'ish sikli. 0 kiritilguncha kiritilgan sonlar yig'indisini hisoblovchi dastur yoziladi.

4-qadam. Cheksiz sikl. Qasddan sikl o'zgaruvchisi o'zgartirilmaydi, Thonny da dastur to'xtatiladi (Ctrl+C), sababi tushuntiriladi.

5-qadam. Kollatz gipotezasi. Sonni 1 ga yetguncha o'zgartiruvchi (juft bo'lsa 2 ga bo'lish, toq bo'lsa $3n + 1$) jarayon qadamlari soni sanaladi.

Mustaqil bajarish uchun topshiriqlar

Variant tanlanib, mashg'ulotda ko'rsatilgan tartibda dastur yoziladi. Topshiriq natijasi: while siklli dasturlar va kuzatuv jadvali.

Variant	Topshiriq mavzusi
1	Sonning raqamlari soni
2	Sonning teskarisi (raqamlari teskari tartibda)
3	Sonning eng katta raqami
4	0 kiritilguncha sonlarning o'rtachasi
5	Ikkilik sanoq sistemasiga o'tkazish (bo'lish usuli)
6	n dan kichik bo'lmagan eng kichik 2 ning darajasi
7	Sonning raqamlari orasida juftlar soni
8	EKUB ni Evklid algoritmi bilan topish
9	Omonat foiz bilan necha yilda ikki barobar bo'lishini hisoblash
10	Sonlar ketma-ketligi qat'iy o'sib borishini tekshirish (0 gacha kiritiladi)

Nazorat savollari

1. while va for sikllari qachon tanlanadi?
2. Cheksiz sikl qachon yuzaga keladi va qanday to'xtatiladi?
3. break va continue ning farqi nima?
4. $n \neq 10$ va $n \% 10$ nima uchun birga ishlatiladi?
5. Evklid algoritmi qanday ishlaydi?

Adabiyotlar: [2], [4]

8-AMALIY MASHG'ULOT. ICHMA-ICH SIKLLAR VA NAQSHLAR CHIZISHGA DOIR MASALALAR

Maqsad: ichma-ich sikllar yordamida naqshlar va jadvallar chizish ko'nikmasini shakllantirish.

Kutilayotgan natijalar (TN7): tinglovchi naqsh chizuvchi dasturlar to'plamini yozadi, tashqi va ichki sikl o'zgaruvchilarini to'g'ri bog'laydi.

Kerakli jihozlar: kompyuter, Python muhiti.

Qisqacha nazariy ma'lumot

Ichma-ich siklda tashqi sikl har bir qadamda ichki siklni to'liq bajaradi. Naqshlarda tashqi sikl qatorlarni, ichki sikl qatordagi belgilarni boshqaradi. `print(..., end="")` qatorni tugatmasdan chiqaradi; ichki sikldan keyin bo'sh `print()` yangi qatorga o'tadi. Umumiy iteratsiyalar soni takrorlar ko'paytmasiga teng.

```
n = 5
for i in range(1, n + 1):
    print(" " * (n - i) + "*" * (2 * i - 1))
```

Ishni bajarish tartibi

1-qadam. To'rtburchak. n qator va m ustundan iborat yulduzchalar to'rtburchagi ichma-ich sikl bilan chiziladi.

2-qadam. Uchburchak. Yulduzchalar to'g'ri burchakli uchburchagi chiziladi (1, 2, 3, ... yulduzcha).

3-qadam. Piramida. Yuqoridagi namuna ishga tushirilib, bo'shliqlar va yulduzchalar formulasi tahlil qilinadi.

4-qadam. Ko'paytirish jadvali. 1 dan 9 gacha jadval formatlangan (`f"{i * j:3d}"`) holda chiqariladi.

5-qadam. Iteratsiyalar soni. Ichki siklda hisoblagich qo'yib, umumiy takrorlar soni n va m bo'yicha tekshiriladi.

Mustaqil bajarish uchun topshiriqlar

Variant tanlanib, mashg'ulotda ko'rsatilgan tartibda dastur yoziladi. Topshiriq natijasi: naqsh chizuvchi dasturlar to'plami.

Variant	Topshiriq mavzusi
1	Raqamlar uchburchagi (1; 1 2; 1 2 3; ...)
2	Teskari uchburchak (yulduzchalar kamayib boradi)
3	Romb naqshi
4	Shaxmat taxtasi (# va . almashinuvi)
5	Ramka (ichi bo'sh to'rtburchak)
6	Ko'paytirish jadvali to'liq 10x10
7	Raqamli piramida (1; 121; 12321)
8	"X" shaklidagi naqsh
9	Pascal uchburchagi (5 qator)
10	Ikki sondan iborat barcha juftliklar ($i < j$) va ularning soni

Nazorat savollari

1. Ichma-ich sikl qanday bajariladi?
2. `print(end="")` nima uchun ishlatiladi?

3. Piramidada bo'shliqlar soni qanday topiladi?
4. Ichki sikldan keyingi bo'sh print() nima qiladi?
5. Ichma-ich sikl umumiy takrorlar soni qanday aniqlanadi?

Adabiyotlar: [2], [4]

9-AMALIY MASHG'ULOT. SONLAR NAZARIYASIGA DOIR MASALALAR: TUB SONLAR, EKUB, EKUK, RAQAMLAR YIG'INDISI

Maqsad: sonlar nazariyasiga oid masalalarni (tub sonlar, EKUB, EKUK, raqamlar yig'indisi) dasturlash ko'nikmasini shakllantirish.

Kutilayotgan natijalar (TN9): tinglovchi tub sonni tekshiradi, EKUB va EKUK ni hisoblaydi, sodda sonlar nazariyasi masalalarini effektiv yechadi.

Kerakli jihozlar: kompyuter, Python muhiti.

Qisqacha nazariy ma'lumot

Tub son faqat 1 ga va o'ziga bo'linadigan 1 dan katta natural son. Tubligini tekshirishda bo'luvchilarni $\sqrt{n}$ gacha qidirish yetarli. EKUB ni Evklid algoritmi topadi: $EKUB(a, b) = EKUB(b, a \% b)$. $EKUK = a * b // EKUB(a, b)$. Python da `math.gcd()` ham mavjud.

```
def tub(n):
    if n < 2:
        return False
    i = 2
    while i * i <= n:
        if n % i == 0:
            return False
        i += 1
    return True
```

Ishni bajarish tartibi

1-qadam. Tub sonni tekshirish. Yuqoridagi funksiya yoziladi, 2 dan 50 gacha tub sonlar chiqariladi.

2-qadam. EKUB va EKUK. Evklid algoritmi bilan EKUB topiladi, EKUK formula orqali hisoblanadi.

```
def ekub(a, b):
    while b:
        a, b = b, a % b
    return a
```

3-qadam. Samaradorlikni solishtirish. Bo'luvchilarni n gacha va $\sqrt{n}$ gacha tekshirish usullari katta son uchun vaqt bo'yicha taqqoslanadi.

4-qadam. Eratosfen g'alviri. n gacha barcha tub sonlar ro'yxati g'alvir usulida topiladi va natija oldingi usul bilan solishtiriladi.

5-qadam. Juftlikda tekshirish. Juft bir-birining kodini sinab, chekka holatlar (1, 2, juda katta son) kiritadi.

Mustaqil bajarish uchun topshiriqlar

Variant tanlanib, mashg'ulotda ko'rsatilgan tartibda dastur yoziladi. Topshiriq natijasi: sonlar nazariyasi bo'yicha dasturlar.

Variant	Topshiriq mavzusi
1	n gacha barcha tub sonlar
2	Sonning tub ko'paytuvchilarga yoyilmasi
3	Ikki sonning EKUB va EKUK
4	Mukammal sonlarni (bo'luvchilari yig'indisi o'ziga teng) n gacha topish
5	Egizak tub sonlar juftliklari (farqi 2) n gacha
6	Armstrong sonlarini (raqamlari kublari yig'indisi o'ziga teng) topish
7	Sonning raqamli ildizi (bir xonali bo'lguncha raqamlar yig'indisi)
8	n gacha sonlar orasida o'zaro tub juftliklar soni
9	Berilgan sonning bo'luvchilari ro'yxati
10	Goldbax gipotezasini tekshirish: juft sonni ikki tub son yig'indisi sifatida yozish

Nazorat savollari

1. Tub son nima va uni qanday tekshirish mumkin?
2. Nima uchun bo'luvchini $\sqrt{n}$ gacha qidirish yetarli?
3. Evklid algoritmi qanday ishlaydi?
4. EKUK ni EKUB orqali qanday topamiz?
5. Eratosfen g'alviri g'oyasi nimada?

Adabiyotlar: [4], [6]

10-AMALIY MASHG'ULOT. SATRLAR USTIDA AMALLAR VA SATR METODLARI

Maqsad: satrlar va satr metodlari bilan ishlash, indekslash va kesimlarni qo'llash ko'nikmasini shakllantirish.

Kutilayotgan natijalar (TN7): tinglovchi satr metodlari bo'yicha dasturlar yozadi, kesimlar va f-string dan to'g'ri foydalanadi.

Kerakli jihozlar: kompyuter, Python muhiti.

Qisqacha nazariy ma'lumot

Satr o'zgarmas belgilar ketma-ketligi; indeks 0 dan boshlanadi, manfiy indeks oxiridan sanaydi. Kesim `s[a:b:q]` da `b` kirmaydi. Metodlar yangi satr qaytaradi: `upper()`, `lower()`, `strip()`, `replace()`, `split()`, `join()`, `find()`, `count()`, `startswith()`.

```
s = "  Informatika  "
t = s.strip().lower()
print(t[:4], t[::-1], t.count("a"), t.find("m"))
```

Ishni bajarish tartibi

1-qadam. Indeks va kesim. Berilgan satrdan turli kesimlar olinib, natijalar daftarga yoziladi.

2-qadam. Metodlar jadvali. upper, lower, strip, replace, split, join metodlari bittadan misol bilan sinab ko‘riladi.

3-qadam. Kiritilgan matnni tozalash. input().strip().lower() zanjiri bilan kiritilgan ismni standart ko‘rinishga keltiruvchi dastur yoziladi.

4-qadam. split va join. Gapni so‘zlarga ajratib, so‘zlar tartibini teskari qilib qayta birlashtiriladi.

```
gap = input()
print(" ".join(gap.split()[::-1]))
```

5-qadam. Satr o‘zgarmasligi. s[0] = “X” xatosi ko‘rib, to‘g‘ri usul (kesimlar bilan yangi satr yaratish) ko‘rsatiladi.

Mustaqil bajarish uchun topshiriqlar

Variant tanlanib, mashg‘ulotda ko‘rsatilgan tartibda dastur yoziladi. Topshiriq natijasi: satr metodlari bo‘yicha dasturlar.

Variant	Topshiriq mavzusi
1	Ismning birinchi harfini katta, qolganini kichik qilish
2	Satrdagi so‘zlar soni
3	Satrdagi ma’lum harf necha marta uchrashini sanash
4	Satrnı teskari yozish
5	Satrdagi barcha bo‘shliqlarnı “_” bilan almashtirish
6	Fayl nomidan kengaytmasini ajratib olish
7	Ikki so‘zdan eng uzunini topish
8	Email manzilda “@” va nuqta borligini tekshirish
9	Satrning har ikkinchi belgisini chiqarish
10	Ism va familiyadan bosh harflarnı (inisiاللarnı) hosil qilish

Nazorat savollari

1. Satr nima va nima uchun o‘zgarmas deyiladi?
2. Kesimda oxirgi chegara kiradimi?
3. split() va join() nima qiladi?
4. find() topilmasa nima qaytaradi?
5. strip() qaysi hollarda foydali?

Adabiyotlar: [1], [2]

11-AMALIY MASHG‘ULOT. SATRLARNI QAYTA ISHLASHGA DOIR MASALALAR: PALINDROM, SO‘ZLARNI SANASH

Maqsad: satrlarni qayta ishlashga oid masalalar (palindrom, so‘zlarni sanash, shifrlash) yechish ko‘nikmasini shakllantirish.

Kutilayotgan natijalar (TN7): tinglovchi matn tahlili va shifrlash dasturlarini yozadi, harf kodlari bilan ishlaydi.

Kerakli jihozlar: kompyuter, Python muhiti.

Qisqacha nazariy ma’lumot

Matnni qayta ishlashda sikl satr belgilari bo‘yicha yuradi. `ord(h)` belgi kodini, `chr(k)` kod bo‘yicha belgini beradi. Sezar shifrida har bir harf alifbo bo‘yicha belgilangan qadamga siljiriladi. Palindrom – ikki tomondan bir xil o‘qiladigan satr; probel va katta-kichik harf farqi inobatga olinmaydi.

```
def sezar(matn, k):  
    natija = ""  
    for h in matn:  
        if "a" <= h <= "z":  
            natija += chr((ord(h) - ord("a") + k) % 26 + ord("a"))  
        else:  
            natija += h  
    return natija
```

Ishni bajarish tartibi

1-qadam. Palindrom. Matn palindrom ekanini tekshiruvchi funksiya yoziladi (kesim `[::-1]` yordamida).

2-qadam. So‘zlar va harflar. Matndagi so‘zlar soni, unli harflar soni va eng uzun so‘z topiladi.

3-qadam. Sezar shifri. Namuna funksiya ishga tushiriladi; shifrlash va deshifrlash (`k` va `-k`) tekshiriladi.

4-qadam. Ikki juftlikda. Juftlikdagi bir kishi matnni shifrlaydi, ikkinchisi deshifrlaydi; natijalar solishtiriladi.

5-qadam. O‘zbek harflari. o‘ va g‘ harflari turli apostrof bilan yozilganda dastur qanday ishlashi sinab ko‘riladi, muammo muhokama qilinadi.

Mustaqil bajarish uchun topshiriqlar

Variant tanlanib, mashg‘ulotda ko‘rsatilgan tartibda dastur yoziladi. Topshiriq natijasi: satr masalalari yechimlari.

Variant	Topshiriq mavzusi
1	Matndagi unli va undosh harflar soni
2	So‘zlarni uzunligi bo‘yicha tartiblash

Variant	Topshiriq mavzusi
3	Matndagi eng ko‘p uchraydigan harf
4	Matnda takrorlanadigan so‘zlarni topish
5	Har bir so‘zning birinchi harfini katta qilish
6	Matndan raqamlarni ajratib, yig‘indisini topish
7	Matndan faqat harflarni qoldirish
8	Teskari shifr (alifbo oxiridan sanash: a -> z)
9	Gapdagi so‘zlar tartibini teskari qilish
10	Anagrammalarni tekshirish (ikki so‘z bir xil harflardan iboratmi)

Nazorat savollari

1. Palindromni tekshirish algoritmi qanday?
2. ord() va chr() nima vazifani bajaradi?
3. Sezar shifri qanday ishlaydi?
4. Nima uchun % 26 ishlatiladi?
5. Matnni qayta ishlashda qanday tipik xatolar uchraydi?

Adabiyotlar: [2], [4]

12-AMALIY MASHG‘ULOT. RO‘YXATLAR BILAN ISHLASH: YARATISH, QO‘SHISH, O‘CHIRISH, KESIMLAR

Maqsad: ro‘yxatlar bilan ishlash: yaratish, qo‘shish, o‘chirish, kesim va nusxalashni o‘zlashtirish.

Kutilayotgan natijalar (TN7): tinglovchi ro‘yxat metodlari va kesimlarni qo‘llaydi, nusxalash xatosini tushunadi va to‘g‘ri nusxa yaratadi.

Kerakli jihozlar: kompyuter, Python muhiti, ixtiyoriy Python Tutor (onlayn vizualizator).

Qisqacha nazariy ma’lumot

Ro‘yxat o‘zgaruvchan ketma-ketlik: append, insert, remove, pop, sort, reverse, extend metodlari bor. b = a nusxa emas, bitta ob’yektga ikkinchi nom; nusxa uchun a.copy() yoki a[:]. sort() joyida o‘zgartiradi va None qaytaradi, sorted() yangi ro‘yxat beradi.

```
a = [5, 2, 9, 1]
b = a
b.append(7)
print(a)           # [5, 2, 9, 1, 7]
c = a.copy()
c.append(0)
print(a, c)
```

Ishni bajarish tartibi

1-qadam. Ro'yxat yaratish. Klaviaturadan n ta son o'qib ro'yxat tuziladi (`list(map(int, input().split()))`).

2-qadam. Metodlar. `append`, `insert`, `remove`, `pop`, `sort`, `reverse` metodlari ketma-ket sinab ko'riladi; har bir amaldan keyin ro'yxat chiqariladi.

3-qadam. Kesimlar. `a[1:4]`, `a[::-1]`, `a[::2]` kesimlari natijalari taxmin qilinib, tekshiriladi.

4-qadam. Nusxalash tuzog'i. Yuqoridagi namuna ishga tushiriladi; Python Tutor da xotira sxemasi ko'riladi va farq tushuntiriladi.

5-qadam. Ro'yxatni qayta ishlash. Ro'yxatdan juft sonlarni ajratib yangi ro'yxat hosil qilinadi.

Mustaqil bajarish uchun topshiriqlar

Variant tanlanib, mashg'ulotda ko'rsatilgan tartibda dastur yoziladi. Topshiriq natijasi: ro'yxat amallari bo'yicha dasturlar.

Variant	Topshiriq mavzusi
1	Ro'yxatdagi sonlar yig'indisi va o'rtachasi
2	Ro'yxatdagi eng katta va eng kichik element indekslari
3	Ro'yxatni teskari qilish (<code>reverse</code> ishlatmasdan)
4	Ro'yxatdagi takrorlanuvchi elementlarni olib tashlash (tartibni saqlab)
5	Ro'yxatning birinchi va oxirgi elementini almashtirish
6	Ikki ro'yxatni birlashtirib tartiblash
7	Ro'yxatni ikki teng qismga ajratish
8	Ro'yxatdagi musbat va manfiy sonlarni alohida ro'yxatlarga ajratish
9	Ro'yxatni k pozitsiyaga siklik siljitish
10	Ro'yxatdagi har bir elementning o'ng qo'shnisi bilan yig'indisi

Nazorat savollari

1. `append()` va `insert()` ning farqi nima?
2. `b = a` va `b = a.copy()` orasidagi farq nima?
3. `sort()` va `sorted()` ni taqqoslang.
4. Kesimda manfiy indekslar qanday ishlaydi?
5. `pop()` va `remove()` qanday farq qiladi?

Adabiyotlar: [1], [4]

13-AMALIY MASHG'ULOT. RO'YXATLARDA QIDIRISH, ENG KATTA VA ENG KICHIK ELEMENTNI TOPISH, SARALASH

Maqsad: ro'yxatlarda qidirish, eng katta va eng kichik elementni topish hamda saralash algoritmlarini dasturlash ko'nikmasini shakllantirish.

Kutilayotgan natijalar (TN7): tinglovchi chiziqli qidiruv, ekstremumlarni topish va tayyor sort/sorted dan foydalanishni o'zlashtiradi.

Kerakli jihozlar: kompyuter, Python muhiti.

Qisqacha nazariy ma'lumot

Chiziqli qidiruv ro'yxatni boshidan oxirigacha ko'radi: $O(n)$. Eng katta elementni topishda birinchi element boshlang'ich qiymat qilib olinadi. `sort(key=..., reverse=...)` murakkab tartib uchun kalit funksiyasini qabul qiladi; lambda kalit sifatida qulay.

```
a = [4, 9, 2, 9, 1]
eng_katta = a[0]
indeks = 0
for i in range(1, len(a)):
    if a[i] > eng_katta:
        eng_katta, indeks = a[i], i
print(eng_katta, indeks)
```

Ishni bajarish tartibi

1-qadam. Chiziqli qidiruv. Ro'yxatda elementni qidirib, topilgan birinchi indeksni (topilmasa -1) qaytaruvchi funksiya yoziladi.

2-qadam. Ekstremumlar. `max()` va `min()` ishlatmasdan eng katta va eng kichik element topiladi, natija tayyor funksiyalar bilan solishtiriladi.

3-qadam. Saralash. `sort()` va `sorted()` bilan o'sish va kamayish bo'yicha saralanadi.

4-qadam. Kalit bo'yicha saralash. (ism, ball) juftliklari ro'yxati ball bo'yicha kamayish tartibida lambda yordamida saralanadi.

```
talaba = [("Ali", 85), ("Vali", 92), ("Gul", 78)]
talaba.sort(key=lambda t: t[1], reverse=True)
```

5-qadam. Tahlil. Qidiruv algoritmining eng yomon holati (element yo'q) uchun taqqoslashlar soni sanaladi.

Mustaqil bajarish uchun topshiriqlar

Variant tanlanib, mashg'ulotda ko'rsatilgan tartibda dastur yoziladi. Topshiriq natijasi: qidirish va saralash dasturlari.

Variant	Topshiriq mavzusi
1	Elementning barcha uchrashgan indekslarini topish
2	Ikkinchi eng katta elementni topish
3	Ro'yxatda eng ko'p uchraydigan elementni topish
4	Ikki element orasidagi minimal farqni topish
5	Ro'yxat tartiblangan yoki yo'qligini tekshirish
6	Satrlar ro'yxatini uzunligi bo'yicha saralash
7	Talabalarni ball bo'yicha, ball teng bo'lsa ism bo'yicha saralash
8	Musbat elementlarning eng kichigini topish
9	Ro'yxat medianasini topish
10	Eng uzun o'suvchi ketma-ket qism (qo'shni elementlar) uzunligi

Nazorat savollari

1. Chiziqli qidiruvning murakkabligi qanday?
2. Eng katta elementni topish algoritmini tushuntiring.
3. sort(key=...) qanday ishlaydi?
4. lambda funksiya nima?
5. Ro'yxat tartiblangan bo'lsa, qidiruvni qanday tezlashtirish mumkin?

Adabiyotlar: [4], [6]

14-AMALIY MASHG'ULOT. LIST COMPREHENSION VA IKKI O'LCHOVLI RO'YXATLAR (MATRITSALAR)

Maqsad: list comprehension va ikki o'lchovli ro'yxatlar (matritsalar) bilan ishlash ko'nikmasini shakllantirish.

Kutilayotgan natijalar (TN7): tinglovchi comprehension bilan ro'yxat yaratadi, matritsalarini to'g'ri yaratadi va ular ustida amallar bajaradi.

Kerakli jihozlar: kompyuter, Python muhiti.

Qisqacha nazariy ma'lumot

List comprehension [ifoda for x in ketma-ketlik if shart] ro'yxatni ixcham yaratadi. Matritsa – ro'yxatlar ro'yxati; m[i][j] i-qator j-ustun elementi. To'g'ri yaratish: [[0] * m for _ in range(n)]; [[0] * m] * n xato, chunki qatorlar bitta ob'yekt bo'ladi.

```
n, m = 3, 4
mat = [[i * j for j in range(m)] for i in range(n)]
for qator in mat:
    print(*qator)
yig = sum(sum(q) for q in mat)
```

Ishni bajarish tartibi

1-qadam. Comprehension. Kvadratlar, juft sonlar, soʻzlar uzunliklari roʻyxatlari bitta qatorda yaratiladi; oddiy sikl shakli bilan solishtiriladi.

2-qadam. Matritsa yaratish. $n \times m$ matritsa klaviaturadan oʻqiladi va chiqariladi.

3-qadam. Amallar. Matritsa elementlari yigʻindisi, har bir qator va ustun yigʻindisi topiladi.

4-qadam. Xato namunasi. $[[0] * 3] * 3$ bilan matritsa yaratilib, bitta elementni oʻzgartirganda butun ustun oʻzgargani koʻrsatiladi; sababi tushuntiriladi.

5-qadam. Transponirlash. Matritsa transponirlanadi: $[[a[i][j]] \text{ for } i \text{ in range}(n)] \text{ for } j \text{ in range}(m)]$.

Mustaqil bajarish uchun topshiriqlar

Variant tanlanib, mashgʻulotda koʻrsatilgan tartibda dastur yoziladi. Topshiriq natijasi: matritsa va comprehension dasturlari.

Variant	Topshiriq mavzusi
1	Bosh diagonal elementlari yigʻindisi
2	Matritsani songa koʻpaytirish
3	Ikki matritsani qoʻshish
4	Matritsada eng katta element va uning oʻrni
5	Matritsaning har bir qatorining oʻrtachasi
6	Spiral shaklida yurish tartibini chiqarish
7	Matritsani 90 gradusga burish
8	Pascal uchburchagini ikki oʻlchovli roʻyxatda hosil qilish
9	“Zig-zag” tartibida chiqarish (juft qatorlar teskari)
10	Ikki matritsani koʻpaytirish

Nazorat savollari

1. List comprehension nima?
2. Matritsa Python da qanday ifodalanadi?
3. $[[0] * n] * n$ nima uchun notoʻgʻri?
4. Transponirlash nima?
5. Matritsa diagonal elementlariga qanday murojaat qilinadi?

Adabiyotlar: [1], [4]

15-AMALIY MASHGʻULOT. KORTEJLAR VA TOʻPLAMLARGA DOIR MASALALAR

Maqsad: kortejlar va toʻplamlar bilan ishlash koʻnikmasini shakllantirish.

Kutilayotgan natijalar (TN7): tinglovchi kortej va to‘plam amallarini qo‘llaydi, takrorlarni olib tashlaydi, to‘plamlar bilan masalalar yechadi.

Kerakli jihozlar: kompyuter, Python muhiti.

Qisqacha nazariy ma’lumot

Kortej o‘zgarmas ketma-ketlik: koordinata, sana kabi qat’iy guruhlar va funksiyadan bir nechta qiymat qaytarish uchun. To‘plam takrorlanmaydigan elementlar majmuasi: birlashma (|), kesishma (&), ayirma (-), simmetrik ayirma (^). Bo‘sh to‘plam set() bilan yaratiladi, {} esa bo‘sh lug‘at.

```
a = {1, 2, 3, 4}
b = {3, 4, 5}
print(a | b, a & b, a - b, a ^ b)
nuqta = (3, 4)
x, y = nuqta
```

Ishni bajarish tartibi

1-qadam. Kortej. Nuqtalar koordinatalari kortej sifatida saqlanadi, ular orasidagi masofa hisoblanadi.

2-qadam. Unpacking. Funksiya min va max ni kortej sifatida qaytaradi, natija ikkita o‘zgaruvchiga ochiladi.

3-qadam. To‘plam yaratish. Ro‘yxatdan takrorlar olib tashlanadi: set(royxat).

4-qadam. To‘plam amallari. Ikki guruh o‘quvchilari uchun ikkala to‘garakka qatnashuvchilar (kesishma) va faqat birida qatnashuvchilar (ayirma) aniqlanadi.

5-qadam. Tezlikni solishtirish. Katta ro‘yxat va to‘plamda x in tekshiruvi vaqti o‘lchab solishtiriladi.

Mustaqil bajarish uchun topshiriqlar

Variant tanlanib, mashg‘ulotda ko‘rsatilgan tartibda dastur yoziladi. Topshiriq natijasi: kortej va to‘plam masalalari.

Variant	Topshiriq mavzusi
1	Matndagi noyob so‘zlar soni
2	Ikki ro‘yxatning umumiy elementlari
3	Birinchi ro‘yxatda bor, ikkinchisida yo‘q elementlar
4	Uch to‘garak ishtirokchilari bo‘yicha amallar
5	Matndagi noyob harflar to‘plami
6	Kortejlar ro‘yxatini ikkinchi elementi bo‘yicha saralash
7	Ikki nuqta orasidagi masofa (kortej koordinatalari)
8	Sonlar ro‘yxatida takrorlanmaydigan elementlar soni
9	Pangramma tekshiruvi (matnda alifbo barcha harflari bormi)

Variant	Topshiriq mavzusi
10	Funksiya yig'indi, ko'paytma va o'rtachani kortej qilib qaytaradi

Nazorat savollari

1. Kortej ro'yxatdan nimasi bilan farq qiladi?
2. To'plam qanday xususiyatlarga ega?
3. {} bilan bo'sh to'plam yaratish mumkinmi?
4. Kesishma va ayirma amallarini tushuntiring.
5. To'plamda tekshiruv nima uchun tez ishlaydi?

Adabiyotlar: [1], [4]

16-AMALIY MASHG'ULOT. LUG'ATLAR BILAN ISHLASH: CHASTOTA LUG'ATI, TELEFON KITOB

Maqsad: lug'atlar bilan ishlash, chastota lug'ati va telefon kitobi kabi masalalarni yechish ko'nikmasini shakllantirish.

Kutilayotgan natijalar (TN7): tinglovchi lug'at yaratadi, qiymatlarni qo'shadi, o'zgartiradi, o'chiradi, chastota hisoblaydi va lug'at bo'yicha yuradi.

Kerakli jihozlar: kompyuter, Python muhiti.

Qisqacha nazariy ma'lumot

Lug'at kalit : qiymat juftliklari to'plami; kalitlar noyob va o'zgarmas bo'ladi. get(k, std) KeyError dan saqlaydi; items() juftliklar bo'yicha yurishni beradi. Chastota lug'ati: $d[x] = d.get(x, 0) + 1$. Python 3.7 dan lug'at kiritilish tartibini saqlaydi.

```
matn = input().lower().split()
chastota = {}
for s in matn:
    chastota[s] = chastota.get(s, 0) + 1
for s, k in sorted(chastota.items(), key=lambda p: p[1], reverse=True):
    print(s, k)
```

Ishni bajarish tartibi

1-qadam. Lug'at yaratish va o'zgartirish. Talaba haqidagi lug'at yaratilib, kalitlar qo'shiladi, o'zgartiriladi va o'chiriladi.

2-qadam. Xavfsiz murojaat. $d["yo'q"]$ va $d.get("yo'q", "topilmadi")$ farqi ko'riladi.

3-qadam. Chastota lug'ati. Namuna ishga tushirilib, matndagi so'zlar chastotasi hisoblanadi.

4-qadam. Telefon kitobi. Ism-telefon lug'ati yaratilib, qidirish, qo'shish va o'chirish menyusi tuziladi (while sikli bilan).

5-qadam. Eng ko‘p uchraydigan. max(chastota, key=chastota.get) bilan eng ko‘p uchragan so‘z topiladi.

Mustaqil bajarish uchun topshiriqlar

Variant tanlanib, mashg‘ulotda ko‘rsatilgan tartibda dastur yoziladi. Topshiriq natijasi: lug‘atli dasturlar.

Variant	Topshiriq mavzusi
1	Harflar chastotasini hisoblash
2	Telefon kitobi: qo‘shish, qidirish, o‘chirish
3	Sinfdagi o‘quvchilar va baholar lug‘ati: o‘rtacha ball
4	Inglizcha-o‘zbekcha mini lug‘at
5	Do‘kon ombori: mahsulot va miqdor, sotuvdan keyin yangilash
6	Ovoz berish natijalari: nomzod bo‘yicha ovozlar soni va g‘olib
7	Ikki lug‘atni birlashtirish (qiymatlarni qo‘shib)
8	Lug‘atni qiymatlari bo‘yicha saralab chiqarish
9	Satrdagi so‘zlarni uzunligi bo‘yicha guruhlash
10	Kalit va qiymatlarni almashtirish (teskari lug‘at)

Nazorat savollari

1. Lug‘at kaliti qanday talablarga javob berishi kerak?
2. d[k] va d.get(k) farqi nima?
3. Chastota lug‘ati qanday tuziladi?
4. items() nima qaytaradi?
5. Lug‘at va ro‘yxatni qachon qaysi birini tanlaysiz?

Adabiyotlar: [1], [4]

17-AMALIY MASHG‘ULOT. FUNKSIYALAR YARATISH VA CHAQIRISH

Maqsad: funksiyalar yaratish va chaqirish, parametrlar va qaytariladigan qiymatni to‘g‘ri ishlatish ko‘nikmasini shakllantirish.

Kutilayotgan natijalar (TN7): tinglovchi masalalarni funksiyalarga bo‘ladi, docstring yozadi, lokal va global o‘zgaruvchilarni farqlaydi.

Kerakli jihozlar: kompyuter, Python muhiti.

Qisqacha nazariy ma’lumot

Funksiya def bilan yaratiladi; return qiymat qaytaradi, return bo‘lmasa None. Standart qiymatli va nomli parametrlar qulaylik beradi. Funksiya ichidagi o‘zgaruvchi lokal. O‘zgaruvchan ob‘yekt (ro‘yxat) ni standart qiymat qilish xavfli. Katta masala kichik funksiyalarga bo‘linadi (dekompozitsiya).

```
def doira_yuzi(r, pi=3.14159):
    """Radiusi r bo'lgan doira yuzini qaytaradi."""
    return pi * r ** 2
print(doira_yuzi(2), doira_yuzi(r=3))
```

Ishni bajarish tartibi

1-qadam. Oddiy funksiyalar. Ikki sonning kattasi, sonning juft-toqligi, sonning kvadrati funksiyalari yoziladi.

2-qadam. print va return. print qiladigan va return qiladigan funksiya variantlari taqqoslanadi; natijani boshqa ifodada ishlatish sinab ko‘riladi.

3-qadam. Parametrlar. Standart qiymatli va nomli argumentli funksiya yoziladi.

4-qadam. Lokal va global. Bir xil nomli lokal va global o‘zgaruvchi misolida ko‘rinish sohasi tushuntiriladi.

5-qadam. Dekompozitsiya. Katta masala (n ta son statistikasi) kiritish, hisoblash va chiqarish funksiyalariga bo‘linadi.

Mustaqil bajarish uchun topshiriqlar

Variant tanlanib, mashg‘ulotda ko‘rsatilgan tartibda dastur yoziladi. Topshiriq natijasi: funksiyalar to‘plami.

Variant	Topshiriq mavzusi
1	Uch sonning eng kattasini qaytaruvchi funksiya
2	Sonning faktoriali (siklli)
3	Sonning tubligini tekshiruvchi funksiya
4	Ro‘yxatning o‘rtachasini qaytaruvchi funksiya
5	Satr palindrom ekanini tekshiruvchi funksiya
6	Sonning raqamlari yig‘indisini qaytaruvchi funksiya
7	Ro‘yxatdan juft sonlarni ajratuvchi funksiya
8	Ikki nuqta orasidagi masofa funksiyasi
9	Funksiya min, max va o‘rtachani bitta qiymatda (kortej) qaytaradi
10	Yilning kabisaligi va oyning kunlari soni funksiyalari

Nazorat savollari

1. Funksiya nima uchun kerak?
2. Parametr va argumentning farqi nima?
3. print va return farqini tushuntiring.
4. Lokal va global o‘zgaruvchi nima?
5. Docstring nima uchun yoziladi?

Adabiyotlar: [2], [4]

18-AMALIY MASHG‘ULOT. REKURSIV FUNKSIYALAR: FAKTORIAL, FIBONACHCHI SONLARI, XANOY MINORASI

Maqsad: rekursiv funksiyalarni (faktorial, Fibonachchi, Xanoy minorasi) yozish va ularning ishlashini tahlil qilish ko‘nikmasini shakllantirish.

Kutilayotgan natijalar (TN7): tinglovchi bazaviy holat va rekursiv qadamni aniqlaydi, chaqiruvlar jarayonini tahlil qiladi, rekursiv va siklli yechimlarni solishtiradi.

Kerakli jihozlar: kompyuter, Python muhiti, ixtiyoriy Python Tutor.

Qisqacha nazariy ma’lumot

Rekursiv funksiyada bazaviy holat (to‘xtash sharti) va rekursiv qadam bo‘ladi. Chaqiruvlar stekda saqlanadi; bazaviy holatga yetib bo‘lmasa RecursionError yuzaga keladi. Fibonachchining sodda rekursiv usuli bir xil qiymatlarni qayta-qayta hisoblaydi va sekin; siklli usul tezroq. Xanoy minorasi n ta diskni $2^n - 1$ yurishda ko‘chiradi.

```
def hanoy(n, a, b, c):  
    if n == 0:  
        return  
    hanoy(n - 1, a, c, b)  
    print(f"{n}-diskni {a} dan {c} ga")  
    hanoy(n - 1, b, a, c)  
hanoy(3, "A", "B", "C")
```

Ishni bajarish tartibi

1-qadam. Faktorial. Rekursiv faktorial yoziladi; faktorial(4) chaqiruvlar stekini daftarga chizing.

2-qadam. Bazaviy holatsiz sinash. Bazaviy holat olib tashlanib, RecursionError kuzatiladi va sababi yoziladi.

3-qadam. Fibonachchi. Rekursiv va siklli yechimlar yoziladi; $n = 30$ va $n = 35$ uchun bajarilish vaqti o‘lchanadi.

4-qadam. Xanoy minorasi. Yuqoridagi namuna 3 va 4 disk uchun ishga tushirilib, yurishlar soni $2^n - 1$ ekanligi tekshiriladi.

5-qadam. Rekursiv masalalar. Sonlar yig‘indisi, raqamlar yig‘indisi, daraja hisoblashni rekursiv yozish.

Mustaqil bajarish uchun topshiriqlar

Variant tanlanib, mashg‘ulotda ko‘rsatilgan tartibda dastur yoziladi. Topshiriq natijasi: rekursiv funksiyalar va ularning tahlili.

Variant	Topshiriq mavzusi
1	Raqamlar yig'indisini rekursiv topish
2	a ning n-darajasini rekursiv hisoblash
3	Satni rekursiv teskari yozish
4	EKUB ni Evklid algoritmi bilan rekursiv topish
5	Sonning ikkilik yozuvini rekursiv hosil qilish
6	Ro'yxat elementlari yig'indisini rekursiv topish
7	Palindromni rekursiv tekshirish
8	Ikkilik qidiruvni rekursiv yozish
9	Fibonachchini xotira (kesh) bilan tezlashtirish (lug'at yordamida)
10	Satr belgilarining barcha o'rin almashtirishlarini chiqarish

Nazorat savollari

1. Rekursiyada bazaviy holat nima uchun kerak?
2. Chaqiruvlar steki nima?
3. Rekursiv Fibonachchi nima uchun sekin ishlaydi?
4. Xanoy minorasi uchun yurishlar soni qancha?
5. Qaysi hollarda rekursiyadan foydalanish qulay?

Adabiyotlar: [4], [6]

19-AMALIY MASHG'ULOT. STANDART KUTUBXONA MODULLARIDAN FOYDALANISH (MATH, RANDOM, DATETIME)

Maqsad: standart kutubxona modullaridan (math, random, datetime, collections) foydalanish ko'nikmasini shakllantirish.

Kutilayotgan natijalar (TN8): tinglovchi modullarni import qiladi, tayyor funksiyalardan dasturlarda foydalanadi, hujjatlardan kerakli funktsiyani topadi.

Kerakli jihozlar: kompyuter, Python muhiti, docs.python.org hujjatlari.

Qisqacha nazariy ma'lumot

Modul – qayta ishlatiladigan Python fayli. import math, from math import sqrt va import random as rnd shakllari bor. math – matematik funksiyalar, random – tasodifiy sonlar, datetime – sana va vaqt, collections.Counter – sanash. Funktsiya imkoniyatlari rasmiy hujjatdan tekshiriladi.

```
import math, random, datetime
print(math.sqrt(144), math.gcd(12, 18), math.pi)
print(random.randint(1, 6), random.choice(["a", "b", "c"]))
print(datetime.date.today().strftime("%d.%m.%Y"))
```

Ishni bajarish tartibi

1-qadam. Modullar bilan tanishuv. Yuqoridagi namuna ishga tushirilib, har bir funksiya vazifasi tushuntiriladi.

2-qadam. Hujjat bilan ishlash. docs.python.org dan math modulidagi kamida 5 ta funksiya topilib, daftarga yoziladi.

3-qadam. random bilan o‘yin. Kubik tashlash (ikki kubik, yig‘indi) dasturi yoziladi.

4-qadam. datetime bilan hisoblash. Tug‘ilgan kundan hozirgi kungacha o‘tgan kunlar soni hisoblanadi.

5-qadam. Counter. Matndagi harflar chastotasi collections.Counter bilan topiladi va most_common(3) chiqariladi.

Mustaqil bajarish uchun topshiriqlar

Variant tanlanib, mashg‘ulotda ko‘rsatilgan tartibda dastur yoziladi. Topshiriq natijasi: modullardan foydalanuvchi dasturlar.

Variant	Topshiriq mavzusi
1	Kubik tashlash o‘yini (10 marta, natijalar chastotasi)
2	Tasodifiy parol generatori (harf va raqamlardan)
3	Tug‘ilgan kundan keyingi sanagacha qancha kun qolganini hisoblash
4	Aylana yuzi va uzunligi math.pi bilan; natijani math.ceil va floor bilan yaxlitlash
5	Tasodifiy 10 ta son yaratib, ularni tartiblash va eng kattasini chiqarish
6	Kvadrat tenglama ildizlari math.sqrt bilan
7	Hozirgi vaqtni turli formatlarda chiqarish
8	Ro‘yxatdan tasodifiy savol tanlovchi viktorina (random.choice)
9	Kartalar to‘plamini aralashtirish (random.shuffle) va 5 ta karta tarqatish
10	Satrdagi so‘zlar chastotasi (Counter) va eng ko‘p uchragan 3 so‘z

Nazorat savollari

1. import ning qanday shakllari bor?
2. from modul import * nima uchun tavsiya etilmaydi?
3. random.randint va random.choice nima qiladi?
4. Hujjatdan funksiya imkoniyatini qanday topasiz?
5. Counter nimani hal qiladi?

Adabiyotlar: [1], [9]

20-AMALIY MASHG‘ULOT. FAYLLARGA YOZISH VA FAYLLARDAN O‘QISH

Maqsad: fayllarga yozish va fayllardan o‘qish ko‘nikmasini shakllantirish.

Kutilayotgan natijalar (TN8): tinglovchi with operatori bilan matn fayllarini o‘qiydi va yozadi, kodlashni to‘g‘ri ko‘rsatadi, fayl natijalarini qayta ishlaydi.

Kerakli jihozlar: kompyuter, Python muhiti, namunaviy matn fayli.

Qisqacha nazariy ma’lumot

open(nom, rejim, encoding="utf-8") faylni ochadi; rejimlar: “r” (o‘qish), “w” (yozish, eskisini o‘chiradi), “a” (qo‘shish). with fayl yopilishini kafolatlaydi. Qatorlar bo‘yicha yurish – eng tejamkor o‘qish usuli. Yozishda “\n” qo‘lda qo‘shiladi.

```
with open("ismlar.txt", "w", encoding="utf-8") as f:
    for ism in ["Ali", "Vali", "Gul"]:
        f.write(ism + "\n")
with open("ismlar.txt", encoding="utf-8") as f:
    for qator in f:
        print(qator.strip())
```

Ishni bajarish tartibi

1-qadam. Yozish. Klaviaturadan o‘qilgan 5 ta ismni faylga yozuvchi dastur yoziladi.

2-qadam. O‘qish. Fayl qatorlarini o‘qib, soni va eng uzun qatori aniqlanadi.

3-qadam. Qo‘shib yozish. “a” rejimi bilan fayl oxiriga yangi qatorlar qo‘shiladi, “w” bilan farqi ko‘rsatiladi (eski ma’lumot o‘chadi).

4-qadam. Fayl nomidagi xato. Mavjud bo‘lmagan faylni ochishga urinib, FileNotFoundError ko‘riladi (qayta ishlash A22 da).

5-qadam. Matn statistikasi. Matn faylidagi qatorlar, so‘zlar va belgilar soni sanaladi.

Mustaqil bajarish uchun topshiriqlar

Variant tanlanib, mashg‘ulotda ko‘rsatilgan tartibda dastur yoziladi. Topshiriq natijasi: fayl bilan ishlovchi dasturlar.

Variant	Topshiriq mavzusi
1	Sonlarni faylga yozib, so‘ng yig‘indisini hisoblash
2	Fayldagi eng uzun so‘zni topish
3	Faylni ikkinchi faylga nusxalash (qatorlar raqamlanib)
4	Fayldagi qatorlarni teskari tartibda yangi faylga yozish
5	Ikki fayl mazmunini birlashtirish
6	Fayl qatorlarini alifbo tartibida saralab yozish
7	Fayldagi bo‘sh qatorlarni olib tashlash
8	Kundalik (diary): har kirishda sana bilan yangi yozuv qo‘shish
9	Fayldagi so‘zlar chastotasini boshqa faylga yozish
10	Fayldan sonlarni o‘qib, juftlarini alohida faylga yozish

Nazorat savollari

1. “r”, “w” va “a” rejimlarining farqi nima?
2. with operatorining afzalligi nima?
3. encoding=”utf-8” nima uchun ko’rsatiladi?
4. Faylni qator-qator o’qishning afzalligi nima?
5. “w” rejimi qanday xavf tug’diradi?

Adabiyotlar: [1], [2]

21-AMALIY MASHG‘ULOT. CSV VA JSON FAYLLARI BILAN ISHLASH

Maqsad: CSV va JSON fayllar bilan ishlash ko’nikmasini shakllantirish.

Kutilayotgan natijalar (TN8): tinglovchi csv va json modullari yordamida jadval va ichma-ich ma’lumotlarni o’qiydi, yozadi va qayta ishlaydi.

Kerakli jihozlar: kompyuter, Python muhiti, namunaviy CSV fayl.

Qisqacha nazariy ma’lumot

CSV jadval ma’lumotini matn ko’rinishida saqlaydi; csv.writer va csv.DictReader bilan ishlanadi, qiymatlar satr sifatida o’qiladi. JSON lug‘at va ro‘yxatlarni matn shaklida saqlaydi: json.dump va json.load; ensure_ascii=False o‘zbek harflarini saqlaydi.

```
import csv, json
with open("baholar.csv", newline="", encoding="utf-8") as f:
    oqish = list(csv.DictReader(f))
    ortacha = sum(int(q["ball"]) for q in oqish) / len(oqish)
with open("hisobot.json", "w", encoding="utf-8") as f:
    json.dump({"ortacha": ortacha, "soni": len(oqish)}, f,
    ensure_ascii=False, indent=2)
```

Ishni bajarish tartibi

1-qadam. CSV yaratish. 5 ta o’quvchi ismi va ballini csv.writer bilan faylga yozing.

2-qadam. CSV o’qish. DictReader bilan o’qib, o’rtacha ball va eng yuqori ball topiladi.

3-qadam. JSON yozish. Lug‘at va ro‘yxatdan iborat murakkab ma’lumot JSON faylga saqlanadi.

4-qadam. JSON o’qish. Fayl qayta o’qilib, ma’lumot turlari (lug‘at, ro‘yxat) tekshiriladi.

5-qadam. Konvertatsiya. CSV ma’lumotlari JSON formatga o’tkaziladi.

Mustaqil bajarish uchun topshiriqlar

Variant tanlanib, mashg‘ulotda ko’rsatilgan tartibda dastur yoziladi. Topshiriq natijasi: cSV va JSON bilan ishlovchi dasturlar.

Variant	Topshiriq mavzusi
1	Mahsulotlar CSV fayli: umumiy summa va eng qimmat mahsulot
2	O'quvchilar baholari: fan bo'yicha o'rtacha
3	Telefon kitobini JSON ga saqlash va qayta yuklash
4	Ob-havo ma'lumotlari CSV: eng issiq kun
5	Sinf jadvali CSV dan hafta kuni bo'yicha dars soni
6	Dastur sozlamalarini JSON da saqlash (nom, til, mavzu)
7	Viktorina savollarini JSON da saqlash va o'qish
8	CSV dan ballari 85 dan yuqorilarini yangi CSV ga yozish
9	Ikki CSV ni birlashtirish (umumiy ro'yxat)
10	JSON ma'lumotini CSV jadvalga o'tkazish

Nazorat savollari

1. CSV nima va u qanday tuzilgan?
2. CSV dan o'qilgan qiymatlar qanday turda bo'ladi?
3. JSON nima va qayerda qo'llanadi?
4. json.dump va json.load ning farqi nima?
5. ensure_ascii=False nima uchun kerak?

Adabiyotlar: [1], [9]

22-AMALIY MASHG'ULOT. ISTISNOLARNI QAYTA ISHLASH. KIRITILGAN MA'LUMOTLARNI TEKSHIRISH

Maqsad: istisnolarni qayta ishlash (try/except) va kiritilgan ma'lumotni tekshirish ko'nikmasini shakllantirish.

Kutilayotgan natijalar (TN8): tinglovchi xatolarga chidamli dasturlar yozadi, istisno turlarini ajratadi, foydalanuvchi kiritishini tekshiradi.

Kerakli jihozlar: kompyuter, Python muhiti.

Qisqacha nazariy ma'lumot

Istisno – dastur bajarilishi paytidagi xato. try/except uni ushlab, dasturni davom ettiradi. Kutilgan istisnolar nomi bilan ushlanadi (ValueError, ZeroDivisionError, FileNotFoundError); yalang except: xavfli. else xato bo'lmaganda, finally har doim bajariladi; raise istisno chaqiradi.

```
def son_ol(xabar):
    while True:
        try:
            return int(input(xabar))
        except ValueError:
            print("Butun son kiriting!")
```

Ishni bajarish tartibi

1-qadam. Istisnolarni ko‘rish. `int(“abc”), 10 / 0, [1][5], {}[“k”]` ifodalaridan qanday istisno chiqishi daftarga yoziladi.

2-qadam. try/except. Bo‘lishdan oldin `ValueError` va `ZeroDivisionError` ushlanadigan dastur yoziladi.

3-qadam. Kiritishni tekshirish. `son_ol()` funksiyasi yoziladi va undan 3 ta dasturda foydalaniladi.

4-qadam. Fayl xatosi. Mavjud bo‘lmagan faylni ochishda `FileNotFoundError` ushlanadi.

5-qadam. raise va finally. Yoshni tekshiruvchi funksiya manfiy qiymatda `ValueError` chaqiradi; finally bloki sinaladi.

Mustaqil bajarish uchun topshiriqlar

Variant tanlanib, mashg‘ulotda ko‘rsatilgan tartibda dastur yoziladi. Topshiriq natijasi: xatolarga chidamli dasturlar.

Variant	Topshiriq mavzusi
1	Butun son kiritilguncha so‘rash
2	Kasr son kiritilguncha so‘rash
3	Kalkulyator: nolga bo‘lish va noto‘g‘ri amal xatolari
4	Ro‘yxat elementiga murojaat: <code>IndexError</code> ni qayta ishlash
5	Lug‘atdan kalit bo‘yicha olish: <code>KeyError</code> ni qayta ishlash
6	Faylni xavfsiz ochish va o‘qish
7	Yosh va ism tekshiruvi (<code>raise ValueError</code>)
8	Oraliqdagi (1..10) sonni kiritish funksiyasi
9	Sana kiritishni tekshirish (kun, oy, yil)
10	Foydalanuvchi parolini tekshirish (uzunlik, raqam; xato bo‘lsa <code>raise</code>)

Nazorat savollari

1. Istisno nima?
2. try, except, else va finally nima vazifani bajaradi?
3. Nima uchun yalang except: dan qochiladi?
4. raise nima uchun ishlatiladi?
5. Kiritishni qanday ishonchli tekshirish mumkin?

Adabiyotlar: [1], [2]

23-AMALIY MASHG‘ULOT. SINFLAR VA OB‘YEKTLAR YARATISH

Maqsad: sinflar va ob‘yektlar yaratish, atributlar va metodlardan foydalanish ko‘nikmasini shakllantirish.

Kutilayotgan natijalar (TN8): tinglovchi sinf yaratadi, ob'yektlar hosil qiladi, `__init__` va `__str__` dan foydalanadi, ob'yektlar ro'yxati bilan ishlaydi.

Kerakli jihozlar: kompyuter, Python muhiti.

Qisqacha nazariy ma'lumot

Sinf – ob'yektlar andozasi, ob'yekt – uning nusxasi. `__init__` boshlang'ich qiymat beradi, `self` joriy ob'yektga ishora qiladi. Metod – sinf ichidagi funksiya. `__str__` `print()` uchun matn ko'rinishini belgilaydi. Sinf nomi katta harf bilan yoziladi.

```
class Talaba:
    def __init__(self, ism, ball):
        self.ism = ism
        self.ball = ball
    def baho(self):
        return 5 if self.ball >= 86 else 4 if self.ball >= 71 else 3
    def __str__(self):
        return f"{self.ism}: {self.ball} ({self.baho()})"
guruh = [Talaba("Ali", 90), Talaba("Vali", 75)]
for t in guruh:
    print(t)
```

Ishni bajarish tartibi

1-qadam. Sinf loyihasi. Tanlangan ob'yekt (kitob, mahsulot, o'yinchi) uchun atribut va metodlar qog'ozda yoziladi.

2-qadam. Sinf yozish. `__init__`, 2 ta metod va `__str__` bilan sinf kodlanadi.

3-qadam. Ob'yektlar yaratish. Kamida 3 ta ob'yekt yaratilib, metodlari chaqiriladi.

4-qadam. Ob'yektlar ro'yxati. Ob'yektlar ro'yxatga solinib, sikl bilan chiqariladi va biror atribut bo'yicha saralanadi.

5-qadam. self xatosi. `self` ni qasddan tushirib qoldirib, xato xabari o'qiladi va tuzatiladi.

Mustaqil bajarish uchun topshiriqlar

Variant tanlanib, mashg'ulotda ko'rsatilgan tartibda dastur yoziladi. Topshiriq natijasi: sinflar va ob'yektlar bilan dasturlar.

Variant	Topshiriq mavzusi
1	Kitob sinfi (nomi, muallifi, sahifalari; o'qilgan sahifa ulushi)
2	Bank hisobi sinfi (qo'shish, yechish, balans)
3	To'rtburchak sinfi (yuza, perimetr)
4	Mahsulot sinfi (narx, chegirma, yakuniy narx)
5	Hayvon sinfi (ism, yosh, ovoz berish metodi)
6	O'yin qahramoni sinfi (salomatlik, hujum, zarba olish)
7	Nuqta sinfi (koordinata, masofa, siljitish)

Variant	Topshiriq mavzusi
8	Kasr sinfi (surat, maxraj, soddalashtirish)
9	Avtomobil sinfi (yoqilg'i, yurish, bak to'ldirish)
10	Talaba sinfi (ismi, ballari ro'yxati, o'rtacha ball)

Nazorat savollari

1. Sinf va ob'yektning farqi nima?
2. self nima va nima uchun kerak?
3. __init__ qachon chaqiriladi?
4. Metod oddiy funksiyadan nimasi bilan farq qiladi?
5. __str__ nima vazifani bajaradi?

Adabiyotlar: [1], [5]

24-AMALIY MASHG'ULOT. VORISLIK VA POLIMORFIZMGA DOIR MASALALAR

Maqsad: vorislik va polimorfizm bilan ishlash, sinflar ierarxiyasini loyihalash ko'nikmasini shakllantirish.

Kutilayotgan natijalar (TN8): tinglovchi asosiy va hosila sinflarni yaratadi, super() dan foydalanadi, metodni qayta aniqlaydi va polimorfizmni namoyish etadi.

Kerakli jihozlar: kompyuter, Python muhiti.

Qisqacha nazariy ma'lumot

Vorislik hosila sinfga asosiy sinf xususiyatlarini beradi: class Hosila(Asosiy). super() asosiy sinf metodlarini chaqiradi. Metodni qayta aniqlash (override) hosila sinfda bir xil nomli metodni o'zgartiradi. Polimorfizm – bir xil nomli metodni turli ob'yektlar turlicha bajarishi. Vorislik “is-a” munosabatida qo'llanadi.

```
class Shakl:
    def __init__(self, nom):
        self.nom = nom
    def yuza(self):
        return 0
class Doira(Shakl):
    def __init__(self, r):
        super().__init__("doira")
        self.r = r
    def yuza(self):
        return 3.14159 * self.r ** 2
class Kvadrat(Shakl):
    def __init__(self, a):
        super().__init__("kvadrat")
        self.a = a
    def yuza(self):
        return self.a ** 2
for s in [Doira(2), Kvadrat(3)]:
    print(s.nom, s.yuza())
```

Ishni bajarish tartibi

1-qadam. Ierarxiya chizmasi. Tanlangan mavzu uchun sinflar ierarxiyasi (is-a) qog'ozda chiziladi.

2-qadam. Asosiy sinf. Umumiy atribut va metodlarga ega asosiy sinf yoziladi.

3-qadam. Hosila sinflar. Kamida 2 ta hosila sinf `super()` orqali yoziladi va metod qayta aniqlanadi.

4-qadam. Polimorfizm. Turli ob'yektlar ro'yxati sikl bilan aylanib, bir xil nomli metod chaqiriladi.

5-qadam. Tahlil. Qaysi joyda kompozitsiya vorislikdan yaxshiroq ekanligi juftlikda muhokama qilinadi.

Mustaqil bajarish uchun topshiriqlar

Variant tanlanib, mashg'ulotda ko'rsatilgan tartibda dastur yoziladi. Topshiriq natijasi: vorislik ierarxiyasi dasturlari.

Variant	Topshiriq mavzusi
1	Shakllar: Shakl, Doira, Kvadrat, Uchburchak
2	Hayvonlar: Hayvon, Mushuk, It, Qush (ovoz, harakat)
3	Transport: Transport, Avtomobil, Velosiped, Samolyot
4	Xodimlar: Xodim, Menejer, Dasturchi (maosh hisoblash)
5	O'yin qahramonlari: Qahramon, Jangchi, SehrGAR
6	Maktab a'zolari: Shaxs, O'quvchi, O'qituvchi
7	Hisoblar: Hisob, JamgarmaHisob, KreditHisob
8	Mahsulotlar: Mahsulot, Oziq-ovqat (yaroqlilik muddati), Elektronika (kafolat)
9	Kutubxona: Material, Kitob, Jurnal, Disk
10	Fayllar: Fayl, MatnFayl, RasmFayl (hajm va tavsif)

Nazorat savollari

1. Vorislik nima va u qachon qo'llanadi?
2. `super()` nima uchun kerak?
3. Metodni qayta aniqlash (`override`) nima?
4. Polimorfizmni misol bilan tushuntiring.
5. Vorislik va kompozitsiyani qanday farqlaysiz?

Adabiyotlar: [1], [5]

25-AMALIY MASHG'ULOT. PUFAKCHALI, TANLASH VA QO'SHISH USULIDA SARALASH ALGORITMLARINI DASTURLASH

Maqsad: pufakchali, tanlash va qo'shish usulida saralash algoritmlarini dasturlash va solishtirish ko'nikmasini shakllantirish.

Kutilayotgan natijalar (TN7): tinglovchi uchta saralash algoritmini yozadi, taqqoslashlar va almashtirishlar sonini sanaydi, murakkablikni eksperiment bilan tasdiqlaydi.

Kerakli jihozlar: kompyuter, Python muhiti, VisuAlgo (visualgo.net).

Qisqacha nazariy ma'lumot

Pufakchali saralash qo'shni elementlarni almashtiradi; tanlash orqali saralash har safar eng kichik elementni o'z o'rniga qo'yadi; qo'shish orqali saralash elementni tartiblangan qismga kiritadi. Uchalasi $O(n^2)$. Amalda tayyor sorted() ($O(n \log n)$) ishlatiladi.

```
def tanlash(a):
    n = len(a)
    for i in range(n - 1):
        m = i
        for j in range(i + 1, n):
            if a[j] < a[m]:
                m = j
        a[i], a[m] = a[m], a[i]
    return a
```

Ishni bajarish tartibi

1-qadam. Vizual tanishuv. VisuAlgo da uch usulning animatsiyasi ko'riladi, qadamlar daftarga yoziladi.

2-qadam. Pufakchali saralash. Algoritm yoziladi va 6 elementli ro'yxatda har bir o'tish natijasi chiqariladi.

3-qadam. Tanlash va qo'shish. Qolgan ikki algoritm yoziladi.

4-qadam. Hisoblagichlar. Har bir algoritmgaga taqqoslashlar va almashtirishlar hisoblagichi qo'shiladi.

5-qadam. Eksperiment. 1000, 2000 va 4000 elementli ro'yxatlarda vaqt o'lchanadi; n ikki baravar ortganda vaqt taxminan 4 baravar o'sishi tekshiriladi.

Mustaqil bajarish uchun topshiriqlar

Variant tanlanib, mashg'ulotda ko'rsatilgan tartibda dastur yoziladi. Topshiriq natijasi: uch saralash algoritmi dasturi.

Variant	Topshiriq mavzusi
1	Pufakchali saralashni kamayish bo'yicha yozish
2	Tanlash orqali saralashni kamayish bo'yicha yozish

Variant	Topshiriq mavzusi
3	Qo'shish orqali saralash va qadamlarni chiqarish
4	Saralashni optimallashtirish: almashtirish bo'lmasa to'xtash
5	Satrlar ro'yxatini alifbo tartibida saralash
6	Talabalarni ball bo'yicha saralash (juftliklar ro'yxati)
7	Ikki tartiblangan ro'yxatni birlashtirish (merge)
8	Birlashtirib saralash (mergesort) rekursiv
9	Tez saralash (quicksort) rekursiv
10	Uch algoritmnining vaqtini solishtiruvchi jadval va grafik

Nazorat savollari

1. Pufakchali saralash qanday ishlaydi?
2. Tanlash va qo'shish usullarining g'oyasi nimada?
3. Nima uchun bu usullar $O(n^2)$?
4. n ikki baravar ortganda $O(n^2)$ algoritmi vaqti qanday o'zgaradi?
5. Amalda qaysi saralash ishlatiladi?

Adabiyotlar: [6], [12]

26-AMALIY MASHG'ULOT. CHIZIQLI VA IKKILIK QIDIRUV ALGORITMLARI

Maqsad: chiziqli va ikkilik qidiruv algoritmlarini dasturlash va solishtirish ko'nikmasini shakllantirish.

Kutilayotgan natijalar (TN7): tinglovchi ikkala qidiruvni yozadi, qadamlar sonini hisoblaydi, ikkilik qidiruv uchun tartiblanganlik shartini tushunadi.

Kerakli jihozlar: kompyuter, Python muhiti.

Qisqacha nazariy ma'lumot

Chiziqli qidiruv $O(n)$, ikkilik qidiruv $O(\log n)$ va tartiblangan ro'yxatni talab qiladi. Ikkilik qidiruv oraligni har safar yarmiga qisqartiradi. Python da bisect moduli tayyor ikkilik qidiruv funksiyalarini beradi.

```
def ikkilik(a, x):
    p, y = 0, len(a) - 1
    qadam = 0
    while p <= y:
        qadam += 1
        o = (p + y) // 2
        if a[o] == x:
            return o, qadam
        if a[o] < x:
            p = o + 1
        else:
            y = o - 1
    return -1, qadam
```

Ishni bajarish tartibi

1-qadam. Unplugged o‘yin. 1 dan 100 gacha o‘ylangan sonni “katta-kichik” javoblari bilan topish o‘yini o‘ynaladi; savollar soni hisoblanadi.

2-qadam. Chiziqli qidiruv. Funksiya yoziladi va qadamlar hisoblagichi qo‘shiladi.

3-qadam. Ikkilik qidiruv. Namuna kiritiladi va 1000 elementli ro‘yxatda qadamlar soni tekshiriladi.

4-qadam. Solishtirish. Eng yomon holatlar uchun ikkala usulning qadamlar soni jadvalga yoziladi ($n = 100, 1000, 10000$).

5-qadam. Bisect. bisect.bisect_left bilan natija solishtiriladi.

Mustaqil bajarish uchun topshiriqlar

Variant tanlanib, mashg‘ulotda ko‘rsatilgan tartibda dastur yoziladi. Topshiriq natijasi: qidiruv algoritmlari dasturi.

Variant	Topshiriq mavzusi
1	Ikkilik qidiruvning rekursiv varianti
2	Elementning birinchi uchrashgan o‘rnini ikkilik qidiruv bilan topish
3	Ikkilik qidiruv orqali sonning kvadrat ildizini (butun qismini) topish
4	Qidiruvda qadamlar sonini jadval va grafikda ko‘rsatish
5	Ismlar ro‘yxatida alifbo bo‘yicha ikkilik qidiruv
6	“Sonni top” o‘yini: kompyuter ikkilik qidiruv bilan taxmin qiladi
7	Element nechta marta uchrashganini ikkilik qidiruv bilan topish
8	Tartiblangan ro‘yxatga element qo‘shishda o‘rnini topish
9	Eng yaqin qiymatni topish
10	Ikkilik qidiruv “javob bo‘yicha” (minimal x topish, $f(x) \geq k$)

Nazorat savollari

1. Ikkilik qidiruv uchun nima talab qilinadi?
2. Ikkilik qidiruvning murakkabligi nima uchun $O(\log n)$?
3. 1000 elementli ro‘yxatda ikkilik qidiruv necha qadam talab qiladi?
4. Chiziqli qidiruv qachon ma’qul?
5. bisect moduli nima beradi?

Adabiyotlar: [6], [12]

27-AMALIY MASHG‘ULOT. STEK VA NAVBATNI DASTURLASH

Maqsad: stek va navbat tuzilmalarini dasturlash va ularni masalalarda qo‘llash ko‘nikmasini shakllantirish.

Kutilayotgan natijalar (TN7): tinglovchi Stek va Navbat sinflarini yozadi, qavslar balansi kabi masalalarni stek bilan yechadi.

Kerakli jihozlar: kompyuter, Python muhiti.

Qisqacha nazariy ma'lumot

Stek – LIFO (oxirgi kirgan birinchi chiqadi): push, pop, peek. Navbat – FIFO: enqueue, dequeue. Stek list bilan, navbat collections.deque bilan samarali amalga oshiriladi. Stek qavslar balansi, “bekor qilish”, teskari yozishda; navbat vazifalar rejalashtirishda ishlatiladi.

```
class Stek:
    def __init__(self):
        self.a = []
    def push(self, x):
        self.a.append(x)
    def pop(self):
        if not self.a:
            raise IndexError("Stek bo'sh")
        return self.a.pop()
    def bosh_mi(self):
        return not self.a
```

Ishni bajarish tartibi

1-qadam. Stek sinfi. Yuqoridagi sinf yoziladi va sinab ko‘riladi.

2-qadam. Navbat sinfi. deque asosida Navbat sinfi (enqueue, dequeue, bosh_mi) yoziladi.

3-qadam. Qavslar balansi. Stek yordamida “[{()}]” va “[()]” satrlarini tekshiruvchi funksiya yoziladi.

4-qadam. Teskari yozish. Satr belgilari stekka solinib, teskari tartibda olinadi.

5-qadam. Printer navbati. Navbat yordamida chop etish vazifalarini ketma-ket bajaruvchi simulyatsiya yoziladi.

Mustaqil bajarish uchun topshiriqlar

Variant tanlanib, mashg‘ulotda ko‘rsatilgan tartibda dastur yoziladi. Topshiriq natijasi: stek va navbat sinflari.

Variant	Topshiriq mavzusi
1	Qavslar balansi (uch turdagi qavs)
2	Satrn stek bilan teskari yozish
3	Brauzer tarixi: “orqaga” va “oldinga” (ikki stek)
4	Teskari Polsha yozuvi (postfiks ifoda) kalkulyatori
5	Printer navbati simulyatsiyasi
6	Bank navbati simulyatsiyasi (mijozlar kelishi va xizmat)
7	“Bekor qilish” (undo) bilan matn muharriri modeli

Variant	Topshiriq mavzusi
8	Navbat yordamida “Sanash” o‘yini (Josephus masalasi)
9	Ikki stek bilan navbatni amalga oshirish
10	Berilgan sonning ikkilik yozuvini stek yordamida hosil qilish

Nazorat savollari

1. Stek va navbat tamoyillari qanday?
2. Navbat uchun nima uchun deque ishlatiladi?
3. Stek qayerlarda qo‘llanadi?
4. Qavslar balansi algoritmini tushuntiring.
5. Postfiks ifoda stekda qanday hisoblanadi?

Adabiyotlar: [6], [12]

28-AMALIY MASHG‘ULOT. GRAFLARDA KENGLIK VA CHUQURLIK BO‘YICHA QIDIRUV (BFS, DFS) ASOSLARI

Maqsad: graflarni ifodalash va kenglik hamda chuqurlik bo‘yicha qidiruv (BFS, DFS) asoslarini dasturlash ko‘nikmasini shakllantirish.

Kutilayotgan natijalar (TN7): tinglovchi grafni qo‘shnilik ro‘yxati bilan beradi, BFS va DFS ni yozadi, eng qisqa yo‘l va bog‘langanlikni topadi.

Kerakli jihozlar: kompyuter, Python muhiti, VisuAlgo (ixtiyoriy).

Qisqacha nazariy ma’lumot

Graf uchlar va qirralardan iborat. Qo‘shnilik ro‘yxati Python lug‘ati ko‘rinishida beriladi. BFS navbat yordamida qatlam-qatlam yuradi va qirralar soni bo‘yicha eng qisqa yo‘lni topadi; DFS stek yoki rekursiya bilan chuqurlikka boradi. Ikkalasi ham ko‘rilgan uchlar to‘plamini saqlaydi.

```
from collections import deque
graf = {1: [2, 3], 2: [1, 4], 3: [1, 4], 4: [2, 3, 5], 5: [4]}
def bfs(bosh):
    masofa = {bosh: 0}
    navbat = deque([bosh])
    while navbat:
        u = navbat.popleft()
        for v in graf[u]:
            if v not in masofa:
                masofa[v] = masofa[u] + 1
                navbat.append(v)
    return masofa
print(bfs(1))
```

Ishni bajarish tartibi

1-qadam. Graf modeli. Maktab xonalari yoki shaharlar xaritasi graf sifatida chiziladi, qo‘shnilik ro‘yxati tuziladi.

2-qadam. BFS. Namuna ishga tushirilib, uchlargacha masofalar tekshiriladi.

3-qadam. DFS. Rekursiv DFS yoziladi va BFS bilan yurish tartibi solishtiriladi.

4-qadam. Yo‘lni tiklash. BFS ga “ota” lug‘ati qo‘shilib, ikki uch orasidagi qisqa yo‘l chiqariladi.

5-qadam. Bog‘langanlik. Graf bog‘langan yoki yo‘qligi, komponentalar soni aniqlanadi.

Mustaqil bajarish uchun topshiriqlar

Variant tanlanib, mashg‘ulotda ko‘rsatilgan tartibda dastur yoziladi. Topshiriq natijasi: bfs va DFS dasturlari.

Variant	Topshiriq mavzusi
1	Ikki uch orasidagi eng qisqa yo‘lni chiqarish
2	Grafdagi bog‘langan komponentalar soni
3	Labirintda (matritsa) chiqish yo‘lini BFS bilan topish
4	Uch qatlam masofasidagi do‘stlarni topish (ijtimoiy tarmoq modeli)
5	DFS bilan grafdagi barcha uchlarni tartib bilan chiqarish
6	Grafda tsikl borligini tekshirish
7	Orollar soni (0 va 1 lardan iborat matritsada)
8	Berilgan uchdan eng uzoq uchni topish
9	Ikki bo‘lakli (bipartit) graf tekshiruv
10	Shaharlar orasidagi yo‘llar grafi bo‘yicha eng kam yo‘l soni

Nazorat savollari

1. Graf nima va u qanday ifodalanadi?
2. BFS va DFS ning farqi nima?
3. BFS nima uchun eng qisqa yo‘lni beradi?
4. Ko‘rilgan uchlar to‘plami nima uchun kerak?
5. Graflar qaysi real masalalarda uchraydi?

Adabiyotlar: [6], [12]

29-AMALIY MASHG‘ULOT. C++ DA BIRINCHI DASTUR, KIRITISH-CHIQUARISH VA SHARTLI OPERATORLAR

Maqsad: C++ da birinchi dasturni yozish, kiritish-chiqarish va shartli operatorlarni qo‘llash ko‘nikmasini shakllantirish.

Kutilayotgan natijalar (TN6): tinglovchi C++ muhitini sozlaydi, dastur yozib kompilyatsiya qiladi, cin/cout va if-else bilan masalalar yechadi.

Kerakli jihozlar: kompyuter, C++ kompilyatori (g++, Code::Blocks yoki VS Code), yoki onlayn kompilyator.

Qisqacha nazariy ma'lumot

C++ dasturi main() funksiyasidan boshlanadi; turlar e'lon qilinadi; buyruqlar ; bilan tugaydi. cin >> o'qiydi, cout << chiqaradi. Shart: if (shart) { } else { }. Butun sonlar bo'linishi butun natija beradi. int to'lib ketishi mumkin, katta sonlar uchun long long.

```
#include <iostream>
using namespace std;
int main() {
    int a, b;
    cin >> a >> b;
    if (a > b) cout << a << " katta\n";
    else if (a < b) cout << b << " katta\n";
    else cout << "teng\n";
    return 0;
}
```

Ishni bajarish tartibi

1-qadam. Muhitni sozlash. Kompilyator o'rnatiladi yoki onlayn muhit tanlanadi; salom dasturi kompilyatsiya qilinadi (g++ dastur.cpp -o dastur).

2-qadam. Kiritish-chiqarish. Ikki sonni o'qib, yig'indisi, ayirmasi, ko'paytmasi va bo'linmasi (butun va haqiqiy) chiqariladi.

3-qadam. Shartli operator. Namuna dastur yoziladi va 3 xil kirish bilan sinalladi.

4-qadam. Python bilan solishtirish. Bir xil masala Python va C++ da yozilib, farqlar jadvalga kiritiladi.

5-qadam. Kompilyatsiya xatolari. ; ni tushirib qoldirish yoki turni noto'g'ri e'lon qilishdan paydo bo'ladigan kompilyator xabarlari o'qiladi.

Mustaqil bajarish uchun topshiriqlar

Variant tanlanib, mashg'ulotda ko'rsatilgan tartibda dastur yoziladi. Topshiriq natijasi: c++ dasturlari to'plami.

Variant	Topshiriq mavzusi
1	Ikki sonning kattasini topish
2	Sonning juft-toqligini aniqlash
3	Uch sonning o'rtachasini hisoblash (double)
4	Yilning kabisaligini aniqlash
5	Kvadrat tenglama ildizlari (diskriminant bo'yicha)
6	Baho (5/4/3/2) ni ball bo'yicha aniqlash
7	Uchburchak mavjudligini tekshirish
8	Uch xonali son raqamlari yig'indisi
9	Soniyalarni soat:daqiqa:soniyaga o'tkazish
10	int to'lib ketishini namoyish etish va long long bilan tuzatish

Nazorat savollari

1. C++ dasturining minimal tuzilishi qanday?
2. cin va cout qanday ishlaydi?
3. C++ da $7 / 2$ natijasi nima?
4. int to'lib ketishi nima?
5. Python va C++ da shartli operator sintaksisi qanday farq qiladi?

Adabiyotlar: [7], [10]

30-AMALIY MASHG'ULOT. C++ DA SIKLLAR VA MASSIVLAR

Maqsad: C++ da sikllar va massivlar bilan ishlash ko'nikmasini shakllantirish.

Kutilayotgan natijalar (TN6): tinglovchi for/while sikllari va bir o'lchovli massivlar bilan dasturlar yozadi, massiv chegarasiga e'tibor beradi.

Kerakli jihozlar: kompyuter, C++ kompilyatori.

Qisqacha nazariy ma'lumot

for (int i = 0; i < n; i++) sikli to'rt qismdan iborat. Massiv: int a[100]; indeks 0 dan boshlanadi, chegaradan chiqish aniqlanmagan xulq beradi. Massiv o'lchamini masala chegarasidan bir oz kattaroq e'lon qilish odat.

```
#include <iostream>
using namespace std;
int a[100005];
int main() {
    int n;
    cin >> n;
    long long s = 0;
    for (int i = 0; i < n; i++) {
        cin >> a[i];
        s += a[i];
    }
    cout << s << "\n";
}
```

Ishni bajarish tartibi

1-qadam. Sikl. 1 dan n gacha yig'indi va ko'paytma (long long) hisoblanadi.

2-qadam. Massivga o'qish. n ta sonni o'qib, yig'indi va o'rtacha hisoblanadi.

3-qadam. Ekstremumlar. Massivdagi eng katta va eng kichik element va ularning indeksleri topiladi.

4-qadam. Chegaradan chiqish. Massiv chegarasidan tashqariga murojaat natijasi (xulq noaniqligi) kuzatiladi; xavf muhokama qilinadi.

5-qadam. Python bilan solishtirish. Natijalar va yozuvlar uzunligi Python dagi variant bilan taqqoslanadi.

Mustaqil bajarish uchun topshiriqlar

Variant tanlanib, mashg'ulotda ko'rsatilgan tartibda dastur yoziladi. Topshiriq natijasi: c++ da sikl va massiv dasturlari.

Variant	Topshiriq mavzusi
1	Massivni teskari tartibda chiqarish
2	Massivdagi juft sonlar soni va yig'indisi
3	Massivdagi eng katta elementning birinchi uchrashgan indeksi
4	Massivni pufakchali usulda saralash
5	Massiv elementlarini bir pozitsiyaga siklik siljitish
6	Massivdagi takrorlanuvchi elementlar bor-yo'qligini tekshirish
7	Ikki massivni birlashtirish
8	1 dan n gacha tub sonlar (Eratosfen g'alviri)
9	Massiv bo'yicha prefiks yig'indilar massivini yaratish
10	Massivdagi eng uzun ketma-ket o'suvchi qism uzunligi

Nazorat savollari

1. C++ da massiv qanday e'lon qilinadi?
2. for siklining qismlarini tushuntiring.
3. Massiv chegarasidan chiqish nimaga olib keladi?
4. Katta yig'indilar uchun qaysi tur ishlatiladi?
5. Python ro'yxati va C++ massivining asosiy farqi nima?

Adabiyotlar: [7], [10]

31-AMALIY MASHG'ULOT. C++ DA FUNKSIYALAR VA STL (VECTOR, SORT)

Maqsad: C++ da funksiyalar va STL (vector, sort, map) bilan ishlash ko'nikmasini shakllantirish.

Kutilayotgan natijalar (TN6): tinglovchi funksiyalar yozadi, havola orqali parametr uzatadi, vector, sort va map dan foydalanadi.

Kerakli jihozlar: kompyuter, C++ kompilyatori.

Qisqacha nazariy ma'lumot

C++ funksiyasi turi, nomi va parametrlari bilan e'lon qilinadi; & bilan havola orqali uzatilganda funksiya asl qiymatni o'zgartiradi. STL: vector – dinamik massiv, map – kalit-qiymat tuzilmasi, sort(v.begin(), v.end()) – saralash. Python dagi list va dict ga mos keladi.

```
#include <bits/stdc++.h>
using namespace std;
int main() {
    ios::sync_with_stdio(false);
    cin.tie(nullptr);
    int n;
    cin >> n;
    vector<int> v(n);
    for (auto &x : v) cin >> x;
    sort(v.begin(), v.end());
    map<int, int> chastota;
    for (int x : v) chastota[x]++;
    for (auto &p : chastota) cout << p.first << " " << p.second << "\n";
}
```

Ishni bajarish tartibi

1-qadam. Funksiya. Sonning tubligini tekshiruvchi funksiya yoziladi; main() dan chaqiriladi.

2-qadam. Havola bilan uzatish. Ikki sonni almashtiruvchi funksiya (int &a, int &b) yoziladi va qiymat bilan uzatilgan variant bilan farqi ko'rsatiladi.

3-qadam. vector. n ta son o'qilib, vector da saqlanadi, saralanadi va chiqariladi.

4-qadam. map. Namuna ishga tushirilib, sonlar chastotasi hisoblanadi; unordered_map bilan farq muhokama qilinadi.

5-qadam. Solishtirish. Bir xil masala Python (Counter, sorted) va C++ da yechiladi, kod uzunligi va bajarilish vaqti taqqoslanadi.

Mustaqil bajarish uchun topshiriqlar

Variant tanlanib, mashg'ulotda ko'rsatilgan tartibda dastur yoziladi. Topshiriq natijasi: c++ da funksiya va STL dasturlari.

Variant	Topshiriq mavzusi
1	Saralangan vector dan medianani topish
2	Yig'indisi berilgan sonlarga teng juftlikni topish
3	Eng ko'p uchraydigan sonni topish (map)
4	Vector dagi takrorlarni olib tashlash (sort va unique)
5	Satrlar vectorini uzunligi bo'yicha saralash
6	Funksiya: EKUB va EKUK
7	Funksiya: sonning raqamlari yig'indisi
8	Juftliklarni (pair) ikkinchi elementi bo'yicha saralash
9	set yordamida noyob elementlar soni
10	Vector da ikkilik qidiruv (binary_search, lower_bound)

Nazorat savollari

1. Parametrni qiymat va havola bo'yicha uzatishning farqi nima?

2. vector Python ro'yxatidan nimasi bilan farq qiladi?
3. map nima va qanday ishlatiladi?
4. sort() qanday chaqiriladi?
5. C++ da cin/cout ni tezlashtirish uchun nima qilinadi?

Adabiyotlar: [7], [10]

32-AMALIY MASHG'ULOT. TURTLE KUTUBXONASI YORDAMIDA GRAFIK DASTURLAR YARATISH

Maqsad: Turtle kutubxonasi yordamida grafik dasturlar yaratish va sikl, funksiya hamda rekursiyani vizual o'rganish ko'nikmasini shakllantirish.

Kutilayotgan natijalar (TN8): tinglovchi Turtle buyruqlari bilan shakllar chizadi, sikl va funksiyalar yordamida naqshlar hosil qiladi.

Kerakli jihozlar: kompyuter, Python muhiti (turtle standart kutubxonada).

Qisqacha nazariy ma'lumot

Turtle toshbaqasi forward(n), left(a), right(a), penup(), pendown(), goto(x, y), color() buyruqlari bilan chizadi. Ko'pburchak uchun burchak $360 / n$. Takrorlanuvchi shakllar siklda, parametrlil shakllar funksiyalarda yoziladi.

```
import turtle
t = turtle.Turtle()
t.speed(0)
def kopburchak(n, uzunlik):
    for _ in range(n):
        t.forward(uzunlik)
        t.left(360 / n)
for n in range(3, 9):
    kopburchak(n, 60)
    t.left(15)
turtle.done()
```

Ishni bajarish tartibi

1-qadam. Birinchi shakl. Kvadrat va uchburchak oldin ketma-ket buyruqlar, so'ng sikl bilan chiziladi.

2-qadam. Funksiya. kopburchak(n, uzunlik) funksiyasi yoziladi va turli parametrlar bilan chaqiriladi.

3-qadam. Naqsh. Doiralar yoki ko'pburchaklar aylantirilib, mandala naqshi hosil qilinadi.

4-qadam. Rang va to'ldirish. color(), begin_fill() va end_fill() bilan rangli shakllar chiziladi.

5-qadam. Rekursiv shakl. Oddiy rekursiv spiral yoki daraxt chiziladi (rekursiv daraxt: M18 ga qarang).

Mustaqil bajarish uchun topshiriqlar

Variant tanlanib, mashgʻulotda koʻrsatilgan tartibda dastur yoziladi. Topshiriq natijasi: turtle grafik dasturlari.

Variant	Topshiriq mavzusi
1	Yulduzcha (5 va 8 uchli)
2	Spiral (kvadrat va doiraviy)
3	Svetofo
4	Uycha (kvadrat va uchburchak tom)
5	Shaxmat taxtasi
6	Rangli doiralardan iborat mandala
7	Olimpiya halqalari
8	Rekursiv daraxt
9	Koch qor parchasi (1-2 daraja)
10	Soat siferblati (12 ta belgi)

Nazorat savollari

1. Turtle ning asosiy buyruqlari qaysilar?
2. Muntazam n-burchak uchun burchak qanday topiladi?
3. Turtle sikl tushunchasini oʻrgatishga qanday yordam beradi?
4. penup() va pendown() nima uchun kerak?
5. Turtle ning metodik afzalliklari nimalarda?

Adabiyotlar: [1], [9]

33-AMALIY MASHGʻULOT. TKINTER YORDAMIDA ODDIY GRAFIK INTERFEYSLI DASTUR (KALKULYATOR) YARATISH

Maqsad: Tkinter yordamida oddiy grafik interfeysli dastur (kalkulyator) yaratish koʻnikmasini shakllantirish.

Kutilayotgan natijalar (TN8): tinglovchi oyna, vidjetlar va hodisalarni boshqaradi, kiritishni tekshiradi va ishlaydigan kalkulyator yaratadi.

Kerakli jihozlar: kompyuter, Python muhiti (Tkinter odatda Python bilan keladi).

Qisqacha nazariy maʼlumot

Tkinter da asosiy oyna Tk(), vidjetlar: Label, Entry, Button. grid() vidjetlarni jadval koʻrinishida joylashtiradi. Tugma command parametriga funksiya nomi qavssiz beriladi. Entry matni get() bilan olinadi, Label matni config(text=...) bilan oʻzgartiriladi. mainloop() hodisalarni kutadi.

```

from tkinter import *
root = Tk()
root.title("Kalkulyator")
e1 = Entry(root, width=10); e1.grid(row=0, column=0)
e2 = Entry(root, width=10); e2.grid(row=0, column=1)
natija = Label(root, text="Natija: ")
natija.grid(row=2, column=0, columnspan=4)
def hisobla(amal):
    try:
        a, b = float(e1.get()), float(e2.get())
        r = {"+": a + b, "-": a - b, "*": a * b, "/": a / b}[amal]
        natija.config(text=f"Natija: {r}")
    except (ValueError, ZeroDivisionError):
        natija.config(text="Xato kiritish!")
for i, op in enumerate("+-*/*"):
    Button(root, text=op, command=lambda o=op: hisobla(o)).grid(row=1,
column=i)
root.mainloop()

```

Ishni bajarish tartibi

1-qadam. Eskiz. Oyna maketi qog‘ozda chiziladi, vidjetlar joylashuvi belgilanadi.

2-qadam. Oyna va vidjetlar. Namunadagi kod ishga tushirilib, har bir vidjet vazifasi aniqlanadi.

3-qadam. Hodisa. Tugma bosilganda hisobla() chaqirilishi tushuntiriladi; command=hisobla() xatosi ham ko‘rsatiladi.

4-qadam. Kiritishni tekshirish. try/except bilan noto‘g‘ri kiritish va nolga bo‘lish holatlari ishlanadi.

5-qadam. Sinov va takomillashtirish. 6 ta test bilan kalkulyator sinaladi; qo‘shimcha amal (daraja, qoldiq) qo‘shiladi.

Mustaqil bajarish uchun topshiriqlar

Variant tanlanib, mashg‘ulotda ko‘rsatilgan tartibda dastur yoziladi. Topshiriq natijasi: tkinter kalkulyatori.

Variant	Topshiriq mavzusi
1	“Salom” oynasi: ism kiritib salomlashish
2	Harorat konvertori (C va F)
3	Yoshni hisoblash dasturi (tug‘ilgan yildan)
4	Sonni top o‘yini (Tkinter)
5	Vazifalar ro‘yxati (Listbox, qo‘shish va o‘chirish)
6	BMI (tana massa indeksi) kalkulyatori
7	Valyuta konvertori (kurs kiritiladi)
8	Soat (Label ni after() bilan yangilash)
9	Kvadrat tenglama yechuvchisi
10	Rang tanlash: tugmalar bosilganda oyna foni o‘zgaradi

Nazorat savollari

1. Hodisaga asoslangan dasturlash nima?
2. mainloop() nima qiladi?
3. command parametriga funksiya nima uchun qavssiz beriladi?
4. pack, grid va place ning farqi nima?
5. GUI dasturida kiritishni qanday tekshirasiz?

Adabiyotlar: [1], [9]

34-AMALIY MASHG‘ULOT. PANDAS VA MATPLOTLIB YORDAMIDA O‘QUVCHILAR BAHOLARI JADVALINI TAHLIL QILISH

Maqsad: pandas va matplotlib yordamida o‘quvchilar baholari jadvalini tahlil qilish ko‘nikmasini shakllantirish.

Kutilayotgan natijalar (TN8): tinglovchi CSV ni o‘qiydi, tozalaydi, filtrlaydi, guruhlaydi, diagramma quradi va qisqa hisobot yozadi.

Kerakli jihozlar: kompyuter, Python muhiti, pandas va matplotlib paketlari (pip install pandas matplotlib), namunaviy CSV (o‘ylab topilgan ma’lumot).

Qisqacha nazariy ma’lumot

pandas DataFrame jadval ma’lumotlarini boshqaradi: read_csv, describe, filtrlash (df[df[“ball”] > 80]), groupby, sort_values. matplotlib diagramma chizadi. Shaxsiy ma’lumotlar o‘rniga o‘ylab topilgan yoki anonim ma’lumotlardan foydalaniladi. Paketlar versiyasi o‘zgarishi mumkin, hujjatlar tekshiriladi.

```
import pandas as pd
import matplotlib.pyplot as plt
df = pd.read_csv("baholar.csv", encoding="utf-8")
print(df.describe())
ort = df.groupby("sinf")["ball"].mean()
ort.plot(kind="bar", title="Sinflar bo'yicha o'rtacha ball")
plt.ylabel("Ball")
plt.savefig("diagramma.png", dpi=150)
```

Ishni bajarish tartibi

1-qadam. Ma’lumot tayyorlash. Kamida 30 qatorli (ism, sinf, fan, ball) o‘ylab topilgan CSV fayl yaratiladi.

2-qadam. O‘qish va ko‘rish. read_csv, head, shape, describe natijalari tahlil qilinadi.

3-qadam. Tozalash. Bo‘sh qiymatlar (isna) tekshiriladi va davolanadi (dropna yoki fillna).

4-qadam. Tahlil. Filtrlash va guruhlash orqali 3 ta savolga javob topiladi (masalan, qaysi sinfdan o‘rtacha ball eng yuqori).

5-qadam. Vizuallashtirish va hisobot. 2 ta diagramma quriladi va 5–7 jumalak hisobot yoziladi.

Mustaqil bajarish uchun topshiriqlar

Variant tanlanib, mashg‘ulotda ko‘rsatilgan tartibda dastur yoziladi. Topshiriq natijasi: tahlil hisoboti va diagrammalar.

Variant	Topshiriq mavzusi
1	Sinflar bo‘yicha o‘rtacha ball (ustunli diagramma)
2	Fanlar bo‘yicha o‘rtacha ball va eng yuqori natija
3	Ballar gistogrammasi (taqsimot)
4	Qizlar va o‘g‘il bolalar natijalarini solishtirish (o‘ylab topilgan ma’lumot)
5	Eng yaxshi 5 o‘quvchi jadvali
6	Haftalik ob-havo ma’lumoti: chiziqli grafik
7	Maktab bufeti sotuvlari: mahsulot bo‘yicha ulush (doiraviy diagramma)
8	Sport natijalari: o‘rtacha va eng yaxshi natija
9	Kunlik o‘qish vaqti va ball orasidagi bog‘liqlik (sochilish diagrammasi)
10	Mahalliy ochiq ma’lumotlar to‘plami bo‘yicha mini-tahlil

Nazorat savollari

1. DataFrame nima?
2. Ma’lumotni tozalash nima uchun kerak?
3. groupby qanday ishlaydi?
4. Qanday diagramma qachon tanlanadi?
5. Maktab ma’lumotlarini tahlil qilishda maxfiylikka qanday rioya qilinadi?

Adabiyotlar: [8], [9]

**35-AMALIY MASHG‘ULOT. GIT VA GITHUB BILAN ISHLASH:
REPOZITORIY YARATISH, COMMIT, PUSH**

Maqsad: Git va GitHub bilan ishlash: repozitoriy yaratish, commit qilish va push qilish ko‘nikmasini shakllantirish.

Kutilayotgan natijalar (TN10): tinglovchi mahalliy repozitoriy yaratadi, README yozadi, commit qiladi, GitHub ga joylaydi, tarmoq (branch) bilan ishlaydi.

Kerakli jihozlar: kompyuter, Git o‘rnatilgan, GitHub hisobi (maktab siyosati va yosh cheklovlariga mos), Internet.

Qisqacha nazariy ma’lumot

Git o‘zgarishlar tarixini saqlaydi: git init, add, commit, status, log. Masofaviy repozitoriy (GitHub) bilan git remote add, push va pull orqali ishlanadi. Tarmoq

(branch) parallel ishlash uchun. .gitignore keraksiz fayllarni chiqarib tashlaydi. Parol, kalit va shaxsiy ma'lumotlarni repozitoriyga joylamaslik kerak. Platforma interfeysi o'zgarishi mumkin: docs.github.com.

```
git init
git add .
git commit -m "Birinchi versiya"
git branch -M main
git remote add origin https://github.com/FOYDALANUVCHI/loyiha.git
git push -u origin main
```

Ishni bajarish tartibi

1-qadam. Tayyorgarlik. Git versiyasi tekshiriladi (git --version), ism va email sozlanadi (git config).

2-qadam. Mahalliy repozitoriy. Papka yaratilib, git init bajariladi, 2 ta .py fayl qo'shiladi.

3-qadam. Commit. git add va git commit bilan o'zgarishlar saqlanadi; git log bilan tarix ko'riladi; ma'noli commit xabari yoziladi.

4-qadam. README va .gitignore. README.md (maqsad, ishga tushirish, muallif) va .gitignore yoziladi.

5-qadam. GitHub ga joylash. Masofaviy repozitoriy yaratilib, push bajariladi; keyin branch yaratib, o'zgarish kiritiladi va birlashtiriladi.

Mustaqil bajarish uchun topshiriqlar

Variant tanlanib, mashg'ulotda ko'rsatilgan tartibda dastur yoziladi. Topshiriq natijasi: gitHub dagi repozitoriy.

Variant	Topshiriq mavzusi
1	Salomlashish dasturi repozitoriysi
2	Kalkulyator loyihasi repozitoriysi
3	Algoritmlar to'plami (saralash, qidirish)
4	Turtle rasmlari to'plami
5	"Sonni top" o'yini repozitoriysi
6	Viktorina dasturi (JSON savollar bilan)
7	Ma'lumotlar tahlili mini-loyihasi (CSV va diagramma)
8	O'quvchi portfoliosi (README bilan loyihalar ro'yxati)
9	Tarmoq bilan ishlash: yangi imkoniyat uchun branch va merge
10	Ikki kishilik repozitoriy (juftlikda pull va push)

Nazorat savollari

1. Versiyalarni boshqarish nima uchun kerak?
2. commit va push ning farqi nima?
3. Branch nima va merge nima?

4. README da qanday ma'lumotlar bo'lishi kerak?
5. Repozitoriyga nimalarni joylamaslik kerak?

Adabiyotlar: [11]

VI. LABORATORIYA MASHG'ULOTLARI MATERIALLARI

Laboratoriya mashg'ulotlari kompyuter sinfida o'tkaziladi va amaliy mashg'ulotlarda o'zlashtirilgan ko'nikmalarni kengroq, mustaqil loyiha shaklida qo'llashga qaratilgan. Ularda dasturlar majmuasi, o'yinlar, ma'lumotlar bilan ishlovchi dasturlar, olimpiada masalalari, vizuallashtirish, testlash va yakuniy loyiha yaratiladi. Jami 20 ta laboratoriya ishi (40 soat) rejalashtirilgan. Har bir ish quyidagi tuzilishga ega: maqsad, kutilayotgan natija, kerakli jihozlar, ishni bajarish tartibi (qadamma-qadam), topshiriqlar va nazorat savollari. Onlayn xizmatlar va kutubxonalarning interfeysi hamda imkoniyatlari o'zgarib turishi mumkinligi sababli, qadamlar umumiy tavsifda berilgan; tinglovchi joriy hujjatlarga moslab bajaradi.

Laboratoriya ishlari uchun umumiy baholash mezon:

Baho	Mezon
5 (a'lo)	Ish barcha qadamlar bo'yicha bajarilgan, dastur ishlaydi va test qilingan, kod tuzilmali va hujjatlangan; nazorat savollariga to'liq javob berilgan
4 (yaxshi)	Ish bajarilgan, dasturda kichik kamchiliklar bor; savollarga javob asosan to'g'ri
3 (qoniqarli)	Ish qisman bajarilgan; dastur to'liq ishlamaydi; savollarga javob yuzaki
2 (qoniqarsiz)	Ish bajarilmagan yoki dastur ishlamaydi

1-LABORATORIYA ISHI. CHIZIQLI VA TARMOQLANUVCHI ALGORITMLARGA DOIR DASTURLAR MAJMUASINI YARATISH

Maqsad: chiziqli va tarmoqlanuvchi algoritmlarga oid dasturlar majmuasini yaratish va testlash ko'nikmasini shakllantirish.

Kutilayotgan natija (TN8): tinglovchi 10 ta dasturdan iborat majmua yaratadi, har bir dastur uchun test jadvali tuzadi va natijalarni solishtiradi.

Kerakli jihozlar: kompyuter, Python muhiti, matn muharriri (test jadvali uchun).

Ishni bajarish tartibi

1-qadam. Reja. 10 ta masala tanlanadi (5 ta chiziqli, 5 ta tarmoqlanuvchi); har biri uchun kirish va chiqish formati yoziladi.

2-qadam. Dasturlash. Har bir masala alohida faylda yoki funksiyada yoziladi; kiritish va hisoblash qismlari ajratiladi.

```
def baho(ball):  
    if ball >= 86: return 5  
    if ball >= 71: return 4  
    if ball >= 56: return 3  
    return 2
```

3-qadam. Test jadvali. Har bir dastur uchun kamida 5 ta test (shu jumladan chekka holatlar) va kutilgan natija yoziladi.

4-qadam. Avtomatik tekshiruv. assert yordamida testlar kodga o'tkaziladi.

```
assert baho(86) == 5 and baho(71) == 4 and baho(55) == 2
```

5-qadam. Hisobot. Topilgan xatolar va tuzatishlar qisqacha yoziladi.

Topshiriqlar

1. 10 ta dasturdan iborat majmua yarating.
2. Har bir dasturga test jadvali va kamida 3 ta assert qo'shing.
3. Bitta dasturni tanlab, blok-sxemasini chizing.

Nazorat savollari

1. Chiziqli va tarmoqlanuvchi algoritmlarning farqi nima?
2. Chekka holat nima va nima uchun tekshiriladi?
3. assert nima qiladi?
4. Kod va kiritish qismlarini ajratish nima uchun foydali?

Adabiyotlar: [2], [4]

2-LABORATORIYA ISHI. TAKRORLANUVCHI JARAYONLARGA DOIR DASTURLAR MAJMUASINI YARATISH

Maqsad: takrorlanuvchi jarayonlarga oid dasturlar majmuasini yaratish va kuzatuv jadvali bilan tekshirish ko'nikmasini shakllantirish.

Kutilayotgan natija (TN8): tinglovchi for va while siklli dasturlar majmuasini yaratadi, natijalarni kuzatuv jadvali va testlar bilan tekshiradi.

Kerakli jihozlar: kompyuter, Python muhiti.

Ishni bajarish tartibi

1-qadam. Reja. 10 ta masala tanlanadi: yig'indi, hisoblagich, eng katta element, raqamlar bilan ishlash, naqshlar.

2-qadam. for dasturlari. Takrorlar soni ma'lum masalalar for bilan yoziladi.

3-qadam. while dasturlari. Shartga bog'liq masalalar while bilan yoziladi (raqamlar yig'indisi, Evklid algoritmi).

```
def ekub(a, b):  
    while b:  
        a, b = b, a % b  
    return a
```

4-qadam. Kuzatuv jadvali. Ikkita dastur uchun jadval to'ldirilib, kod natijasi bilan solishtiriladi.

5-qadam. Cheksiz sikldan himoya. Takrorlar sonini cheklovchi himoya sinab ko'riladi.

Topshiriqlar

1. 10 ta sikl dasturini yarating (5 ta for, 5 ta while).
2. Har bir dastur uchun test va kutilgan natijani yozing.
3. Ikkita dastur uchun kuzatuv jadvalini to'ldiring.

Nazorat savollari

1. for va while ni qachon tanlaymiz?
2. Kuzatuv jadvali nima beradi?
3. Cheksiz sikl qanday oldini olinadi?
4. Akkumulyator va hisoblagich farqi nima?

Adabiyotlar: [2], [4]

3-LABORATORIYA ISHI. RANDOM MODULI YORDAMIDA “SONNI TOP” O‘YININI YARATISH

Maqsad: random moduli yordamida “Sonni top” o‘yinini yaratish va uni takomillashtirish ko‘nikmasini shakllantirish.

Kutilayotgan natija (TN8): tinglovchi sikl, shart, tasodifiy son va kiritishni tekshirishni birlashtirib ishlaydigan o‘yin yaratadi.

Kerakli jihozlar: kompyuter, Python muhiti.

Ishni bajarish tartibi

1-qadam. Reja. O‘yin qoidalari yoziladi: kompyuter 1..100 oralig‘ida son o‘ylaydi, o‘yinchi taxmin qiladi, dastur “katta” yoki “kichik” deydi.

2-qadam. Asosiy sikl. Namuna asosida o‘yin yoziladi.

```
import random
son = random.randint(1, 100)
urinish = 0
while True:
    try:
        t = int(input("Taxmin: "))
    except ValueError:
        print("Son kiriting!")
        continue
    urinish += 1
    if t < son:
        print("Kattaroq")
    elif t > son:
        print("Kichikroq")
    else:
        print(f"Topdingiz! {urinish} urinishda")
        break
```

3-qadam. Takomillashtirish. Urinishlar soni cheklanadi (masalan, 7); murakkablik darajalari (oson, o‘rta, qiyin) qo‘shiladi.

4-qadam. Rekord. Eng yaxshi natija faylga saqlanadi va o‘yin boshida ko‘rsatiladi.

5-qadam. Sinov. Do'st dasturni sinab, xato va takliflarni yozadi.

Topshiriqlar

1. O'yinni yuqoridagi barcha imkoniyatlar bilan yarating.
2. Optimal strategiya (ikkilik qidiruv) bo'yicha maksimal urinishlar soni nechta ekanini hisoblang.
3. O'yinni sinfda o'tkazish bo'yicha qisqa ko'rsatma yozing.

Nazorat savollari

1. random.randint qanday ishlaydi?
2. try/except o'yinda nima uchun kerak?
3. Ikkilik qidiruv strategiyasi nechta urinishni kafolatlaydi?
4. O'yin o'quv jarayonida qanday maqsadda qo'llanishi mumkin?

Adabiyotlar: [1], [9]

4-LABORATORIYA ISHI. MATN STATISTIKASINI HISOBLOVCHI DASTUR YARATISH

Maqsad: matn statistikasini hisoblovchi dastur yaratish ko'nikmasini shakllantirish.

Kutilayotgan natija (TN8): tinglovchi fayldan matnni o'qib, so'zlar, jumlar, harflar chastotasi va eng ko'p uchraydigan so'zlarni hisoblaydi.

Kerakli jihozlar: kompyuter, Python muhiti, namunaviy matn fayli (UTF-8).

Ishni bajarish tartibi

1-qadam. Ma'lumot. Matn fayli (o'zbek tilida, 200+ so'z) tayyorlanadi.

2-qadam. O'qish va tozalash. Matn kichik harfga o'tkaziladi, tinish belgilari olib tashlanadi.

```
import string
from collections import Counter
with open("matn.txt", encoding="utf-8") as f:
    matn = f.read().lower()
for b in string.punctuation + "--<>":
    matn = matn.replace(b, " ")
sozlar = matn.split()
```

3-qadam. Hisoblash. So'zlar soni, noyob so'zlar, o'rtacha so'z uzunligi hisoblanadi.

4-qadam. Chastota. Counter bilan eng ko'p uchraydigan 10 so'z va harflar chastotasi topiladi.

5-qadam. Hisobot. Natijalar chiroyli jadval (f-string) shaklida chiqariladi va faylga yoziladi.

Topshiriqlar

1. Statistika kamida 6 ta ko'rsatkich bilan hisoblang.
2. Ikki matnni (masalan, ikki she'rning o'qilishi qiyinligi) solishtiring.
3. Apostrof turlari (o', o', o`) tufayli yuzaga keladigan muammoni tahlil qiling va yechim taklif qiling.

Nazorat savollari

1. Matnni tahlildan oldin nima uchun tozalash kerak?
2. Counter nima qiladi?
3. Noyob so'zlar soni nimani ko'rsatadi?
4. Kodlash (encoding) nima uchun muhim?

Adabiyotlar: [2], [4]

5-LABORATORIYA ISHI. MATRITSALAR USTIDA AMALLAR BAJARUVCHI DASTUR YARATISH

Maqsad: matritsalar ustida amallar bajaruvchi dastur yaratish ko'nikmasini shakllantirish.

Kutilayotgan natija (TN8): tinglovchi ikki o'lgovli ro'yxatlar bilan qo'shish, transponirlash, ko'paytirish amallarini bajaradi va natijani NumPy bilan tekshiradi.

Kerakli jihozlar: kompyuter, Python muhiti, NumPy (ixtiyoriy tekshiruv uchun).

Ishni bajarish tartibi

1-qadam. Matritsani kiritish. $n \times m$ matritsani o'qish va chiroyli chiqarish funksiyalari yoziladi.

2-qadam. Qo'shish va songa ko'paytirish. Funksiyalar yoziladi va sinaladi.

3-qadam. Transponirlash. Funksiya yoziladi.

4-qadam. Matritsalarini ko'paytirish. Uch ichma-ich sikl bilan yoziladi.

```
def kopaytir(A, B):  
    n, m, k = len(A), len(B), len(B[0])  
    return [[sum(A[i][t] * B[t][j]) for t in range(m)] for  
            i in range(n)]
```

5-qadam. NumPy bilan tekshirish. Natija numpy.dot bilan solishtiriladi.

Topshiriqlar

1. Uchta amal (qo'shish, transponirlash, ko'paytirish) funksiyalarini yozing.
2. Matritsa determinantini 2×2 va 3×3 uchun hisoblang.
3. Ko'paytirish uchun o'lgamlar mos kelmasa, xatoni qayta ishleng.

Nazorat savollari

1. Matritsalar ko'paytmasi uchun o'lgamlar qanday mos kelishi kerak?

2. Transponirlash nima?
3. Ikki o'lovli ro'yxat qanday yaratiladi?
4. Matritsani ko'paytirish murakkabligi qanday?

Adabiyotlar: [1], [4]

6-LABORATORIYA ISHI. LUG'ATLAR ASOSIDA "LUG'AT-TARJIMON" DASTURINI YARATISH

Maqsad: lug'atlar asosida "Lug'at-tarjimon" dasturini yaratish ko'nikmasini shakllantirish.

Kutilayotgan natija (TN8): tinglovchi lug'atni fayldan yuklaydi, so'zlarni tarjima qiladi, yangi so'z qo'shadi va saqlaydi.

Kerakli jihozlar: kompyuter, Python muhiti.

Ishni bajarish tartibi

1-qadam. Reja. Dastur menyusi: tarjima qilish, yangi so'z qo'shish, barcha so'zlarni ko'rish, chiqish.

2-qadam. Lug'at tuzilmasi. Dastlabki 20 ta so'zdan iborat JSON fayl tuziladi.

3-qadam. Asosiy funksiyalar. Yuklash, saqlash, tarjima funksiyalari yoziladi.

```
import json
def yukla(fayl="lugat.json"):
    try:
        with open(fayl, encoding="utf-8") as f:
            return json.load(f)
    except FileNotFoundError:
        return {}
def saqla(d, fayl="lugat.json"):
    with open(fayl, "w", encoding="utf-8") as f:
        json.dump(d, f, ensure_ascii=False, indent=2)
```

4-qadam. Menu. while sikli bilan menu tuziladi; so'z topilmasa yangi tarjima so'raladi.

5-qadam. Sinov. Dastur ishga tushirilib, ma'lumotlar saqlanishi tekshiriladi.

Topshiriqlar

1. Dasturni to'liq yarating.
2. Teskari tarjima (ikkala yo'nalishda) imkoniyatini qo'shing.
3. Kichik-katta harf va bo'shliqlarni normallashtiring.

Nazorat savollari

1. Lug'at bu masalada nima uchun qulay?
2. JSON da ma'lumot qanday saqlanadi?
3. FileNotFoundError nima uchun qayta ishlanadi?

4. Ma'lumotni normallashtirish nima beradi?

Adabiyotlar: [1], [4]

7-LABORATORIYA ISHI. FUNKSIYALAR KUTUBXONASINI (MODUL) YARATISH VA UNDAN FOYDALANISH

Maqsad: funksiyalar kutubxonasini (modul) yaratish va undan foydalanish ko'nikmasini shakllantirish.

Kutilayotgan natija (TN8): tinglovchi o'z modulini yaratadi, hujjatlaydi, import qiladi va testlaydi.

Kerakli jihozlar: kompyuter, Python muhiti.

Ishni bajarish tartibi

1-qadam. Reja. Modul tarkibi tanlanadi: geometriya, sonlar nazariyasi yoki matn funksiyalari (kamida 6 ta funksiya).

2-qadam. Modul yozish. Har bir funksiyaga docstring yoziladi.

```
# sonlar.py
def tub_mi(n):
    """n tub son bo'lsa True qaytaradi."""
    if n < 2:
        return False
    i = 2
    while i * i <= n:
        if n % i == 0:
            return False
        i += 1
    return True
if __name__ == "__main__":
    assert tub_mi(7) and not tub_mi(9)
```

3-qadam. Import. Boshqa fayldan modul import qilinib, funksiyalar chaqiriladi (import sonlar; from sonlar import tub_mi).

4-qadam. Sinov. if __name__ == "__main__": ichida testlar yoziladi; modul ham skript sifatida ham ishlashi tekshiriladi.

5-qadam. Hujjat. help(sonlar) natijasi tekshiriladi va qisqa README yoziladi.

Topshiriqlar

1. Kamida 6 funksiyali modul yarating.
2. Modulni ishlatuvchi alohida dastur yozing.
3. Fayl nomini random.py qilib qo'yib (sinab), nima bo'lishini kuzating va sababini tushuntiring.

Nazorat savollari

1. Modul nima va u nima uchun yaratiladi?
2. if __name__ == "__main__": nima qiladi?

3. Docstring nima uchun kerak?
4. Modul nomini standart modul nomi bilan bir xil qo'yish nima uchun xavfli?

Adabiyotlar: [1], [9]

8-LABORATORIYA ISHI. REKURSIV ALGORITMLAR: KOMBINATORIKA MASALALARINI DASTURLASH

Maqsad: rekursiv algoritmlar yordamida kombinatorika masalalarini dasturlash ko'nikmasini shakllantirish.

Kutilayotgan natija (TN8): tinglovchi o'rin almashtirishlar, kombinatsiyalar va qism to'plamlarni rekursiv hosil qiladi, natijalar sonini formulalar bilan solishtiradi.

Kerakli jihozlar: kompyuter, Python muhiti.

Ishni bajarish tartibi

1-qadam. Reja. Uchta masala: n belgining barcha o'rin almashtirishlari, barcha qism to'plamlar, k ta tanlash (kombinatsiya).

2-qadam. Qism to'plamlar. Har bir element uchun "olish yoki olmaslik" tanlovi rekursiya bilan yoziladi.

```
def qismlar(a, i=0, joriy=None):
    joriy = joriy or []
    if i == len(a):
        print(joriy)
        return
    qismlar(a, i + 1, joriy)
    qismlar(a, i + 1, joriy + [a[i]])
qismlar([1, 2, 3])
```

3-qadam. O'rin almashtirishlar. Rekursiv funksiya yoziladi, $n = 3$ va 4 uchun natijalar soni $n!$ bilan tekshiriladi.

4-qadam. Kombinatsiyalar. k ta element tanlash dasturi yoziladi; soni $n!/(k!(n-k)!)$ formulasi bilan solishtiriladi.

5-qadam. itertools bilan solishtirish. Natijalar `itertools.permutations` va `combinations` bilan tekshiriladi.

Topshiriqlar

1. Uch masalani rekursiv yeching.
2. Natijalar soni formulalar bilan mos kelishini jadvalda ko'rsating.
3. $n = 10$ uchun o'rin almashtirishlar soni va dastur vaqtini baholang ($n!$ o'sishi haqida xulosa yozing).

Nazorat savollari

1. Rekursiyada "tanlov" qanday amalga oshiriladi?
2. n elementli to'plamning qism to'plamlari soni nechta?

3. O‘rin almashtirishlar soni qanday o‘sadi?
4. Orqaga qaytish (backtracking) g‘oyasi nima?

Adabiyotlar: [4], [6]

9-LABORATORIYA ISHI. FAYL ASOSIDAGI “O‘QUVCHILAR RO‘YXATI” DASTURINI YARATISH (QO‘SHISH, QIDIRISH, SARALASH)

Maqsad: fayl asosidagi “O‘quvchilar ro‘yxati” dasturini (qo‘shish, qidirish, saralash) yaratish ko‘nikmasini shakllantirish.

Kutilayotgan natija (TN8): tinglovchi ma‘lumotlarni faylda saqlaydigan, qo‘shish, qidirish, tahrirlash va saralash imkoniyatiga ega dastur yaratadi.

Kerakli jihozlar: kompyuter, Python muhiti.

Ishni bajarish tartibi

1-qadam. Ma‘lumot tuzilmasi. Har bir o‘quvchi uchun ism, sinf, o‘rtacha ball (lug‘at) belgilanadi; fayl formati (CSV) tanlanadi.

2-qadam. Yuklash va saqlash. Funksiyalar yoziladi, fayl yo‘q bo‘lsa bo‘sh ro‘yxat qaytariladi.

3-qadam. Amallar. Qo‘shish, ismi bo‘yicha qidirish, ball bo‘yicha saralash va o‘chirish funksiyalari yoziladi.

```
def qidir(royxat, ism):  
    return [o for o in royxat if ism.lower() in o["ism"].lower()]  
def saralash(royxat):  
    return sorted(royxat, key=lambda o: o["ball"], reverse=True)
```

4-qadam. Menyu va xatolar. Menyu tuziladi; noto‘g‘ri kiritish try/except bilan qayta ishlanadi.

5-qadam. Sinov. 10 ta o‘ylab topilgan o‘quvchi bilan dastur sinaladi.

Topshiriqlar

1. Dasturni to‘liq yarating.
2. Sinf bo‘yicha o‘rtacha ball hisoblash imkoniyatini qo‘shing.
3. Real ismlar o‘rniga o‘ylab topilgan ma‘lumotdan foydalanishning sababini tushuntiring.

Nazorat savollari

1. Ma‘lumotlarni faylda saqlash nima beradi?
2. CSV va JSON ni solishtiring.
3. Dasturda xatolarni qayta ishlash nima uchun muhim?
4. Shaxsiy ma‘lumotlar bilan ishlashda qanday qoidalar bor?

Adabiyotlar: [1], [2]

10-LABORATORIYA ISHI. JSON ASOSIDA TEST (VIKTORINA) DASTURINI YARATISH

Maqsad: JSON asosida test (viktorina) dasturini yaratish ko'nikmasini shakllantirish.

Kutilayotgan natija (TN8): tinglovchi savollarni JSON faylda saqlaydi, tasodifiy tartibda so'raydi, ball hisoblaydi va natijani saqlaydi.

Kerakli jihozlar: kompyuter, Python muhiti.

Ishni bajarish tartibi

1-qadam. Savollar bazasi. JSON faylga kamida 10 ta savol (matn, javob variantlari, to'g'ri javob) yoziladi.

```
[
  {
    "savol": "Python da ro'yxat qaysi qavs bilan yoziladi?",
    "variantlar": ["()", "[]", "{}"], "javob": 1
  }
]
```

2-qadam. Yuklash. json.load bilan savollar o'qiladi, random.shuffle bilan aralashtiriladi.

3-qadam. Test sikli. Har bir savol chiqarilib, javob tekshiriladi, ball yig'iladi.

```
import json, random
with open("savollar.json", encoding="utf-8") as f:
    savollar = json.load(f)
    random.shuffle(savollar)
    ball = 0
    for s in savollar:
        print(s["savol"])
        for i, v in enumerate(s["variantlar"], 1):
            print(i, v)
            if int(input("Javob: ")) - 1 == s["javob"]:
                ball += 1
    print(f"Natija: {ball}/{len(savollar)}")
```

4-qadam. Natijani saqlash. Natijalar sana bilan faylga yoziladi.

5-qadam. Takomillashtirish. Qiyinlik darajasi va vaqt cheklovi qo'shiladi (ixtiyoriy).

Topshiriqlar

1. Viktorinani 10 savol bilan yarating.
2. Kiritilgan javobni tekshirish (try/except) qo'shing.
3. Savollar bazasini fan bo'yicha toifalarga ajrating.

Nazorat savollari

1. Savollarni JSON da saqlashning afzalligi nima?
2. random.shuffle nima qiladi?
3. Foydalanuvchi kiritishini nima uchun tekshiramiz?

4. Viktorinani informatika darsida qanday qo‘llash mumkin?

Adabiyotlar: [1], [9]

11-LABORATORIYA ISHI. OYD ASOSIDA “KUTUBXONA” TIZIMI MODELINI YARATISH

Maqsad: OYD asosida “Kutubxona” tizimi modelini yaratish ko‘nikmasini shakllantirish.

Kutilayotgan natija (TN8): tinglovchi Kitob, Kitobxon va Kutubxona sinflarini loyihalaydi, ularning o‘zaro ta‘sirini dasturlaydi.

Kerakli jihozlar: kompyuter, Python muhiti.

Ishni bajarish tartibi

1-qadam. Loyiha. Sinflar va ularning atributlari, metodlari qog‘ozda chiziladi (sinf diagrammasi).

2-qadam. Kitob va Kitobxon. Sinflar `__init__` va `__str__` bilan yoziladi.

3-qadam. Kutubxona. Kitoblarni qo‘shish, qidirish, berish, qaytarish metodlari yoziladi.

```
class Kutubxona:
    def __init__(self):
        self.kitoblar = []
    def qosh(self, kitob):
        self.kitoblar.append(kitob)
    def ber(self, nom, kitobxon):
        for k in self.kitoblar:
            if k.nom == nom and k.mavjud:
                k.mavjud = False
                kitobxon.olingan.append(k)
                return True
        return False
```

4-qadam. Xatolar. Mavjud bo‘lmagan yoki berilgan kitob uchun xabar beriladi.

5-qadam. Sinov. Kamida 5 ta kitob va 2 kitobxon bilan ssenariy ishga tushiriladi.

Topshiriqlar

1. Modelni to‘liq yarating.
2. Kitobxonning qarzdorlik muddatini (datetime) qo‘shing.
3. Ma’lumotlarni JSON ga saqlash va qayta yuklashni qo‘shing.

Nazorat savollari

1. Bu masalada qaysi sinflar aniqlandi va nima uchun?
2. Sinflar o‘zaro qanday bog‘langan (kompozitsiya)?
3. Inkapsulyatsiya qaysi joyda qo‘llandi?
4. Sinf diagrammasi nima uchun chiziladi?

Adabiyotlar: [1], [5]

12-LABORATORIYA ISHI. SARALASH ALGORITMLARINING SAMARADORLIGINI TAJRIBADA SOLISHTIRISH

Maqsad: saralash algoritmlarining samaradorligini tajribada solishtirish ko'nikmasini shakllantirish.

Kutilayotgan natija (TN8): tinglovchi turli hajmdagi ma'lumotlarda vaqt o'lchaydi, jadval va grafik quradi, nazariy murakkablik bilan solishtiradi.

Kerakli jihozlar: kompyuter, Python muhiti, matplotlib.

Ishni bajarish tartibi

1-qadam. Algoritmlar. Pufakchali, qo'shish, tez saralash va tayyor sorted() tayyorlanadi.

2-qadam. O'lchash funksiyasi. time.perf_counter bilan vaqt o'lchovchi funksiya yoziladi.

```
import time, random
def olcha(f, n):
    a = [random.randint(1, 10**6) for _ in range(n)]
    t = time.perf_counter()
    f(a)
    return time.perf_counter() - t
```

3-qadam. Tajriba. $n = 500, 1000, 2000, 4000$ uchun har bir algoritmi 3 marta o'lchanib, o'rtacha olinadi.

4-qadam. Jadval va grafik. Natijalar jadvalga yoziladi va matplotlib bilan grafik quriladi.

5-qadam. Tahlil. n ikki baravar ortganda vaqt o'zgarishi nazariy $O(n^2)$ va $O(n \log n)$ bilan solishtirilib, xulosa yoziladi.

Topshiriqlar

1. To'rt algoritmi solishtiruvchi dastur yarating.
2. Allaqachon tartiblangan ro'yxatda natijalar qanday o'zgarishini tekshiring.
3. O'lchash natijalari kompyuterga bog'liq ekanini hisobotda izohlang.

Nazorat savollari

1. Algoritmi samaradorligi qanday o'lchanadi?
2. Bir o'lchovni bir necha marta takrorlash nima uchun kerak?
3. Nazariya va tajriba natijalari nima uchun taxminan mos keladi?
4. Qaysi saralash amalda ishlatiladi?

Adabiyotlar: [6], [12]

13-LABORATORIYA ISHI. MA'LUMOTLAR TUZILMALARIDAN FOYDALANIB MASALALAR YECHISH

Maqsad: ma'lumotlar tuzilmalaridan foydalanib masalalar yechish ko'nikmasini shakllantirish.

Kutilayotgan natija (TN8): tinglovchi masalaga mos tuzilmani tanlaydi (list, dict, set, stek, navbat, graf) va tanlovni asoslaydi.

Kerakli jihozlar: kompyuter, Python muhiti.

Ishni bajarish tartibi

1-qadam. Masalalar to'plami. 5 ta masala beriladi: noyob so'zlar, chastota, qavslar balansi, vazifalar navbati, eng qisqa yo'l.

2-qadam. Tuzilma tanlash jadvali. Har bir masala uchun mos tuzilma va sababi jadvalga yoziladi.

3-qadam. Dasturlash. Masalalar tanlangan tuzilmalar bilan yechiladi.

```
from collections import deque, Counter
noyob = set("informatika matnidagi so'zlar".split())
chastota = Counter("abracadabra")
navbat = deque(["vazifa1", "vazifa2"])
```

4-qadam. Samaradorlikni solishtirish. Tegishlilik tekshiruvi list va set da vaqt bo'yicha solishtiriladi.

5-qadam. Himoya. Har bir tanlov og'zaki asoslanadi.

Topshiriqlar

1. 5 ta masalani yeching va tuzilma tanlovini asoslang.
2. Bitta masalani ikki xil tuzilma bilan yechib, solishtiring.
3. Tuzilma tanlash qo'llanmasi (bir sahifali) tayyorlang.

Nazorat savollari

1. Qaysi masalada set, qaysida dict afzal?
2. Navbat va stek qayerda qo'llanadi?
3. x in list va x in set farqi nima?
4. Tuzilma tanlashda qanday mezonlar hisobga olinadi?

Adabiyotlar: [6], [12]

14-LABORATORIYA ISHI. ONLAYN MASALALAR TIZIMLARIDA (CODEFORCES VA BOSHQALAR) OLIMPIADA MASALALARINI YECHISH

Maqsad: onlayn masalalar tizimlarida (Codeforces va boshqalar) olimpiada masalalarini yechish ko'nikmasini shakllantirish.

Kutilayotgan natija (TN9): tinglovchi tizimda ro'yxatdan o'tadi, masalalarni o'qiydi, yechadi, hukmlarni tahlil qiladi va hisobot yozadi.

Kerakli jihozlar: kompyuter, Internet, onlayn masalalar tizimi hisobi (maktab siyosati va yosh cheklovlariga mos).

Ishni bajarish tartibi

1-qadam. Tizim bilan tanishuv. Tizimda masala tuzilishi (shart, kirish, chiqish, chegaralar, namuna) o'rganiladi; hukmlar (Accepted, Wrong Answer, Time Limit Exceeded, Runtime Error) tushuntiriladi.

2-qadam. Masalalarni tanlash. Oson darajadagi 10 ta masala tanlanadi (qiyinlik tizimda ko'rsatilgan).

3-qadam. Yechish. Har bir masala uchun avval shart qog'ozda tahlil qilinadi, chegaralar va chekka holatlar yoziladi; so'ng kodlanadi.

```
n = int(input())
a = list(map(int, input().split()))
print(max(a) - min(a))
```

4-qadam. Hukmlar tahlili. Qabul qilinmagan har bir urinish uchun sabab (xato yechim, vaqt chegarasi, chekka holat) aniqlanadi.

5-qadam. Hisobot. Masala, g'oya, murakkablik, hukm va olingan saboq jadvalda yoziladi.

Topshiriqlar

1. Kamida 10 ta masalani yeching.
2. Kamida 2 ta masalani ikki xil usulda (sekin va tez) yeching.
3. Hukmlar tahlili jadvalini to'ldiring.

Nazorat savollari

1. Masala shartini o'qishda nimalarga e'tibor beriladi?
2. Time Limit Exceeded nimani bildiradi?
3. Chegaralar yechim tanloviga qanday ta'sir qiladi?
4. Maktabda onlayn tizimdan foydalanishda qanday qoidalar bo'lishi kerak?

Adabiyotlar: [6], [13]

15-LABORATORIYA ISHI. C++ TILIDA OLIMPIADA TIPIDAGI MASALALARNI YECHISH

Maqsad: C++ tilida olimpiada tipidagi masalalarni yechish ko'nikmasini shakllantirish.

Kutilayotgan natija (TN9): tinglovchi C++ shablonini tayyorlaydi, masalalarni yechadi, vaqt va xotira chegaralarini hisobga oladi.

Kerakli jihozlar: kompyuter, C++ kompilyatori, namunaviy test fayllari.

Ishni bajarish tartibi

1-qadam. Shablon. Olimpiada uchun C++ shabloni tayyorlanadi.

```
#include <bits/stdc++.h>
using namespace std;
int main() {
    ios::sync_with_stdio(false);
    cin.tie(nullptr);
    // yechim
    return 0;
}
```

2-qadam. Masalalar. 6 ta masala yechiladi: yig'indi, eng katta, chastota, saralash, ikkilik qidiruv, prefiks yig'indi.

3-qadam. Test. Har bir masala uchun namunaviy kirish fayli va kutilgan natija bilan tekshiriladi (cin < kirish.txt).

4-qadam. Chegaralar. Kirish hajmi katta bo'lganda $O(n^2)$ yechim sekin ishlashi tajribada ko'riladi va $O(n \log n)$ ga almashtiriladi.

5-qadam. Python bilan solishtirish. Bitta masala Python va C++ da bajarilish vaqti bo'yicha solishtiriladi.

Topshiriqlar

1. 6 ta masalani C++ da yeching.
2. Kamida 2 ta masalani tezroq algoritmi bilan qayta yozing.
3. Python va C++ vaqtlarini jadvalga kiriting.

Nazorat savollari

1. Nima uchun olimpiadada C++ keng qo'llanadi?
2. sync_with_stdio(false) nima uchun ishlatiladi?
3. long long qachon kerak bo'ladi?
4. Katta kirishda $O(n^2)$ yechimning muammosi nima?

Adabiyotlar: [7], [10]

16-LABORATORIYA ISHI. TURTLE YORDAMIDA FRAKTALLAR VA GEOMETRIK NAQSHLAR CHIZISH

Maqsad: Turtle yordamida fraktallar va geometrik naqshlar chizish ko'nikmasini shakllantirish.

Kutilayotgan natija (TN8): tinglovchi rekursiv fraktallar (daraxt, Koch egri chizig'i, Serpinskiy) va murakkab naqshlarni chizadi.

Kerakli jihozlar: kompyuter, Python muhiti (turtle).

Ishni bajarish tartibi

1-qadam. Rekursiv daraxt. Namuna ishga tushiriladi va daraxt chuqurligi o'zgartiriladi.

```

import turtle
t = turtle.Turtle(); t.speed(0); t.left(90)
def daraxt(uz, daraja):
    if daraja == 0:
        return
    t.forward(uz)
    t.left(30); daraxt(uz * 0.7, daraja - 1)
    t.right(60); daraxt(uz * 0.7, daraja - 1)
    t.left(30); t.backward(uz)
daraxt(80, 7)

```

2-qadam. Koch egri chizig‘i. Rekursiv funksiya yoziladi: har bir kesma uchta qismga bo‘linib, o‘rtasiga burchak qo‘shiladi.

3-qadam. Serpinskiy uchburchagi. Rekursiya chuqurligi 3–4 bilan chiziladi.

4-qadam. Parametrlarni o‘zgartirish. Burchak, qisqarish koeffitsienti va rang bilan tajriba o‘tkaziladi.

5-qadam. Tahlil. Chuqurlik oshganda chaqiruvlar soni qanday o‘sishi tushuntiriladi (2^d).

Topshiriqlar

1. Uchta fraktalni chizing.
2. O‘zingizning rekursiv naqshingizni yarating.
3. Fraktallarni sinfda rekursiyani tushuntirishda qanday qo‘llashni yozing.

Nazorat savollari

1. Fraktal nima?
2. Rekursiv chizmada bazaviy holat qanday?
3. Chuqurlik oshganda ish hajmi qanday o‘sadi?
4. Rekursiyani Turtle bilan o‘rgatishning afzalligi nima?

Adabiyotlar: [1], [9]

17-LABORATORIYA ISHI. TKINTER DA GRAFIK INTERFEYSLI O‘QUV DASTURINI YARATISH

Maqsad: Tkinter da grafik interfeysli o‘quv dasturini yaratish ko‘nikmasini shakllantirish.

Kutilayotgan natija (TN8): tinglovchi o‘quv maqsadidagi GUI dastur (masalan, ko‘paytirish jadvali trenajyori) yaratadi, hodisalarni boshqaradi, ma’lumotlarni saqlaydi.

Kerakli jihozlar: kompyuter, Python muhiti (Tkinter).

Ishni bajarish tartibi

1-qadam. G‘oya va eskiz. O‘quv dasturi tanlanadi (ko‘paytirish jadvali, ingliz so‘zlari, mantiqiy o‘yin); oyna eskizi chiziladi.

2-qadam. Asosiy oyna. Savol (Label), javob (Entry) va tekshirish (Button) vidjetlari yoziladi.

```
import random
from tkinter import *
a = b = ball = 0
def yangi():
    global a, b
    a, b = random.randint(2, 9), random.randint(2, 9)
    savol.config(text=f"{a} x {b} = ?"); javob.delete(0, END)
def tekshir():
    global ball
    try:
        if int(javob.get()) == a * b:
            ball += 1; natija.config(text=f"To'g'ri! Ball: {ball}")
        else:
            natija.config(text="Xato, qayta urining")
    except ValueError:
        natija.config(text="Son kiriting")
    yangi()
```

3-qadam. Interfeysni yig'ish. Vidjetlar grid bilan joylashtiriladi; oyna sarlavhasi va o'lchami beriladi.

4-qadam. Takomillashtirish. Ball, rekord, daraja tanlash va fayl orqali saqlash qo'shiladi.

5-qadam. Sinov. Foydalanuvchi sinovi o'tkaziladi: do'st dasturni ishlatib, qulaylik bo'yicha fikr bildiradi.

Topshiriqlar

1. Dasturni yuqoridagi imkoniyatlar bilan yarating.
2. Dasturni OYD (oyna sinfi) shaklida qayta yozing.
3. Dasturdan darsda foydalanish bo'yicha qisqa yo'riqnoma yozing.

Nazorat savollari

1. Global o'zgaruvchilarni GUI da ishlatish qanday muammo tug'diradi?
2. Dasturni sinfga o'tkazish nima beradi?
3. Interfeys qulayligi qanday baholanadi?
4. GUI dasturini o'quvchilarga topshiriq sifatida qanday berish mumkin?

Adabiyotlar: [1], [9]

18-LABORATORIYA ISHI. MA'LUMOTLARNI VIZUALLASHTIRISH: DIAGRAMMALAR QURISH

Maqsad: ma'lumotlarni vizuallashtirish: diagrammalar qurish ko'nikmasini shakllantirish.

Kutilayotgan natija (TN8): tinglovchi turli diagramma turlarini quradi, to'g'ri tanlaydi, sarlavha va o'qlarni to'g'ri belgilaydi.

Kerakli jihozlar: kompyuter, Python muhiti, matplotlib va pandas.

Ishni bajarish tartibi

1-qadam. Ma'lumot. O'ylab topilgan yoki ochiq ma'lumotlar to'plami tayyorlanadi (kamida 30 qator).

2-qadam. Ustunli va chiziqli diagrammalar. Toifalarni solishtirish va vaqt bo'yicha o'zgarishni ko'rsatuvchi diagrammalar quriladi.

```
import matplotlib.pyplot as plt
kunlar = ["Du", "Se", "Ch", "Pa", "Ju"]
harorat = [18, 21, 19, 24, 22]
plt.plot(kunlar, harorat, marker="o")
plt.title("Haftalik harorat")
plt.xlabel("Kun"); plt.ylabel("Harorat (°C)")
plt.grid(True)
plt.savefig("harorat.png", dpi=150)
```

3-qadam. Gistogramma va sohilish. Taqsimot va ikki o'zgaruvchi orasidagi bog'liqlik ko'rsatiladi.

4-qadam. Doiraviy diagramma. Ulushlar diagrammasi quriladi; qachon mos kelmasligi muhokama qilinadi.

5-qadam. Tahlil. Har bir diagramma uchun 2 jumalak xulosa yoziladi; chalg'ituvchi diagramma namunalari (kesilgan o'q) muhokama qilinadi.

Topshiriqlar

1. Kamida 4 xil diagramma quring.
2. Bitta ma'lumotni uch xil diagramma bilan ko'rsating va eng mosini tanlab asoslang.
3. Diagrammalar bo'yicha o'quvchilar uchun yo'riqnomalar tayyorlang.

Nazorat savollari

1. Qaysi diagramma turi qachon tanlanadi?
2. Diagrammada nimalar ko'rsatilishi shart?
3. Doiraviy diagramma qachon mos emas?
4. Diagramma qanday chalg'itishi mumkin?

Adabiyotlar: [8], [9]

19-LABORATORIYA ISHI. KODNI TESTLASH (UNITTEST) VA XATOLARNI TUZATISH (DEBUGGING)

Maqsad: kodni testlash (unittest va pytest) va xatolarni tuzatish (debugging) ko'nikmasini shakllantirish.

Kutilayotgan natija (TN10): tinglovchi funksiyalar uchun test yozadi, xatoli kodni debugger yordamida tuzatadi va testlar to'plamini yaratadi.

Kerakli jihozlar: kompyuter, Python muhiti, unittest (standart), ixtiyoriy pytest (pip install pytest).

Ishni bajarish tartibi

1-qadam. Xatoli dastur. O‘qituvchi bergan 3 ta xatoli funksiya ishga tushirilib, xatolar belgilanadi.

2-qadam. unittest. Funksiyalar uchun testlar yoziladi.

```
import unittest
def yuza(a, b):
    return a * b
class TestYuza(unittest.TestCase):
    def test_oddiy(self):
        self.assertEqual(yuza(3, 4), 12)
    def test_nol(self):
        self.assertEqual(yuza(0, 5), 0)
if __name__ == "__main__":
    unittest.main()
```

3-qadam. Debugging. Thonny yoki VS Code debugger bilan to‘xtash nuqtalari qo‘yilib, o‘zgaruvchilar kuzatiladi va xato topiladi.

4-qadam. Tuzatish. Xatolar tuzatilib, barcha testlar o‘tkaziladi.

5-qadam. Hisobot. Xato, uning sababi va tuzatish jadvalda yoziladi.

Topshiriqlar

1. Kamida 3 ta funksiya uchun har biriga 4 tadan test yozing.
2. O‘qituvchi bergan xatoli kodni tuzating va hisobot yozing.
3. “Rezina o‘rdak” usulini juftlikda sinab, natijani yozing.

Nazorat savollari

1. Birlik testi nima?
2. Test uchun qanday chekka holatlar tanlanadi?
3. Debugging bosqichlari qanday?
4. To‘xtash nuqtasi (breakpoint) nima?

Adabiyotlar: [9], [11]

20-LABORATORIYA ISHI. YAKUNIY LOYIHA: MAKTAB UCHUN FOYDALI DASTUR (TEST TIZIMI, DARS JADVALI GENERATORI VA H.K.) YARATISH, GITHUB GA JOYLASHTIRISH VA HIMOYA QILISH

Maqsad: o‘rganilgan bilimlar asosida maktab uchun foydali yakuniy loyiha yaratish, uni GitHub ga joylashtirish va himoya qilish ko‘nikmasini shakllantirish.

Kutilayotgan natija (TN10): tinglovchi loyihani rejalashtiradi, ishlab chiqadi, testlaydi, hujjatlaydi, GitHub ga joylaydi va taqdim etadi.

Kerakli jihozlar: kompyuter, Python muhiti, Git va GitHub, taqdimot dasturi.

Ishni bajarish tartibi

1-qadam. G‘oya va reja. Maktab uchun foydali dastur tanlanadi (test tizimi, dars jadvali generatori, baholar daftarchasi, lug‘at trenajyori va h.k.); talablar, foydalanuvchi va imkoniyatlar yoziladi.

2-qadam. Arxitektura. Modullar, sinflar va ma’lumotlar tuzilmasi belgilanadi.

3-qadam. Ishlab chiqish. Loyiha kichik bosqichlarda yoziladi; har bir bosqichdan keyin commit qilinadi.

4-qadam. Testlash va hujjatlash. Testlar yoziladi; README (maqsad, o‘rnatish, foydalanish, muallif) tayyorlanadi.

5-qadam. Joylash va himoya. Loyiha GitHub ga joylanadi; taqdimot tayyorlanib, himoya qilinadi.

Loyiha hujjatlari tarkibi:

№	Qism	Mazmuni
1	Loyiha tavsifi	Maqsad, foydalanuvchi, asosiy imkoniyatlar, cheklovlar
2	Arxitektura	Modullar, sinflar, ma’lumotlar tuzilmasi, fayl formatlari
3	Dastur kodi	Tuzilmali, nomlari mazmunli, docstring va izohlar bilan
4	Testlar	Kamida 10 ta test (unittest yoki pytest, yoki assert)
5	README.md	O‘rnatish, ishga tushirish, foydalanish misoli, muallif, litsenziya
6	GitHub repozitoriysi	Kamida 8 ta ma’noli commit, .gitignore, tarmoq (branch) tarixi
7	Taqdimot	7–10 slayd

Taqdimot tuzilmasi (7–10 slayd): (1) loyiha nomi va muallif; (2) muammo va maqsad; (3) foydalanuvchilar va talablar; (4) arxitektura; (5) asosiy kod namunasi; (6) testlash natijalari; (7) namoyish (demo); (8) qiyinchiliklar va xulosa; (9) kelgusi rivojlantirish rejasi.

Himoya savollari: Nima uchun aynan shu muammo tanlandi? Ma’lumotlar tuzilmasi qanday tanlandi? Dasturdagi eng murakkab qism qaysi va u qanday ishlaydi? Dastur qanday sinaldi? Sun’iy intellekt yordamchilari ishlatilgan bo‘lsa, qaysi qismda va qanday tekshirildi? Loyihani maktab sharoitida qanday qo‘llash mumkin?

Baholash rubrikasi:

Mezon	3 ball	2 ball	1 ball
Dastur ishlashi	Barcha imkoniyatlar ishlaydi	Asosiy qismi ishlaydi	Ishlamaydi
Kod sifati	Tuzilmali, o‘qilishi oson	Qisman tuzilmali	Tartibsiz
Testlash	Kamida 10 test, o‘tadi	Bir nechta test	Testlar yo‘q
Hujjatlash va GitHub	To‘liq README, commitlar ma’noli	Qisman	Yo‘q
Himoya	Asosli va aniq	Qisman asosli	Asoslay olmaydi

Topshiriqlar

1. Loyihani to‘liq yarating va GitHub ga joylang.
2. Taqdimot (7–10 slayd) tayyorlang.
3. Loyihaning maktab o‘qituvchisi ishidagi qo‘llanishi bo‘yicha qisqa yo‘riqnoma yozing.

Nazorat savollari

1. Loyiha qanday bosqichlardan iborat?
2. Nima uchun commitlar kichik va ma’noli bo‘lishi kerak?
3. README da nimalar bo‘ladi?
4. Loyihani qanday sinaysiz va takomillashtirasiz?

Adabiyotlar: [1], [11], [14]

VII. MUSTAQIL TA'LIM MASHG'ULOTLARI

Mustaqil ta'limni tashkil etish

Mustaqil ta'lim tinglovchining auditoriyadan tashqari mustaqil faoliyati bo'lib, fan bo'yicha 6 ta topshiriqni bajarishni o'z ichiga oladi. Topshiriqlar dasturlash ko'nikmalarini mustahkamlashga va ularni maktab amaliyotiga bog'lashga yo'naltirilgan: natija ishlaydigan dastur, masalalar to'plami, tahlil yoki loyiha ko'rinishida topshiriladi. Topshiriqlarning uchdan bir qismi (5- va 6-topshiriqlar) hamkorlikda, kichik guruhlarda bajariladi; qolganlari individual bajariladi.

Mustaqil ish topshirish tartibi

Ishlar elektron shaklda (fayl yoki GitHub havolasi) topshiriladi. Topshirish muddatlari ishchi o'quv dasturidagi jadvalga mos. Dasturlar ishga tushirib tekshiriladi, tinglovchi qisqa taqdimot yoki og'zaki himoya bilan natijasini asoslaydi va o'z kodining har bir qismini tushuntira oladi. Plagiat aniqlangan ish qayta bajarishga qaytariladi. Sun'iy intellekt vositalaridan yordamchi sifatida foydalanilgan bo'lsa, buni ish oxirida ochiq ko'rsatish (qaysi qismda, qanday) va natijani mustaqil tekshirib, tushunib topshirish talab etiladi.

Mustaqil ishlarni baholash mezon

Mezon	5	4	3	2	Maks. ball
Topshiriq talablariga muvofiqlik	5	4	3	0	5
Dastur to'g'riligi va sinovdan o'tganligi	5	4	3	0	5
Kod sifati (nomlar, tuzilma, izohlar)	5	4	3	0	5
Tuzilma va rasmiylashtirish	5	4	3	0	5
Himoya va savollarga javob	5	4	3	0	5

Jami ball 25 dan; 22–25 ball – “5”, 18–21 ball – “4”, 13–17 ball – “3”, 13 dan kam – “2” baho sifatida qabul qilinadi.

1-topshiriq. Dasturlash tillarining qiyosiy tahlili: taqdimot tayyorlash

Maqsad: dasturlash tillarini belgilari bo'yicha solishtirish va maktab uchun til tanlashni asoslash. **Bog'liq mavzu:** M1, M2, M16. **Adabiyotlar:** [1], [3].

Talablar

- ikki-uch tilni (Python, C++, Java, JavaScript va boshqalar) kamida 6 belgi bo'yicha solishtiruvchi jadval (tiplash, bajarilish usuli, sintaksis, tezlik, qo'llanish sohasi, o'rganish qulayligi);

- bir xil masala (masalan, ikki sonning yig'indisi va sonlar ro'yxatining eng kattasi) tanlangan tillarda kod bilan taqqoslanadi;
- maktabda birinchi va ikkinchi tilni tanlashga asoslangan tavsiya (1 bet);
- 10–12 slaydli taqdimot; manbalar ro'yxati.

Variantlar

Variant	Solishtiriladigan tillar
1	Python va C++
2	Python va Java
3	Python va JavaScript
4	C++ va Java
5	Python va Pascal (tarixiy ta'limiy til)
6	Python va Scratch (blokli til)
7	Python va Go
8	Python va Kotlin
9	C++ va Rust (asosiy tushunchalar darajasida)
10	Python, C++ va JavaScript (uch til)

2-topshiriq. Python bo'yicha darajalangan masalalar to'plamini tayyorlash

Maqsad: masalalarni qiyinlik darajasiga ko'ra tizimlashtirish va har biriga yechim hamda metodik izoh yozish. **Bog'liq mavzu:** M3–M9, A2–A18. **Adabiyotlar:** [2], [4].

Talablar

- kamida 20 ta masala, 4 ta daraja (A – oson, B – o'rta, C – qiyin, D – olimpiada uslubi) bo'yicha;
- har bir masalada: shart, kirish va chiqish namunasi, yechim (Python kodi), kalit g'oya va mumkin bo'lgan tipik xato;
- har bir masala uchun kamida 3 ta test (shu jumladan chekka holat);
- masalalar to'plamining dars yoki to'garakda qo'llanish rejasi (qisqa);
- barcha yechimlar ishga tushirib tekshirilgan bo'lishi shart.

Variantlar

Variant	Mavzu
1	Chiziqli algoritmlar va arifmetik amallar
2	Shartli operatorlar
3	for sikli
4	while sikli
5	Satrlar
6	Ro'yxatlar

Variant	Mavzu
7	Lug‘atlar va to‘plamlar
8	Funksiyalar
9	Rekursiya
10	Aralash mavzular (yakuniy takrorlash)

3-topshiriq. Saralash va qidirish algoritmlari bo‘yicha vizual o‘quv material yaratish

Maqsad: algoritmlarni o‘quvchilar uchun ko‘rgazmali tushuntiruvchi material tayyorlash. **Bog‘liq mavzu:** M14, A25, A26, L12. **Adabiyotlar:** [6], [12].

Talablar

- tanlangan algoritm bo‘yicha 6–8 bosqichli vizual tushuntirish (animatsiyali taqdimot, Turtle yoki Tkinter animatsiyasi, yoki chizmalar to‘plami);
- algoritmning psevdokodi va Python kodi;
- namunaviy kirish ma’lumotida har bir qadam natijasi (kuzatuv jadvali);
- murakkablikning qisqa tahlili va eksperiment natijasi (vaqt o‘lchash jadvali);
- materialni darsda qo‘llash bo‘yicha qisqa ko‘rsatma.

Variantlar

Variant	Algoritm
1	Pufakchali saralash
2	Tanlash orqali saralash
3	Qo‘shish orqali saralash
4	Chiziqli qidiruv
5	Ikkilik qidiruv
6	Birlashtirib saralash (mergesort)
7	Tez saralash (quicksort)
8	Hisoblab saralash
9	Eratosfen g‘alviri
10	Xanoy minorasi (rekursiya)

4-topshiriq. Onlayn masalalar tizimida kamida 20 ta masalani yechish va hisobot yozish

Maqsad: olimpiada tipidagi masalalarni mustaqil yechish va xatolarni tahlil qilish ko‘nikmasini shakllantirish. **Bog‘liq mavzu:** M14, M17, L14, L15. **Adabiyotlar:** [6], [13].

Talablar

- ochiq onlayn tizimdan (masalan, Codeforces yoki o‘quv tizimlari) kamida 20 ta masala tanlanadi va yechiladi (qiyinlik darajasi tizimda ko‘rsatilgan);
- hisobotda har bir masala bo‘yicha: nomi va havolasi, shart qisqacha, g‘oya, murakkablik, hukm, olingan saboq;
- kamida 5 ta masala C++ da yechiladi;
- qabul qilinmagan har bir urinish tahlil qilinadi (sabab va tuzatish);
- hisobotning umumiy xulosasi: qaysi mavzularda qiyinchilik bo‘ldi va nima uchun.

Variantlar

Variant	Mavzular yo‘nalishi
1	Arifmetika va shartlar
2	Sikllar
3	Massivlar
4	Satrlar
5	Saralash
6	Qidirish (ikkilik qidiruv)
7	Sonlar nazariyasi (bo‘luvchilar, tub sonlar)
8	Rekursiya va qaytish (backtracking)
9	Lug‘at va to‘plam
10	Aralash mavzular

5-topshiriq. OYD asosida kichik loyiha yaratish (kichik guruhlarda hamkorlikda)

Maqsad: OYD tamoyillarini real loyihada qo‘llash va jamoada ishlash ko‘nikmasini shakllantirish. ***Bog‘liq mavzu:*** M12, M13, A23, A24, L11. ***Adabiyotlar:*** [1], [5].

Talablar

- guruh 3–4 kishidan iborat; rollar: loyiha rahbari, dasturchi, test mutaxassisi, hujjatlovchi (rollar almashinib turadi);
- kamida 4 ta sinf, ularning orasida vorislik va kompozitsiya qo‘llanadi; sinf diagrammasi chiziladi;
- fayl orqali ma’lumot saqlash (CSV yoki JSON) va xatolarni qayta ishlash;
- kamida 8 ta test; README; guruh a’zolarining hissasi ko‘rsatilgan hisobot;
- har bir ishtirokchi loyihaning o‘zi yozgan qismini himoyada tushuntiradi.

Variantlar

Variant	Loyiha mavzusi
1	Maktab kutubxonasi tizimi
2	Sinf jurnali va baholar hisobi
3	Do'kon ombori va sotuvlar
4	O'yin: qahramonlar va dushmanlar (matnli)
5	Bank hisoblari modeli
6	Dars jadvali boshqaruvi
7	Vazifalar (to-do) ro'yxati menejeri
8	Viktorina tizimi (savollar bazasi)
9	Oddiy sport musobaqalari natijalari tizimi
10	Transport vositalari parki modeli

6-topshiriq. Ma'lumotlar tahlili mini-loyihasi va GitHub da joylash (kichik guruhlarda hamkorlikda)

Maqsad: real (yoki o'ylab topilgan) ma'lumotlar bo'yicha tahlil o'tkazish va natijani GitHub orqali taqdim etish. **Bog'liq mavzu:** M19, M20, A34, A35, L18. **Adabiyotlar:** [8], [11].

Talablar

- guruh 3–4 kishi; ma'lumotlar to'plami kamida 50 qator (ochiq ma'lumot yoki o'ylab topilgan, shaxsiy ma'lumotsiz);
- pandas bilan tozalash, filtrlash, guruhlash; kamida 3 ta diagramma (matplotlib);
- 1–2 betlik tahlil hisoboti: savol, usul, natija, xulosa va cheklovlar;
- loyiha GitHub repozitoriysida: kod, ma'lumot (yoki unga havola), README, kamida 8 ta ma'noli commit, ishtirokchilar hissasi commitlar tarixida ko'rinadi;
- shaxsiy ma'lumotlar, parollar va maxfiy kalitlar repozitoriyga joylanmaydi.

Variantlar

Variant	Mavzu
1	Maktab o'quvchilarining baholari (o'ylab topilgan ma'lumot)
2	Ob-havo ma'lumotlari tahlili
3	Maktab bufeti sotuvlari
4	Sport natijalari
5	Kunlik ekran vaqti so'rovnomasi (anonim)
6	Shahar havosi sifati ma'lumotlari
7	Kitob o'qish statistikasi
8	Kundalik transport harakati hisobi

Variant	Mavzu
9	Mahalliy ochiq ma'lumotlar (aholi, ta'lim va h.k.)
10	Dasturlash tillari mashhurligi (ochiq so'rovnoma natijalari asosida)

Referat va tahliliy ish mavzulari (qo'shimcha)

Tinglovchi yuqoridagi topshiriqlarga qo'shimcha ravishda quyidagi mavzulardan birida qisqa (6–8 bet) tahliliy ish tayyorlashi mumkin:

- Dasturlash tillari tarixi va ularning rivojlanish tendensiyalari.
- Python va C++ ni maktabda o'qitish: qiyosiy metodik tahlil.
- Python tilining ta'limdagi o'rni: afzalliklar va cheklovlar.
- Algoritmilar murakkabligini maktab o'quvchilariga tushuntirish usullari.
- Ob'yektga yo'naltirilgan dasturlashni maktab darajasida o'qitish muammolari.
- PRIMM va juft dasturlash metodlarining dasturlashni o'rganishdagi samarasi.
- Olimpiada tayyorgarligida o'quvchilarning tipik xatolari va ularni bartaraf etish.
- Maktabda Git va GitHub dan foydalanish: imkoniyat va xavfsizlik.
- Dasturlashni baholashda avtomatik tekshiruv tizimlari.
- Sun'iy intellekt yordamchilari davrida dasturlashni o'qitish va baholash.
- Ma'lumotlar tahlilini maktab informatikasiga kiritish.
- Dasturiy ta'minot sifati: testlash va kod uslubi.

VIII. KEYSLAR BANKI

Keys-stadi – real vaziyatni tahlil qilish orqali tinglovchining muammoni aniqlash, variantlarni solishtirish va asosli qaror qabul qilish ko‘nikmasini rivojlantiruvchi interfaol metod. Quydagi 15 ta keys fan mavzulari bilan bog‘liq. Har bir keysda vaziyat, topshiriqlar va yechim yo‘nalishi (yagona “to‘g‘ri javob” emas, balki tahlil uchun tayanch) beriladi. Keys kichik guruhda 20–30 daqiqada muhokama qilinadi; guruhlar yechimlarini taqdim etadi va umumiy xulosa chiqariladi.

1-keys. “Qaysi tilni tanlaymiz?” (M1)

Vaziyat: maktabda dasturlash to‘garagi ochilmoqda. Bir o‘qituvchi C++ ni, boshqasi Python ni birinchi til sifatida taklif qilmoqda; ota-onalar esa “eng mashhur” tilni talab qilmoqda.

Topshiriqlar

- Tanlov uchun mezonlar ro‘yxatini tuzing (yosh, tayyorgarlik, maqsad, resurslar).
- Har ikki variantning kuchli va zaif tomonlarini jadvalda solishtiring.
- Ota-onalarga tushuntirish uchun qisqa asoslangan tavsiya yozing.

Yechim yo‘nalishi

Maqsad (tafakkur yoki olimpiada) va o‘quvchilar yoshi hal qiluvchi: boshlovchilar uchun Python soddaligi va tezkor natija tufayli qulay; C++ keyingi bosqichda olimpiada va tizimli tushunchalar uchun. “Mashhurlik” yagona mezon emas; til emas, tafakkur o‘rgatilishi muhim.

2-keys. “Muhit o‘rnatilmayapti” (M2)

Vaziyat: darsning boshida 8 ta kompyuterda Python ishga tushmayapti: buyruq satrida “python” topilmadi, deb xabar chiqmoqda. Dars 45 daqiqa, o‘quvchilar kutib turibdi.

Topshiriqlar

- Muammoning ehtimoliy sabablarini (PATH, versiya, o‘rnatilmagan) aniqlang.
- Dars vaqtini yo‘qotmasdan zaxira rejani taklif qiling (onlayn muharrir, juftlikda ishlash).
- Kelgusida bunday holatning oldini olish bo‘yicha kabinet tayyorgarligi ro‘yxatini tuzing.

Yechim yo‘nalishi

Avval tez tekshiruv (python –version, PATH), imkon bo‘lsa o‘rnatilgan Thonny ni ishlatish; zaxira: onlayn muharrir yoki ishlaydigan kompyuterlarda

juftlikda ishlash. Dars boshlanishidan oldin muhitni sinash va o'rnatish yo'riqnomasini tayyorlash.

3-keys. "Hammasi ishlaydi, lekin natija noto'g'ri" (M3, M4)

Vaziyat: o'quvchi ikki sonning o'rtachasini hisoblovchi dastur yozdi: $a + b /$
2. Dastur xatosiz ishlaydi, lekin natija noto'g'ri. O'quvchi: "Python hisoblashni bilmaydi".

Topshiriqlar

- O'quvchi bilan suhbat uchun yo'naltiruvchi savollar tuzing (tayyor javobni aytmasdan).
- Operatorlar ustuvorligi tushunchasini tushuntirish uchun mini-mashq yozing.
- Test jadvali orqali xatoni topish usulini ko'rsating.

Yechim yo'nalishi

O'quvchi o'zi ifodani qo'lda hisoblab ko'rsin ($a = 4$, $b = 6$: natija 7, 5 emas); kuzatuv va ustuvorlik jadvali; qavslardan foydalanish. Mantiqiy xato dasturchining xatosi, mashinaniki emas.

4-keys. "Shart ishlamayapti" (M4)

Vaziyat: baholash dasturida o'quvchi shartlarni $\text{ball} \geq 56$, $\text{ball} \geq 71$, $\text{ball} \geq 86$ tartibida yozgan. 90 ball kiritilganda dastur "3" chiqarmoqda.

Topshiriqlar

- Xato sababini tahlil qiling.
- Shartlar tartibini to'g'ri qurish bo'yicha qoidani shakllantiring.
- Chekka holatlar test jadvalini tuzing.

Yechim yo'nalishi

elif zanjirida birinchi rost shart bajariladi; shartlar tor-kengdan emas, aniq-umumiya qarab tartiblanadi (yoki oraliq chegaralari ikki tomonlama yoziladi). Chegaraviy qiymatlar (55, 56, 70, 71, 85, 86) bilan sinash.

5-keys. "Dastur to'xtamayapti" (M5)

Vaziyat: while sikli bilan raqamlar yig'indisi hisoblanayotgan dasturda o'quvchi $n \neq 10$ qatorini unutgan. Dastur "qotib qoldi".

Topshiriqlar

- O'quvchiga cheksiz sikl sababini topishga yordam beruvchi savollar tuzing.
- Kuzatuv jadvali orqali xatoni ko'rsating.
- Dastur to'xtatilishi va qayta ishga tushirish tartibini (Ctrl+C, debugger) tushuntiring.

Yechim yo‘nalishi

Sikl o‘zgaruvchisi o‘zgarmasa, shart doim rost qoladi. Kuzatuv jadvali (n qiymati har qadamda) buni ko‘rsatadi. Debuggerda qadamma-qadam bajarish va sikl uchun to‘xtash sharti haqida o‘ylash odati.

6-keys. “Ro‘yxat o‘zgarib qoldi” (M7)

Vaziyat: o‘quvchi ro‘yxat nusxasini $b = a$ bilan olib, b ni o‘zgartirganda a ham o‘zgardi. “Dastur buzilgan”, deydi o‘quvchi.

Topshiriqlar

- Hodisani vizual tushuntirish usulini taklif qiling (Python Tutor, qog‘ozda xotira sxemasi).
- Nusxalash usullarini (copy, kesim, list()) solishtiring.
- Ichma-ich ro‘yxatlarda chuqur nusxa muammosini muhokama qiling.

Yechim yo‘nalishi

O‘zgaruvchi qiymat emas, ob‘yektga havola; $b = a$ bitta ob‘yektga ikkinchi nom beradi. Python Tutor xotira diagrammasi tushunchani aniq ko‘rsatadi; mustaqil nusxa uchun `a.copy()`; ichma-ichda `copy.deepcopy`.

7-keys. “Lug‘at yoki ro‘yxat?” (M8)

Vaziyat: o‘quvchi 1000 ta ismdan iborat ro‘yxatda har bir so‘rovda ismni qidirmoqda va dastur sekin ishlayapti.

Topshiriqlar

- Muammoning sababini tahlil qiling.
- Qaysi tuzilma mos kelishini asoslang (to‘plam yoki lug‘at).
- Tezlikni o‘lchaydigan kichik tajriba rejasini tuzing.

Yechim yo‘nalishi

Ro‘yxatda tekshiruv $O(n)$, to‘plam va lug‘atda o‘rtacha $O(1)$. Tegishlilikni tekshirish ko‘p bo‘lsa, set; kalit bo‘yicha qiymat kerak bo‘lsa, dict. time moduli bilan solishtirish ta’sirchan.

8-keys. “Funksiya hech narsa qaytarmayapti” (M9)

Vaziyat: o‘quvchi funksiya ichida natijani `print()` qilgan, so‘ng uni boshqa hisobda ishlatmoqchi, lekin `None` ni olmoqda.

Topshiriqlar

- `print` va `return` farqini tushuntiruvchi mashq tuzing.
- Funksiya “qora quti” ekanligini ko‘rsatadigan o‘yin yoki analogiyani taklif qiling.
- Funksiyalarni tushunishni tekshiruvchi 5 savollik mini-test tuzing.

Yechim yo‘nalishi

print ekranga chiqaradi, return qiymatni chaqirilgan joyga qaytaradi; qaytarmagan funksiya None beradi. “Avtomat” analogiyasi (tanga tashla – mahsulot ol). Mini-testda kod parchalarining natijasini taxmin qilish savollari.

9-keys. “Rekursiya tushunarsiz” (M9)

Vaziyat: 9-sinfda faktorial rekursiyasi o‘qitilgach, o‘quvchilarning ko‘pchiligi “funksiya o‘zini qanday chaqiradi va qanday to‘xtaydi” tushunmayapti.

Topshiriqlar

- Rekursiyani tushuntirishning kamida 3 ta usulini (analogiya, stek chizmasi, kuzatuv) taklif qiling.
- Bazaviy holatni topishga yo‘naltiruvchi mashqlar tuzing.
- Rekursiya va siklni solishtiruvchi topshiriq yozing.

Yechim yo‘nalishi

Matryoshka analogiyasi, chaqiruvlar steki chizmasi, bazaviy holatdan boshlash; Turtle bilan vizual rekursiya (spiral, daraxt); bir masalani rekursiya va sikl bilan yechib solishtirish.

10-keys. “Fayl topilmadi” (M11)

Vaziyat: o‘quvchining dasturi uyda ishlaydi, maktab kompyuterida FileNotFoundError beradi. O‘quvchi “kompyuter noto‘g‘ri” deydi.

Topshiriqlar

- Mumkin bo‘lgan sabablarni aniqlang (joriy papka, nisbiy yo‘l, fayl nomi).
- Xatoni dasturda chiroyli qayta ishlash usulini yozing.
- Fayl yo‘llari bo‘yicha qisqa qoidalar to‘plamini tuzing.

Yechim yo‘nalishi

Nisbiy yo‘l joriy ishchi papkaga bog‘liq; fayl nomi va joylashuvini tekshirish; try/except bilan foydalanuvchiga tushunarli xabar; pathlib yordamida yo‘lni aniq berish.

11-keys. “Bitta sinf juda katta bo‘lib ketdi” (M12, M13)

Vaziyat: guruh loyihasida bitta sinf 500 qatorga yetgan: unda ma’lumot, hisob, fayl va chiqarish aralash. Guruh a’zolari bir-biriga xalaqit bermoqda.

Topshiriqlar

- Muammoni “mas’uliyatlar” nuqtayi nazaridan tahlil qiling.
- Sinfni bir nechta kichik sinfga bo‘lish rejasini tuzing.
- Vorislik va kompozitsiyadan qaysi biri qayerda mos kelishini asoslang.

Yechim yo‘nalishi

Har bir sinf bitta vazifaga mas’ul bo‘lishi kerak; ma’lumot, mantiq va kiritish-chiqarishni ajratish; kompozitsiya (“has-a”) ko‘pincha moslashuvchanroq; refaktoringdan keyin testlar bilan xulq saqlanganini tekshirish.

12-keys. “Mening kodim sekin” (M14)

Vaziyat: o‘quvchi saralash dasturini yozgan: 10 000 elementda dastur bir necha soniya ishlaydi. Do‘sti sorted() bilan shu ishni bir lahzada bajaryapti.

Topshiriqlar

- Farq sababini murakkablik nuqtayi nazaridan tushuntiring.
- Eksperiment rejasini tuzing (turli n , vaqt o‘lchash).
- O‘quvchini tayyor vositalardan foydalanishga va algoritmni tushunishga o‘rgatish muvozanatini muhokama qiling.

Yechim yo‘nalishi

$O(n^2)$ va $O(n \log n)$ farqi n ortgani sari keskin kattalashadi; eksperiment buni ko‘rsatadi. O‘z algoritmini yozish tushunish uchun foydali, amaliyotda esa sinalgan tayyor vositalar ishlatiladi; ikkalasi ham o‘z o‘rnida.

13-keys. “Olimpiadada Time Limit” (M17)

Vaziyat: o‘quvchi masalani to‘g‘ri yechgan, namunaviy testlarda javob to‘g‘ri, lekin tizim “Time Limit Exceeded” hukmini bermoqda.

Topshiriqlar

- Mumkin bo‘lgan sabablarni aniqlang (algoritm murakkabligi, kiritish-chiqarish, sikl ichidagi qimmat amallar).
- Chegaralar asosida zarur murakkablikni baholash usulini tushuntiring.
- Tuzatish strategiyasini yozing.

Yechim yo‘nalishi

Chegaradan (masalan, $n = 10^5$) kelib chiqib, $O(n^2)$ ishlamasligini baholash; samaraliroq algoritm yoki tuzilma (lug‘at, saralash, prefiks yig‘indi); C++ da tezkor kiritish-chiqarish; katta test bilan o‘z tekshiruvi.

14-keys. “Tkinter da tugma ishlamayapti” (M18)

Vaziyat: o‘quvchi Tkinter oynasida tugma yaratgan, lekin tugma bosilmasdan darhol amal bajarilmoqda va keyin tugma hech narsa qilmayapti.

Topshiriqlar

- Xato sababini aniqlang.
- Hodisaga asoslangan dasturlash tushunchasini tushuntirish uchun mashq tuzing.

- Debuggingni osonlashtiruvchi usullarni taklif qiling.

Yechim yo‘nalishi

command=funksiya() yozilganda funksiya darhol chaqiriladi; to‘g‘risi command=funksiya (qavssiz). Hodisa “kutish” modeli haqida analogiya (qo‘ng‘iroq tugmasi). Tekshirish uchun print() yoki kichik funksiya bilan sinash.

15-keys. “Tayyor kodni SI berdi” (M20)

Vaziyat: o‘quvchi loyihani GitHub ga joylagan; kod yuqori sifatli, lekin o‘quvchi kodning bir qismini tushuntira olmayapti va “sun’iy intellekt yozdi” demoqda. Bir qismida esa begona maxfiy kalit ham qolib ketgan.

Topshiriqlar

- Pedagogik jihatdan o‘qituvchining harakatini rejalashtiring.
- SI vositalaridan foydalanish bo‘yicha sinf qoidalarini yozing.
- Maxfiy ma’lumotlarning repozitoriyga tushib qolishining oldini olish bo‘yicha tavsiya tuzing.

Yechim yo‘nalishi

Jazolashdan oldin tushuntirish va o‘rganishga yo‘naltirish: o‘quvchi koddan bir qismini sharhlab beradi yoki o‘xshash yangi topshiriq bajaradi; qoidalar (qachon ruxsat, qanday ko‘rsatiladi, tekshirish majburiy); kalitni darhol bekor qilish (almashtirish), .gitignore va muhit o‘zgaruvchilari.

IX. GLOSSARIY

Glossariyda fanning asosiy atamaları o‘zbek, ingliz va rus tillarida izohi bilan keltirilgan. Atamalar o‘zbek alifbosi tartibida joylashtirilgan. Glossariy tinglovchilarga fan tushunchalarini aniq o‘zlashtirish, mavzu bo‘yicha matn tayyorlash va xalqaro manbalar bilan ishlashda yordam beradi.

№	O‘zbekcha	Inglizcha	Ruscha	Izoh
1	Akademik halollik	Academic integrity	Академическая честность	Ishni mustaqil bajarish, manbalarni ochiq ko‘rsatish va boshqalar mehnatini o‘zlashtirmaslik qoidalari
2	Akkumulyator	Accumulator	Аккумулятор	Sikl davomida natija yig‘iladigan o‘zgaruvchi (yig‘indi, ko‘paytma)
3	Algoritm	Algorithm	Алгоритм	Masalani yechish uchun bajariladigan amallarning aniq, chekli va tushunarli ketma-ketligi
4	Algoritmilar murakkabligi	Algorithm complexity	Сложность алгоритма	Algoritm bajaradigan amallar soni yoki ishlatadigan xotiraning kirish hajmiga bog‘liq o‘sishi
5	Argument	Argument	Аргумент	Funksiya chaqirilganda unga uzatiladigan aniq qiymat
6	Assembler	Assembly language	Ассемблер	Mashina buyruqlari qisqa nomlar bilan yoziladigan, protsessorga bog‘liq quyi darajali til
7	Atribut	Attribute	Атрибут	Ob‘yektga yoki sinfga tegishli ma’lumot (o‘zgaruvchi)
8	Bayt-kod	Bytecode	Байт-код	Manba kodidan hosil qilingan, virtual mashina bajaradigan oraliq kod
9	Bazaviy holat	Base case	Базовый случай	Rekursiyani to‘xtatuvchi, javobi bevosita ma’lum bo‘lgan holat
10	Birlik testi	Unit test	Модульный тест	Dasturning kichik qismi (funksiya) to‘g‘riligini tekshiruvchi kod
11	Bog‘langan ro‘yxat	Linked list	Связный список	Har bir tuguni qiymat va keyingi tugunga havolani saqlovchi chiziqli tuzilma
12	Bo‘lib tashla va hukmronlik qil	Divide and conquer	Разделяй и властвуй	Masalani kichik qismlarga bo‘lib yechib, natijalarni birlashtirish tamoyili
13	Branch (tarmoq)	Branch	Ветка	Git da asosiy tarixdan ajralib, parallel rivojlanadigan ishlanma chizig‘i

№	O'zbekcha	Inglizcha	Ruscha	Izoh
14	Daraxt	Tree	Дерево	Ildiz, tugunlar va barglardan iborat ierarxik ma'lumotlar tuzilmasi
15	DataFrame	DataFrame	DataFrame	pandas dagi ikki o'lchovli, nomlangan ustunli jadval ob'yekti
16	Debugging (xatolarni tuzatish)	Debugging	Отладка	Dasturdagi xatolarni topish va tuzatish jarayoni
17	Dekompozitsiya	Decomposition	Декомпозиция	Murakkab masalani kichik, oson yechiladigan qismlarga bo'lish
18	Dinamik tiplash	Dynamic typing	Динамическая типизация	O'zgaruvchi turi dastur bajarilishi paytida qiymatga qarab aniqlanishi
19	Docstring	Docstring	Строка документации	Funksiya, sinf yoki modul vazifasini tushuntiruvchi qo'shtirnoqdagi satr
20	Elementlar chastotasi	Frequency	Частота	Elementning ma'lumotlar to'plamida necha marta uchrashishi
21	Fayl	File	Файл	Doimiy xotirada nom bilan saqlanadigan ma'lumotlar majmuasi
22	Fraktal	Fractal	Фрактал	O'z qismlariga o'xshash, rekursiv qoida bilan hosil qilinadigan shakl
23	Funksiya	Function	Функция	Nom berilgan, qayta ishlatiladigan va qiymat qaytarishi mumkin bo'lgan kod bloki
24	Git	Git	Git	Fayllar o'zgarishlar tarixini boshqaruvchi versiyalarni boshqarish tizimi
25	GitHub	GitHub	GitHub	Git repozitoriylarni onlayn saqlash va hamkorlikda ishlash platformasi
26	Global o'zgaruvchi	Global variable	Глобальная переменная	Funksiyalardan tashqarida yaratilgan va butun dasturda ko'rindigan o'zgaruvchi
27	Graf	Graph	Граф	Uchlar va ularni bog'lovchi qirralardan iborat tuzilma
28	Hisoblagich	Counter	Счётчик	Shart bajarilishlari sonini sanovchi o'zgaruvchi
29	Hodisa	Event	Событие	Foydalanuvchi yoki tizim harakati (tugma bosish, sichqoncha harakati)

№	O'zbekcha	Inglizcha	Ruscha	Izoh
30	Ibtidoiy tur	Primitive type	Примитивный тип	Tilning asosiy ma'lumot turi (butun son, haqiqiy son, mantiqiy qiymat)
31	IDE	IDE (integrated development environment)	Среда разработки	Muharrir, ishga tushirish va xatolarni topish vositalarini birlashtirgan dasturlash muhiti
32	Ikkilik qidiruv	Binary search	Двоичный поиск	Tartiblangan ma'lumotda qidirish oraliq'ini har safar yarmiga qisqartiruvchi usul
33	Indeks	Index	Индекс	Ketma-ketlikdagi elementning tartib raqami (0 dan boshlanadi)
34	Inkapsulyatsiya	Encapsulation	Инкапсуляция	Ma'lumot va amallarni birlashtirib, ichki tuzilishga tashqaridan murojaatni cheklash
35	Interaktiv rejim (REPL)	Interactive mode (REPL)	Интерактивный режим	Har bir buyruq kiritilishi bilan darhol bajariladigan rejim
36	Interpretator	Interpreter	Интерпретатор	Dasturni qator-qator o'qib, darhol bajaruvchi dastur
37	Istisno	Exception	Исключение	Dastur bajarilishi paytida yuz bergan va qayta ishlanishi mumkin bo'lgan xato holati
38	Iteratsiya	Iteration	Итерация	Siklning bitta bajarilishi (qadami)
39	Jupyter Notebook	Jupyter Notebook	Jupyter Notebook	Kod, matn va natijalarni "katak"lar shaklida birlashtiruvchi interaktiv hujjat muhiti
40	Kalit	Key	Ключ	Lug'atda qiymatni topish uchun ishlatiladigan noyob identifikator
41	Kenglik bo'yicha qidiruv (BFS)	Breadth-first search	Поиск в ширину	Grafda qatlam-qatlam yuruvchi, navbatdan foydalanadigan usul
42	Kesim	Slice	Срез	Ketma-ketlikning bir qismini ajratib olish: s[a:b:q]
43	Kod sharhi	Code review	Код-ревью	Hamkasb tomonidan kodni o'qib, tavsiyalar berish jarayoni
44	Kompilyator	Compiler	Компилятор	Dastur matnini butunligicha mashina kodiga tarjima qiluvchi dastur
45	Kompozitsiya	Composition	Композиция	Ob'yektning boshqa ob'yektlarni o'z ichiga olishi ("has-a" munosabati)

№	O'zbekcha	Inglizcha	Ruscha	Izoh
46	Konstruktor	Constructor	Конструктор	Ob'yekt yaratilganda chaqiriladigan va boshlang'ich qiymat beruvchi metod (__init__)
47	Kortej	Tuple	Кортеж	O'zgarmas tartiblangan elementlar ketma-ketligi
48	Kuzatuv jadvali	Trace table	Таблица трассировки	Dastur bajarilishida o'zgaruvchilar qiymatlarini qadam-baqadam yozib borish jadvali
49	Lambda funksiya	Lambda function	Лямбда-функция	Nomsiz, bir ifodali qisqa funksiya
50	Lokal o'zgaruvchi	Local variable	Локальная переменная	Funksiya ichida yaratilgan va faqat shu yerda ko'rinadigan o'zgaruvchi
51	Lug'at	Dictionary	Словарь	Kalit va qiymat juftliklaridan iborat ma'lumotlar tuzilmasi
52	Mantiqiy ifoda	Boolean expression	Логическое выражение	Natijasi rost yoki yolg'on bo'lgan ifoda
53	Massiv	Array	Массив	Bir xil turdagi elementlarning ketma-ket joylashgan tartiblangan to'plami
54	Mashina kodi	Machine code	Машинный код	Protessor to'g'ridan-to'g'ri bajaradigan ikkilik buyruqlar
55	Ma'lumotlar tuzilmasi	Data structure	Структура данных	Ma'lumotlarni saqlash va ular bilan ishlash usuli
56	Metod	Method	Метод	Sinf ob'yektiga tegishli funksiya
57	Modul	Module	Модуль	Boshqa dasturlarda qayta ishlatish mumkin bo'lgan Python fayli
58	Navbat	Queue	Очередь	Birinchi kirgan birinchi chiqadigan (FIFO) ma'lumotlar tuzilmasi
59	NumPy	NumPy	NumPy	Sonli massivlar va vektorlashtirilgan hisoblar uchun Python kutubxonasi
60	Ob'yekt	Object	Объект	Sinfning aniq nusxasi: atributlar va metodlarga ega bo'lgan birlik
61	Operator	Operator	Оператор	Qiymatlar ustida amal bajaruvchi belgi yoki kalit so'z
62	Ochiq ma'lumotlar	Open data	Открытые данные	Hamma erkin foydalanishi mumkin bo'lgan, ochiq e'lon qilingan ma'lumotlar to'plami
63	OYD	OOP (object-oriented programming)	ООП	Dasturni o'zaro ta'sirlashuvchi ob'ektlar sifatida tuzish paradigmasi

№	O‘zbekcha	Inglizcha	Ruscha	Izoh
64	O-notatsiya	Big O notation	О-нотация	Algoritm samaradorligining kirish hajmi ortishi bilan o‘shish tartibini ifodalash usuli
65	Paket	Package	Пакет	Bir nechta modul joylashgan papka yoki tarqatiladigan kutubxona
66	pandas	pandas	pandas	Jadval ma’lumotlarini tahlil qilish uchun Python kutubxonasi
67	Paradigma	Paradigm	Парадигма	Dastur yozish uslubi va fikrlash modeli (protsedurali, OYD, funksional)
68	Parametr	Parameter	Параметр	Funksiya ta’rifida ko‘rsatilgan, qiymat qabul qiluvchi nom
69	PEP 8	PEP 8	PEP 8	Python kodini yozish uslubi bo‘yicha qo‘llanma
70	pip	pip	pip	Python paketlarini o‘rnatish uchun paket menejeri
71	Polimorfizm	Polymorphism	Полиморфизм	Bir xil nomli metodning turli ob’ektlarda turlicha bajarilishi
72	Protsedurali dasturlash	Procedural programming	Процедурное программирование	Dastur funksiyalar (protseduralar) ketma-ketligi sifatida yoziladigan paradigma
73	Pull request	Pull request	Запрос на слияние	O‘zgarishni asosiy tarmoqqa qo‘shishni taklif qilish va ko‘rib chiqish mexanizmi
74	Qo‘shnilik ro‘yxati	Adjacency list	Список смежности	Grafni har bir uchning qo‘shnilari ro‘yxati bilan ifodalash usuli
75	Refaktoring	Refactoring	Рефакторинг	Kod xulqini o‘zgartirmasdan uning tuzilmasini yaxshilash
76	Rekursiya	Recursion	Рекурсия	Funksiyaning o‘zini o‘zi chaqirishi
77	Repozitoriy	Repository	Репозиторий	Loyiha fayllari va ularning o‘zgarishlar tarixi saqlanadigan ombor
78	Ro‘yxat	List	Список	O‘zgaruvchan tartiblangan elementlar ketma-ketligi
79	Saralash	Sorting	Сортировка	Elementlarni berilgan tartibda (o‘shish yoki kamayish) joylashtirish
80	Satr	String	Строка	Belgilar ketma-ketligi
81	Semantika	Semantics	Семантика	Til konstruksiyalarining ma’nosi, ya’ni nimani bajarishi
82	Sikl	Loop	Цикл	Amallar ketma-ketligini bir necha marta takrorlash tuzilmasi

№	O'zbekcha	Inglizcha	Ruscha	Izoh
83	Sinf	Class	Класс	Ob'yektlar yaratish uchun andoza (qolip)
84	Sintaksis	Syntax	Синтаксис	Dasturlash tilida yozuv qoidalari majmuasi
85	Statik tiplash	Static typing	Статическая типизация	O'zgaruvchi turi dastur yozilishida e'lon qilinib, o'zgarmasligi
86	Stek	Stack	Стек	Oxirgi kirgan birinchi chiqadigan (LIFO) ma'lumotlar tuzilmasi
87	STL	STL (standard template library)	STL	C++ ning tayyor konteynerlar va algoritmlar kutubxonasi
88	Chekka holat	Edge case	Граничный случай	Chegaraviy yoki kam uchraydigan kirish qiymatlari (nol, bo'sh, eng katta) bo'yicha holat
89	Chiziqli qidiruv	Linear search	Линейный поиск	Elementlarni birin-ketin ko'rib chiqib qidirish usuli
90	Chuqurlik bo'yicha qidiruv (DFS)	Depth-first search	Поиск в глубину	Grafda bir yo'nalishda oxirigacha borib, so'ng qaytib yuruvchi usul
91	Tiplash	Typing	Типизация	Til o'zgaruvchilar turlarini qanday boshqarishi tizimi
92	Tkinter	Tkinter	Tkinter	Python ning standart grafik interfeys kutubxonasi
93	To'plam	Set	Множество	Takrorlanmaydigan elementlarning tartibsiz majmuasi
94	Traceback	Traceback	Трассировка стека	Xato yuz berganda Python chiqaradigan xato joyi va turi haqidagi xabar
95	Turtle	Turtle	Turtle	"Toshbaqa" yordamida chizish orqali dasturlash o'rgatuvchi kutubxona
96	Vektorlashtirish	Vectorization	Векторизация	Amalni sikl yozmasdan butun massivga bir vaqtda qo'llash
97	Versiyalarni boshqarish	Version control	Контроль версий	Fayllar o'zgarishlarini tarixi bilan saqlash va boshqarish tizimi
98	Vidjet	Widget	Виджет	Grafik interfeys elementi (tugma, matn maydoni, yozuv)
99	Virtual mashina	Virtual machine	Виртуальная машина	Bayt-kodni bajaruvchi dastur (masalan, Python VM, JVM)
100	Virtual muhit	Virtual environment	Виртуальное окружение	Loyiha uchun alohida o'rnatilgan paketlar to'plami (venv)

№	O‘zbekcha	Inglizcha	Ruscha	Izoh
10 1	Vizuallashtirish	Visualization	Визуализация	Ma’lumotlarni diagramma va grafiklar ko‘rinishida tasvirlash
10 2	Vorislik	Inheritance	Наследование	Hosila sinfning asosiy sinf xususiyatlarini olishi
10 3	Commit	Commit	Коммит	Git da o‘zgarishlar to‘plamini tarixga saqlash amali
10 4	CSV	CSV (comma-separated values)	CSV	Jadval ma’lumotlarini matn ko‘rinishida, ustunlar vergul bilan ajratilgan holda saqlash formati

X. TEST SAVOLLARI

Test savollari tinglovchilarning nazariy bilimlarini joriy nazorat va o‘z-o‘zini tekshirish maqsadida baholash uchun mo‘ljallangan. Jami 200 ta savol (har bir ma‘ruza mavzusi bo‘yicha 10 tadan) berilgan; har bir savolda 4 ta javob varianti va bitta to‘g‘ri javob mavjud. To‘g‘ri javoblar kaliti bo‘lim oxirida keltirilgan. Savollar o‘quv natijalariga mos ravishda eslash, tushunish va qo‘llash darajalarini qamrab oladi; noto‘g‘ri variantlar (distraktorlar) o‘quvchilarning tipik xatolariga asoslangan.

M1. Dasturlash tillari tasnifi, kompilyatsiya va interpretatsiya

1. Dasturlash tilining sintaksisi nimani belgilaydi?
 - A) Dasturning bajarilish tezligini
 - B) Dastur ishlatadigan xotira hajmini
 - C) Dastur muallifining ismini
 - D) Dastur yozish qoidalarini (yozuv shaklini)
2. Dasturlash tilining semantikasi nima?
 - A) Buyruqlarning yozilish tartibi
 - B) Kompilyator o‘rnatiladigan joy
 - C) Til konstruksiyalarining ma‘nosi, ya‘ni ular nimani bajarishi
 - D) Dasturdagi izohlar soni
3. Mashina kodi qanday ko‘rinishga ega?
 - A) Inson tilidagi gaplar
 - B) Protsessor to‘g‘ridan-to‘g‘ri bajaradigan buyruqlar ketma-ketligi (ikkilik kod)
 - C) Faqat grafik belgilar
 - D) Veb-sahifa kodi
4. Qaysi til yuqori darajali til hisoblanadi?
 - A) Mashina kodi
 - B) Python
 - C) Assembler
 - D) Ikkilik buyruqlar to‘plami
5. Kompilyator nimani bajaradi?
 - A) Butun dasturni bajarishdan oldin mashina kodiga tarjima qiladi
 - B) Dasturni qator-qator bajarib boradi
 - C) Faqat dastur matnini formatlaydi
 - D) Dasturni Internetga yuklaydi
6. Interpretatorning xususiyati qaysi qatorda to‘g‘ri ko‘rsatilgan?
 - A) Hech qachon xato topmaydi

- B) Faqat kompilyator bilan birga ishlaydi
 - C) Dasturni faqat bir marta, ish boshlanishidan oldin tarjima qiladi
 - D) Dasturni qator-qator o'qib, darhol bajaradi
7. Python dasturi odatda qanday bajariladi?
- A) Faqat to'g'ridan-to'g'ri assemblerga o'giriladi
 - B) Faqat qo'lda tarjima qilinadi
 - C) Avval bayt-kodga o'giriladi, so'ng virtual mashina bajaradi
 - D) Faqat brauzerda bajariladi
8. Statik tiplash nimani anglatadi?
- A) O'zgaruvchi turi har safar tasodifiy tanlanadi
 - B) O'zgaruvchilar umuman turga ega emas
 - C) O'zgaruvchi turi dastur yozilishida aniqlanadi va o'zgarmaydi
 - D) Dastur faqat butun sonlar bilan ishlaydi
9. Quyidagi tillardan qaysi biri dinamik tiplashga ega?
- A) C++
 - B) Python
 - C) Java
 - D) C
10. Qaysi til asosan veb-brauzerlarda ishlovchi dasturlar uchun qo'llanadi?
- A) JavaScript
 - B) Assembler
 - C) Fortran
 - D) Pascal

M2. Python muhiti: o'rnatish, IDE va Jupyter

11. Python o'rnatishda Windows da qaysi belgini qo'yish tavsiya etiladi?
- A) "Delete all files"
 - B) "Disable Python"
 - C) "Hide interpreter"
 - D) "Add Python to PATH"
12. Python versiyasini buyruq satridan tekshirish uchun qaysi buyruq ishlatiladi?
- A) python –delete
 - B) run python –hide
 - C) python –format
 - D) python –version
13. IDE nima?

- A) Faqat kompilyator
 - B) Faqat grafik muharrir
 - C) Dasturlash uchun muharrir, ishga tushirish va xatolarni topish vositalarini birlashtirgan muhit
 - D) Faqat antivirus dasturi
14. Quyidagilardan qaysi biri boshlovchilar uchun mo'ljallangan oddiy Python IDE si?
- A) Photoshop
 - B) Thonny
 - C) Excel
 - D) Paint
15. Jupyter Notebook ning asosiy xususiyati nima?
- A) Kod, matn va natijalarni bitta hujjatda "katak"lar bilan birlashtiradi
 - B) Faqat C++ kodini bajaradi
 - C) Dasturni mashina kodiga kompilyatsiya qiladi
 - D) Faqat rasmlarni tahrirlaydi
16. Python interaktiv rejimida (REPL) har bir buyruq qanday bajariladi?
- A) Kiritilgandan so'ng darhol bajariladi va natija ko'rsatiladi
 - B) Faqat dastur oxirida bajariladi
 - C) Hech qachon natija ko'rsatmaydi
 - D) Faqat faylga yoziladi
17. Python dasturi fayli odatda qanday kengaytma bilan saqlanadi?
- A) .cpp
 - B) .exe
 - C) .doc
 - D) .py
18. Xato xabari (traceback) da odatda nima ko'rsatiladi?
- A) Faqat dasturchi nomi
 - B) Faqat kompyuter modeli
 - C) Xato turi, fayl va qator raqami
 - D) Faqat sana
19. print("Salom" qatorida qavs yopilmagan. Python qanday xato beradi?
- A) ZeroDivisionError
 - B) SyntaxError
 - C) KeyError
 - D) FileNotFoundError
20. Maktab uchun dasturlash muhitini tanlashda nima hisobga olinadi?

- A) Faqat dasturning narxi yuqori bo'lishi
- B) Kompyuter resurslari, o'rnatish osonligi va o'quvchi uyda ham foydalana olishi
- C) Faqat rangli interfeys
- D) Faqat eng og'ir muhit

M3. O'zgaruvchilar, turlar, operatorlar, kiritish-chiqarish

21. Python da quyidagi nomlardan qaysi biri o'zgaruvchi nomi sifatida to'g'ri?
- A) talaba_yoshi
 - B) 2talaba
 - C) talaba yoshi
 - D) class
22. `type("5")` natijasi qanday?
- A) int
 - B) float
 - C) bool
 - D) str
23. `17 // 5` ifodasining natijasi nechaga teng?
- A) 3.4
 - B) 2
 - C) 3
 - D) 12
24. `17 % 5` ifodasining natijasi nechaga teng?
- A) 3
 - B) 3.4
 - C) 2
 - D) 85
25. `2 ** 10` ifodasining natijasi qancha?
- A) 20
 - B) 1024
 - C) 100
 - D) 12
26. `input()` funksiyasi qanday turdagi qiymat qaytaradi?
- A) str (satr)
 - B) int
 - C) float

- D) bool
27. `int(7.9)` natijasi nechaga teng?
- A) 8
 - B) 7.9
 - C) Xato beradi
 - D) 7
28. `"5" + "3"` ifodasining natijasi qanday?
- A) 8
 - B) `"8"`
 - C) Xato beradi
 - D) `"53"`
29. Quyidagilardan qaysi biri qiymat berish operatori?
- A) `==`
 - B) `!=`
 - C) `=`
 - D) `<=`
30. `0.1 + 0.2` ifodasi nima uchun aniq 0.3 chiqmaydi?
- A) Python qo'shishni bilmaydi
 - B) Haqiqiy sonlar kompyuterda taxminiy saqlanadi
 - C) Bu ifoda mantiqiy xato
 - D) Sonlar satr sifatida qo'shiladi

M4. Shartli operatorlar: if, elif, else, match

31. Python da blok chegarasi nima bilan belgilanadi?
- A) Chekinish (indentatsiya) bilan
 - B) Figurali qavslar bilan
 - C) `begin` va `end` bilan
 - D) Nuqtali vergul bilan
32. Quyidagi shartlardan qaysi biri `x` ning 0 va 10 orasida ekanligini to'g'ri tekshiradi?
- A) `0 < x < 10`
 - B) `0 < x or 10`
 - C) `x == 0 to 10`
 - D) `x in 0 and 10`
33. `x == 1 or 2` yozuvi nima uchun noto'g'ri?
- A) `or` operatori Python da yo'q
 - B) `x` ni 1 bilan taqqoslab bo'lmaydi

- C) Bu yozuv sintaksis xatosi beradi
D) 2 har doim rost qiymat deb hisoblanadi, shart har doim rost bo'ladi
34. Quyidagilardan qaysi biri Python da "yolg'on" (falsy) hisoblanadi?
A) 1
B) "False" (bo'sh bo'lmagan satr)
C) 0
D) [0]
35. elif zanjirida shartlar qanday tekshiriladi?
A) Pastdan yuqoriga
B) Yuqoridan pastga; birinchi rost shartning bloki bajariladi
C) Barcha shartlar doim tekshirilib, barcha roslari bajariladi
D) Tasodifiy tartibda
36. ball >= 56 sharti ball >= 86 shartidan oldin yozilsa, 90 ball uchun nima bo'ladi?
A) Dastur to'xtaydi
B) Birinchi (kengroq) shart rost bo'lib, noto'g'ri baho beriladi
C) Eng yuqori baho beriladi
D) Hech narsa chiqmaydi
37. match operatori Python ning qaysi versiyasidan boshlab mavjud?
A) 3.10
B) 2.7
C) 3.2
D) 3.6
38. True and False ifodasining natijasi qanday?
A) True
B) None
C) Xato
D) False
39. not (3 > 5) ifodasi natijasi qanday?
A) False
B) 3
C) True
D) None
40. Chekka holatlarni tekshirishning maqsadi nima?
A) Dastur tezligini oshirish
B) Dasturni qisqartirish
C) Chegaraviy qiymatlarda dasturning to'g'ri ishlashini aniqlash

D) Izohlar sonini ko'paytirish

M5. Takrorlash operatorlari: for, while, break, continue

41. range(1, 6) qanday ketma-ketlikni beradi?

- A) 1, 2, 3, 4, 5, 6
- B) 1, 2, 3, 4, 5
- C) 0, 1, 2, 3, 4, 5
- D) 2, 3, 4, 5, 6

42. range(10, 0, -2) qanday ketma-ketlikni beradi?

- A) 10, 8, 6, 4, 2
- B) 10, 8, 6, 4, 2, 0
- C) 0, 2, 4, 6, 8, 10
- D) 10, 9, 8, 7, 6

43. Akkumulyatorga yig'indi hisoblashda qanday boshlang'ich qiymat beriladi?

- A) 1
- B) -1
- C) 10
- D) 0

44. Ko'paytma hisoblashda akkumulyator boshlang'ich qiymati qanday bo'lishi kerak?

- A) 0
- B) -1
- C) 100
- D) 1

45. Takrorlar soni oldindan noma'lum bo'lib, shartga bog'liq bo'lsa qaysi sikl qulay?

- A) for faqat
- B) if
- C) while
- D) match

46. Cheksiz sikl qachon paydo bo'ladi?

- A) for sikli range bilan yozilsa
- B) while shartida sikl o'zgaruvchisi o'zgarmay, shart doim rost qolsa
- C) break ishlatilsa
- D) Sikl tanasi bo'sh bo'lsa

47. break operatori nima qiladi?

- A) Siklni darhol to'xtatadi
 - B) Joriy iteratsiyani o'tkazib yuboradi
 - C) Dasturni o'chiradi
 - D) Siklni boshidan boshlaydi
48. continue operatori nima qiladi?
- A) Joriy iteratsiyaning qolgan qismini o'tkazib, keyingisiga o'tadi
 - B) Siklni butunlay to'xtatadi
 - C) Dasturni tugatadi
 - D) Hisoblagichni nolga tushiradi
49. $n = 345$ uchun $\text{while } n > 0: n //= 10$ sikli necha marta bajariladi?
- A) 2
 - B) 4
 - C) 345
 - D) 3
50. Kuzatuv jadvali (trace) nima uchun ishlatiladi?
- A) Dastur nusxasini saqlash uchun
 - B) Kompilyatorni sozlash uchun
 - C) Dastur bajarilishida o'zgaruvchilar qiymatlarini qadam-baqadam kuzatish uchun
 - D) Rasmlarni tahrirlash uchun

M6. Satrlar va satr metodlari, f-string

51. "Python"[0] ifodasi natijasi qanday?
- A) "y"
 - B) "P"
 - C) "n"
 - D) Xato
52. "Python"[-1] ifodasi natijasi qanday?
- A) "P"
 - B) "n"
 - C) "o"
 - D) Xato
53. "Informatika"[:4] ifodasi natijasi qanday?
- A) "Info"
 - B) "Inf"
 - C) "Infor"
 - D) "orma"

54. "salom"[::-1] ifodasi natijasi qanday?
- A) "salom"
 - B) "mosal"
 - C) "olsam"
 - D) "molas"
55. Satr s = "abc" uchun s[0] = "x" yozuvi nima beradi?
- A) s = "xbc" bo'ladi
 - B) s = "xabc" bo'ladi
 - C) Xato, chunki satrlar o'zgarmas
 - D) Hech narsa o'zgarmaydi, xatosiz
56. "a b c".split() natijasi qanday?
- A) "abc"
 - B) ['abc']
 - C) ['a', 'b', 'c']
 - D) ('a', 'b', 'c')
57. "-".join(["1", "2", "3"]) natijasi qanday?
- A) "123"
 - B) "1-2-3"
 - C) ["1-2-3"]
 - D) "1 2 3"
58. "banan".count("a") natijasi nechaga teng?
- A) 2
 - B) 3
 - C) 1
 - D) 5
59. f"{3.14159:.2f}" natijasi qanday?
- A) "3.1"
 - B) "3.142"
 - C) "3.14159"
 - D) "3.14"
60. ord("A") funksiyasi nimani qaytaradi?
- A) Belgining kichik harfini
 - B) Belgi uzunligini
 - C) Belgidan keyingi harfni
 - D) "A" belgisining Unicode kodini (65)

M7. Ro'yxatlar va kortejlar

61. `a = [5, 2, 9]`; `a.append(1)` dan so'ng `a` qanday bo'ladi?
- A) `[1, 5, 2, 9]`
 - B) `[5, 2, 1]`
 - C) `[5, 2, 9, 1]`
 - D) `[6, 3, 10]`
62. `sort()` va `sorted()` ning asosiy farqi nima?
- A) Ular butunlay bir xil
 - B) `sort()` ro'yxatni joyida o'zgartiradi, `sorted()` yangi ro'yxat qaytaradi
 - C) `sorted()` ro'yxatni o'zgartiradi, `sort()` yangisini qaytaradi
 - D) `sort()` faqat satrlar uchun
63. `b = a` bajarilsa nima bo'ladi (`a` ro'yxat)?
- A) Ikkala nom ham bitta ro'yxatga ishora qiladi
 - B) `b` da `a` ning mustaqil nusxasi paydo bo'ladi
 - C) `a` o'chiriladi
 - D) Xato yuz beradi
64. Ro'yxatning mustaqil nusxasini qaysi usul beradi?
- A) `a.copy()`
 - B) `b = a`
 - C) `a = b`
 - D) `del a`
65. `a = [10, 20, 30, 40, 50]` uchun `a[1:4]` natijasi qanday?
- A) `[10, 20, 30, 40]`
 - B) `[20, 30, 40, 50]`
 - C) `[30, 40]`
 - D) `[20, 30, 40]`
66. `[x * x for x in range(1, 4)]` natijasi qanday?
- A) `[1, 2, 3]`
 - B) `[1, 4, 9, 16]`
 - C) `[1, 4, 9]`
 - D) `[0, 1, 4]`
67. Kortej ro'yxatdan qaysi xususiyati bilan farq qiladi?
- A) Faqat sonlarni saqlaydi
 - B) O'zgarmas (immutable)
 - C) Indeksylanmaydi
 - D) Faqat bitta elementdan iborat bo'ladi

68. $[[0] * 3] * 3$ usuli bilan yaratilgan matritsada bitta elementni o'zgartirsak nima bo'ladi?

- A) Faqat shu element o'zgaradi
- B) Qatorlar bitta ob'yekt bo'lgani uchun bir necha qatordagi element o'zgaradi
- C) Dastur to'xtaydi
- D) Matritsa o'chib ketadi

69. $x, y = (3, 4)$ yozuvi qanday ataladi?

- A) Ochish (unpacking)
- B) Siklga o'tkazish
- C) Rekursiya
- D) Nusxalash

70. `enumerate(a)` nima beradi?

- A) Faqat elementlarni teskari tartibda
- B) Ro'yxat uzunligini
- C) Eng katta elementni
- D) Indeks va elementdan iborat juftliklarni

M8. Lug'atlar va to'plamlar

71. Lug'at (dict) qanday juftliklardan iborat?

- A) Indeks : element
- B) Nom : tur
- C) Kalit : qiymat
- D) Qator : ustun

72. Lug'at kaliti qanday bo'lishi kerak?

- A) Faqat ro'yxat
- B) Faqat o'zgaruvchan ob'yekt
- C) Noyob va o'zgarmas turdagi
- D) Faqat bo'sh satr

73. `d = {"a": 1}; d.get("b", 0)` natijasi qanday?

- A) `KeyError` xatosi
- B) 0
- C) None
- D) "b"

74. `d["b"]` ifodasi `d = {"a": 1}` da nima beradi?

- A) `KeyError` xatosi
- B) 0

C) None

D) {}

75. Lugʻat boʻyicha kalit va qiymat juftliklarini qaysi metod beradi?

A) keys()

B) values()

C) pop()

D) items()

76. Soʻzlar chastotasini hisoblashda odatda qaysi qator ishlatiladi?

A) d.append(s)

B) d[s] = s

C) d.sort(s)

D) d[s] = d.get(s, 0) + 1

77. {1, 2, 2, 3, 3, 3} toʻplami nechta elementdan iborat?

A) 6

B) 2

C) 3

D) 1

78. $a = \{1, 2, 3\}$, $b = \{2, 3, 4\}$. $a \& b$ natijasi qanday?

A) {1, 2, 3, 4}

B) {2, 3}

C) {1}

D) {4}

79. Boʻsh toʻplam qanday yaratiladi?

A) set()

B) {}

C) []

D) ()

80. Qaysi tuzilma noyob elementlarni saqlash va tegishlilikni tez tekshirish uchun qulay?

A) Toʻplam (set)

B) Kortej

C) Satr

D) Roʻyxat

M9. Funksiyalar, koʻrinish sohasi, lambda, rekursiya

81. Funksiya Python da qaysi kalit soʻz bilan yaratiladi?

A) func

- B) function
 - C) lambda faqat
 - D) def
82. Funksiyada return bo'lmasa, u nima qaytaradi?
- A) 0
 - B) Bo'sh satr
 - C) None
 - D) Xato beradi
83. print va return ning farqi nima?
- A) Ular butunlay bir xil
 - B) print ekranga chiqaradi, return qiymatni chaqirilgan joyga qaytaradi
 - C) return ekranga chiqaradi, print qiymat qaytaradi
 - D) print faqat sonlar uchun
84. Funksiya ichida yaratilgan o'zgaruvchi qanday ataladi?
- A) Global o'zgaruvchi
 - B) Lokal o'zgaruvchi
 - C) Standart o'zgaruvchi
 - D) Abstrakt o'zgaruvchi
85. def f(a=[]): kabi standart qiymat nima uchun xavfli?
- A) O'zgaruvchan ob'yekt barcha chaqiruvlarda umumiy bo'lib qoladi
 - B) Ro'yxatni funksiya parametri qilib bo'lmaydi
 - C) Sintaksis xatosi beradi
 - D) Funksiya ishlamaydi
86. lambda x: x * x nima?
- A) Sikl turi
 - B) Modul nomi
 - C) Ma'lumot turi
 - D) Nomsiz, bir ifodali qisqa funksiya
87. Rekursiv funksiya bazaviy holat nima uchun kerak?
- A) Funksiya tezligini oshirish uchun
 - B) Natijani chop etish uchun
 - C) Rekursiyani to'xtatish uchun
 - D) Parametrlarni o'zgartirish uchun
88. faktorial(n) funksiyasi uchun bazaviy holat qaysi?
- A) $n == 10$ bo'lganda 0 qaytarish
 - B) $n < 0$ bo'lganda n qaytarish
 - C) $n == 0$ bo'lganda 1 qaytarish

D) Bazaviy holat kerak emas

89. Bazaviy holatsiz rekursiv funksiya oxir-oqibat qanday xato beradi?

A) SyntaxError

B) RecursionError

C) KeyError

D) FileNotFoundError

90. Rekursiv Fibonachchi nima uchun sekin ishlaydi?

A) Bir xil qiymatlarni qayta-qayta hisoblaydi

B) Python rekursiyani bilmaydi

C) Faqat katta sonlarda xato beradi

D) Siklga qaraganda xotira ishlatmaydi

M10. Modullar, standart kutubxona, pip

91. Modul nima?

A) Dastur izohi

B) Ma'lumotlar turi

C) Faqat kompilyator

D) Boshqa dasturlarda qayta ishlatish mumkin bo'lgan Python fayli

92. `math.sqrt(16)` natijasi qanday?

A) 4

B) 8

C) 256

D) 4.0

93. Quyidagi import shakllaridan qaysi biri to'g'ri?

A) `import pi from math`

B) `include math`

C) `from math import pi`

D) `using math`

94. `from modul import *` nima uchun tavsiya etilmaydi?

A) Modul ishlamay qoladi

B) Nomlar to'qnashuviga olib kelishi mumkin

C) Dastur tezligi pasayadi

D) Python bunga ruxsat bermaydi

95. `random.randint(1, 6)` nima qaytaradi?

A) 1 dan 6 gacha (6 ham kiradi) tasodifiy butun son

B) 1 dan 5 gacha tasodifiy son

C) 0 dan 6 gacha haqiqiy son

D) Faqat 6

96. `datetime.date.today()` nimani beradi?

A) Bugungi sanani

B) Joriy soatni faqat

C) Tasodifiy sanani

D) Kecha sanani

97. `collections.Counter` nima uchun ishlatiladi?

A) Fayl o'chirish uchun

B) Tasodifiy son olish uchun

C) Matnni tarjima qilish uchun

D) Elementlarning uchrash sonini hisoblash uchun

98. `pip` nima?

A) Matn muharriri

B) Grafik kutubxona

C) Python paketlarini o'rnatuvchi paket menejeri

D) Ma'lumotlar bazasi

99. `Virtual` muhit (`venv`) nima uchun kerak?

A) Dasturni tezlashtirish uchun

B) Loyihalar paketlarini bir-biridan ajratish uchun

C) Kompyuterni o'chirish uchun

D) Faylni shifrlash uchun

100. `if __name__ == "__main__":` sharti nima vazifani bajaradi?

A) Dasturni to'xtatadi

B) Kod fayl to'g'ridan-to'g'ri ishga tushirilgandagina bajariladi (`import` da emas)

C) Modulni o'chiradi

D) Python versiyasini tekshiradi

M11. Fayllar, CSV/JSON, istisnolar

101. `open(fayl, "w")` rejimi mavjud faylga nima qiladi?

A) Eski mazmunni o'chirib, qaytadan yozadi

B) Oxiriga qo'shadi

C) Faqat o'qishga ochadi

D) Faylni o'chirmaydi va o'zgartirmaydi

102. Fayl oxiriga yangi ma'lumot qo'shish uchun qaysi rejim ishlatiladi?

A) "r"

B) "w"

C) “x” faqat

D) “a”

103. with open(...) as f: konstruksiyasining afzalligi nima?

A) Fayl hajmi kamayadi

B) Fayl shifrlanadi

C) Fayl ish tugagach avtomatik yopiladi

D) Fayl nusxalanadi

104. O‘zbek harflari bilan ishlashda fayl ochishda nimani ko‘rsatish tavsiya etiladi?

A) mode=”uz”

B) lang=”uz”

C) encoding=”utf-8”

D) format=”text”

105. csv moduli yordamida o‘qilgan qiymatlar qanday turda bo‘ladi?

A) Har doim int

B) Satr (str)

C) Har doim float

D) bool

106. JSON da ma’lumotlar qanday ko‘rinishda saqlanadi?

A) Lug‘at va ro‘yxatlarga o‘xshash matn shaklida

B) Faqat ikkilik rasm shaklida

C) Faqat jadval katakchalarida

D) Faqat shifrlangan holda

107. json.dump(...) nima qiladi?

A) JSON faylni o‘qiydi

B) Faylni o‘chiradi

C) Faylni nusxalaydi

D) Ma’lumotni JSON formatida faylga yozadi

108. int(“abc”) qanday istisno beradi?

A) ZeroDivisionError

B) KeyError

C) IndexError

D) ValueError

109. try/except konstruksiyasida else bloki qachon bajariladi?

A) Har doim

B) Faqat xato bo‘lganda

C) try blokida xato bo‘lmagan hollarda

D) Hech qachon

110. Nima uchun yalang except: dan qochish kerak?

A) U sintaksis xatosi beradi

B) U barcha xatolarni, jumladan dasturchi xatolarini ham yashirib yuboradi

C) U dasturni sekinlashtiradi

D) Python bunga ruxsat bermaydi

M12. OYD: sinflar, ob'yektlar, atributlar, metodlar

111. Sinf (class) nima?

A) Ob'yektlar yaratish uchun andoza (qolip)

B) Dasturning bitta qatori

C) Matematik funksiya

D) Fayl turi

112. Ob'yekt nima?

A) Sinfning aniq nusxasi

B) Sinfning nomi

C) Sinfning izohi

D) Modul

113. __init__ metodi qachon chaqiriladi?

A) Dastur tugaganda

B) Ob'yekt o'chirilganda

C) Hech qachon avtomatik chaqirilmaydi

D) Sinfdan yangi ob'yekt yaratilganda

114. Metod ta'rifidagi birinchi parametr self nimani bildiradi?

A) Sinf nomini

B) Dastur nomini

C) Metod chaqirilayotgan ob'yektning o'ziga ishorani

D) Fayl nomini

115. Ob'yekt atributi va sinf atributining farqi nima?

A) Ular bir xil

B) Ob'yekt atributi har bir ob'yektda alohida, sinf atributi barcha ob'yektlar uchun umumiy

C) Sinf atributi faqat sonlar bo'ladi

D) Ob'yekt atributi faqat satr bo'ladi

116. __str__ metodi nima uchun ishlatiladi?

A) Ob'yektni o'chirish uchun

B) print() da ob'yektning matn ko'rinishini belgilash uchun

- C) Ob'yektni nusxalash uchun
 - D) Sinf dan voris olish uchun
117. class Talaba: nomi nima uchun odatda katta harf bilan boshlanadi?
- A) Sinf nomlari uchun PascalCase uslubi qabul qilingan
 - B) Python boshqa nomga ruxsat bermaydi
 - C) Kichik harf xato beradi
 - D) Katta harf tezroq ishlaydi
118. t = Talaba("Ali", 17) ifodasi nimani bajaradi?
- A) Talaba sinfini o'chiradi
 - B) Talaba funksiyasini chop etadi
 - C) Faylni ochadi
 - D) Talaba sinfidan yangi ob'yekt yaratadi
119. OYD ning asosiy g'oyasi qaysi qatorda to'g'ri ko'rsatilgan?
- A) Dasturni faqat sikllar bilan yozish
 - B) Faqat global o'zgaruvchilardan foydalanish
 - C) Ma'lumot va ular ustidagi amallarni ob'yektda birlashtirish
 - D) Dasturda funksiyalardan foydalanmaslik
120. __add__ maxsus metodi qaysi amal xulqini belgilaydi?
- A) - operatori
 - B) == operatori
 - C) + operatori
 - D) len() funksiyasi

M13. OYD tamoyillari: inkapsulyatsiya, vorislik, polimorfizm

121. Inkapsulyatsiya nima?
- A) Sinfni o'chirish
 - B) Ma'lumot va amallarni birlashtirib, ichki tuzilishga tashqaridan murojaatni cheklash
 - C) Dasturni tezlashtirish
 - D) Bir nechta faylni birlashtirish
122. Python da atribut nomi oldiga pastki chiziq (_nom) qo'yish nimani bildiradi?
- A) Ichki foydalanish uchun, tashqaridan murojaat qilmaslik kelishuvi
 - B) Atribut o'chirilgan
 - C) Atribut faqat son bo'ladi
 - D) Atribut avtomatik saqlanadi
123. @property dekoratori nima uchun ishlatiladi?

- A) Sinfni o‘chirish uchun
- B) Faylni ochish uchun
- C) Siklni to‘xtatish uchun
- D) Metodni oddiy atribut kabi o‘qish imkonini berish uchun

124. Vorislik nima?

- A) Ikki ob’yektni solishtirish
- B) Funksiyani o‘chirish
- C) Atributni yashirish
- D) Hosila sinf asosiy sinf xususiyatlarini olishi

125. Hosila sinfda asosiy sinf metodini chaqirish uchun nima ishlatiladi?

- A) self.parent
- B) import
- C) super()
- D) base()

126. Polimorfizm nima?

- A) Sinfning ko‘p nusxasi
- B) Bir xil nomli metodni turli ob’yektlar o‘ziga xos tarzda bajarishi
- C) Faqat bitta sinf mavjudligi
- D) Ma’lumotlarni yashirish

127. Metodni qayta aniqlash (override) nima?

- A) Hosila sinfda asosiy sinfdagi metodni o‘zgartirib yozish
- B) Metodni o‘chirish
- C) Sinf nomini o‘zgartirish
- D) Fayl nomini o‘zgartirish

128. “Talaba – bu Shaxs” munosabati qanday ataladi?

- A) “is-a” (vorislik)
- B) “has-a” (kompozitsiya)
- C) “do-a” (sikl)
- D) “no-a” (inkor)

129. “Avtomobil dvigatelga ega” munosabati qaysi tushunchaga mos?

- A) Vorislik (“is-a”)
- B) Polimorfizm
- C) Rekursiya
- D) Kompozitsiya (“has-a”)

130. Shubha bo‘lganda odatda qaysi yondashuv tavsiya etiladi?

- A) Har doim vorislikdan foydalanish
- B) Hech qachon sinf yaratmaslik

- C) Kompozitsiyani vorislikdan afzal ko‘rish
- D) Faqat global o‘zgaruvchi ishlatish

M14. Algoritmlar murakkabligi, saralash va qidirish

131. O-notatsiya nimani ifodalaydi?

- A) Dastur hajmini baytlarda
- B) Amallar sonining kirish hajmi n ga bog‘liq o‘rish tartibini
- C) Kompyuterning chastotasini
- D) Dasturchining malakasini

132. Ro‘yxat elementiga indeks bo‘yicha murojaat qanday murakkablikka ega?

- A) $O(n)$
- B) $O(1)$
- C) $O(n^2)$
- D) $O(\log n)$

133. Chiziqli qidiruvning murakkabligi qanday?

- A) $O(n)$
- B) $O(1)$
- C) $O(\log n)$
- D) $O(n^2)$

134. Ikkilik qidiruv uchun nima talab qilinadi?

- A) Ma’lumotlar teskari tartibda bo‘lishi
- B) Faqat satrlar bo‘lishi
- C) Ma’lumotlar bo‘sh bo‘lishi
- D) Ma’lumotlar tartiblangan bo‘lishi

135. Ikkilik qidiruvning murakkabligi qanday?

- A) $O(n)$
- B) $O(n^2)$
- C) $O(\log n)$
- D) $O(1)$

136. 1000 elementli tartiblangan ro‘yxatda ikkilik qidiruv taxminan necha qadamda natijaga erishadi?

- A) 1000
- B) 500
- C) Taxminan 10
- D) 100000

137. Pufakchali saralashning odatdagi murakkabligi qanday?

A) $O(n \log n)$

B) $O(n^2)$

C) $O(\log n)$

D) $O(1)$

138. Birlashtirib saralash (mergesort) ning murakkabligi qanday?

A) $O(n \log n)$

B) $O(n^2)$

C) $O(1)$

D) $O(n!)$ faqat

139. “Bo‘lib tashla va hukmronlik qil” tamoyiliga qaysi algoritm asoslangan?

A) Pufakchali saralash

B) Chiziqli qidiruv

C) Eng katta elementni topish

D) Tez saralash va birlashtirib saralash

140. Ichma-ich ikkita sikl (har biri n marta) odatda qanday murakkablik beradi?

A) $O(n)$

B) $O(\log n)$

C) $O(1)$

D) $O(n^2)$

M15. Ma'lumotlar tuzilmalari: stek, navbat, bog'langan ro'yxat, daraxt, graf

141. Stek qaysi tamoyil asosida ishlaydi?

A) FIFO (birinchi kirgan birinchi chiqadi)

B) Tasodifiy chiqish

C) LIFO (oxirgi kirgan birinchi chiqadi)

D) Tartiblangan chiqish

142. Navbat qaysi tamoyil asosida ishlaydi?

A) LIFO

B) FIFO (birinchi kirgan birinchi chiqadi)

C) Tasodifiy

D) Teskari tartib

143. Navbat uchun Python da qaysi tuzilma samaraliroq?

A) `collections.deque`

B) Oddiy ro'yxat (`pop(0)`)

C) Kortej

D) To'plam

144. Qavslar balansini tekshirishda qaysi tuzilma qo'llanadi?
- A) Stek
 - B) Navbat
 - C) Lug'at
 - D) To'plam
145. Bog'langan ro'yxatning har bir tuguni nimani saqlaydi?
- A) Faqat indeks
 - B) Faqat o'lcham
 - C) Faqat tur
 - D) Qiymat va keyingi tugunga havola
146. Daraxtda bolasi yo'q tugun qanday ataladi?
- A) Ildiz
 - B) Qirra
 - C) Barg
 - D) Uch
147. Ikkilik daraxtda har bir tugunning nechta bolasi bo'lishi mumkin?
- A) Faqat bitta
 - B) Ko'pi bilan ikkita
 - C) Kamida uchta
 - D) Cheksiz ko'p
148. Graf nimalardan iborat?
- A) Faqat ildizdan
 - B) Uchlar va qirralardan
 - C) Faqat massivlardan
 - D) Faqat funksiyalardan
149. BFS algoritmi qaysi tuzilmadan foydalanadi?
- A) Navbat
 - B) Stek (odatda)
 - C) To'plam faqat
 - D) Kortej
150. Eng qisqa yo'lni (qirralar soni bo'yicha) qaysi usul topadi?
- A) DFS har doim
 - B) Pufakchali saralash
 - C) Chiziqli qidiruv
 - D) BFS

M16. C++ ga kirish: dastur tuzilishi, turlar, boshqaruv operatorlari

151. C++ dasturi qaysi funksiyadan boshlanadi?

- A) start()
- B) begin()
- C) main()
- D) run()

152. C++ da buyruq qanday belgi bilan tugaydi?

- A) Ikki nuqta (:)
- B) Nuqta (.)
- C) Nuqtali vergul (;)
- D) Vergul (,)

153. C++ qanday til hisoblanadi?

- A) Faqat interpretatsiya qilinadigan, dinamik tiplashgan til
- B) Kompilyatsiya qilinadigan, statik tiplashgan til
- C) Faqat vizual til
- D) Faqat skript til

154. C++ da $7 / 2$ (ikkala operand int) natijasi qanday?

- A) 3
- B) 3.5
- C) 4
- D) 2

155. Katta butun sonlar uchun (taxminan 10^{18} gacha) qaysi tur mos?

- A) int
- B) char
- C) bool
- D) long long

156. int turi to'lib ketishi (overflow) nima?

- A) Xotiraning tozalanishi
- B) Dasturning tezlashishi
- C) Faylning o'chishi
- D) Qiymat turning chegarasidan oshganda noto'g'ri natija chiqishi

157. C++ da qiymat o'qish uchun qaysi ifoda ishlatiladi?

- A) input(a)
- B) read a
- C) cin >> a;
- D) get(a)

158. C++ da chiqarish uchun qaysi ifoda ishlatiladi?

- A) print(a)
- B) cout << a;
- C) output a
- D) write(a)

159. for (int i = 0; i < 5; i++) sikli necha marta bajariladi?

- A) 5
- B) 4
- C) 6
- D) 0

160. Python bilan solishtirganda C++ da blok qanday belgilanadi?

- A) Figurali qavslar { } bilan
- B) Faqat chekinish bilan
- C) begin va end bilan
- D) Ikki nuqta bilan

M17. C++ da massivlar, funksiyalar, STL, olimpiada

161. C++ da int a[5] massivining indeksleri qaysi oraliqda?

- A) 1 dan 5 gacha
- B) 0 dan 5 gacha
- C) 1 dan 4 gacha
- D) 0 dan 4 gacha

162. C++ da massiv chegarasidan chiqib murojaat qilish nimaga olib keladi?

- A) Har doim aniq xato xabariga va avtomatik tuzatishga
- B) Avtomatik kattalashishga
- C) Aniqlanmagan xulqqa (xato natija yoki to'xtash)
- D) Hech narsaga

163. void swap2(int &a, int &b) funksiyasida & belgisi nimani anglatadi?

- A) Parametr o'chirilishini
- B) Parametr havola orqali uzatilishini
- C) Parametr satr ekanligini
- D) Mantiqiy "va" amalini

164. STL dagi vector Python ning qaysi tuzilmasiga o'xshaydi?

- A) dict (lug'at)
- B) list (ro'yxat)
- C) set (to'plam)
- D) str (satr)

165. STL dagi map Python ning qaysi tuzilmasiga o'xshaydi?

- A) dict (lugʻat)
- B) list (roʻyxat)
- C) tuple (kortej)
- D) range

166. `vector<int> v; v.push_back(5);` nima qiladi?

- A) Vektordan 5 ni oʻchiradi
- B) Vektorni saralaydi
- C) Vektorni boʻshatadi
- D) Vektor oxiriga 5 ni qoʻshadi

167. `sort(v.begin(), v.end())` nima qiladi?

- A) Vektorni teskari qiladi
- B) Vektorni oʻchiradi
- C) Vektorni oʻsish tartibida saralaydi
- D) Eng katta elementni topadi

168. `ios::sync_with_stdio(false);` nima uchun yoziladi?

- A) Dasturni toʻxtatish uchun
- B) Xotirani tozalash uchun
- C) `cin` va `cout` ni tezlashtirish uchun
- D) Faylni ochish uchun

169. Olimpiada masalasida “Time Limit Exceeded” hukmi nimani bildiradi?

- A) Dastur toʻgʻri ishlagan
- B) Dastur belgilangan vaqtdan koʻproq ishlagan
- C) Dastur kompilyatsiya boʻlmagan
- D) Dastur xotira ajratmagan

170. Olimpiada masalasini yechishdan oldin nima qilish tavsiya etiladi?

- A) Shart, chegaralar va chekka holatlarni diqqat bilan tahlil qilish
- B) Darhol kodlashni boshlash
- C) Chegaralarga eʼtibor bermaslik
- D) Faqat namunaviy testga qarab yozish

M18. Tkinter va Turtle

171. Hodisaga asoslangan dasturlashda dastur nimani kutadi?

- A) Faqat vaqt tugashini
- B) Faqat fayl oʻchishini
- C) Hech narsani
- D) Foydalanuvchi harakatlarini (tugma bosish, matn kiritish)

172. Tkinter da dastur oynasi qaysi buyruq bilan “tinglash” sikliga oʻtadi?

A) start()

B) run()

C) loop()

D) mainloop()

173. Tkinter da matn kiritish uchun qaysi vidjet ishlatiladi?

A) Label

B) Canvas

C) Entry

D) Frame

174. Tkinter da tugma command parametriga funksiya qanday beriladi?

A) Funksiya nomi qavs bilan (command=hisobla())

B) Funksiya nomi qavssiz (command=hisobla)

C) Funksiya matni qo'shtirnoqda

D) Funksiya raqami bilan

175. Tkinter da vidjetlarni jadval ko'rinishida joylashtirish usuli qaysi?

A) grid()

B) table()

C) line()

D) put()

176. Turtle da toshbaqani 100 piksel oldinga yurgizish buyrug'i qaysi?

A) forward(100)

B) go(100)

C) move(100)

D) front(100)

177. Turtle da muntazam n-burchak chizishda har bir burchakka necha gradusga burilinadi?

A) $180 / n$

B) $90 / n$

C) $n * 360$

D) $360 / n$

178. Turtle da chizmasdan joyini o'zgartirish uchun nima qilinadi?

A) pendown() ni ishlatish

B) clear() ni ishlatish

C) penup() bilan qalamni ko'tarib, harakatlanish

D) exit() ni ishlatish

179. Turtle yordamida rekursiyani o'rgatishda qaysi shakl mos?

A) Faqat to'g'ri chiziq

- B) Rekursiv daraxt va fraktallar
 - C) Faqat matn
 - D) Faqat nuqta
180. Turtle ning metodik afzalligi nima?
- A) Katta loyihalar uchun eng yaxshi vosita
 - B) Natija darhol ko‘rinadi, xatolar vizual sezilarli
 - C) Faqat professional dasturlash uchun
 - D) Xatoga yo‘l qo‘ymaydi

M19. NumPy, pandas va matplotlib

181. NumPy ning ro‘yxatga nisbatan asosiy afzalligi nima?
- A) Sonli hisoblar samarali va vektorlashtirilgan amallar mumkin
 - B) U faqat satrlar bilan ishlaydi
 - C) U ro‘yxatdan sekinroq ishlaydi va faqat butun sonlarni saqlaydi
 - D) U faqat grafik chizadi
182. `np.array([1, 2, 3]) * 2` natijasi qanday?
- A) `[1 2 3 1 2 3]`
 - B) `[1 2 3 2]`
 - C) Xato
 - D) `[2 4 6]`
183. `[1, 2, 3] * 2` (oddiy ro‘yxat) natijasi qanday?
- A) `[2, 4, 6]`
 - B) `[1, 2, 3, 2]`
 - C) `[1, 2, 3, 1, 2, 3]`
 - D) Xato
184. pandas da ikki o‘lchovli jadval ob’yekti qanday ataladi?
- A) Series
 - B) Matrix
 - C) DataFrame
 - D) Tuple
185. CSV faylni pandas bilan o‘qish uchun qaysi funksiya ishlatiladi?
- A) `pd.load_text()`
 - B) `pd.read_csv()`
 - C) `pd.open_csv()`
 - D) `pd.file()`
186. `df[df[“ball”] >= 90]` ifodasi nima qiladi?
- A) Balli 90 va undan yuqori qatorlarni tanlaydi

- B) Barcha qatorlarni o‘chiradi
 - C) Faqat ustun nomlarini chiqaradi
 - D) Jadvalni saralaydi
187. `df.groupby(“sinf”)[“ball”].mean()` nima beradi?
- A) Eng katta ballni
 - B) Barcha ballar yig‘indisini bitta songa
 - C) Qatorlar sonini
 - D) Har bir sinf bo‘yicha o‘rtacha ballni
188. Vaqt bo‘yicha o‘zgarishni ko‘rsatishga qaysi diagramma turi mos?
- A) Doiraviy
 - B) Doiraviy yoki gistogramma emas, ularning birortasi ham mos kelmaydi
 - C) Uch o‘lchovli jadval
 - D) Chiziqli
189. Ma’lumotlarni tahlil qilishning to‘g‘ri ketma-ketligi qaysi?
- A) Xulosa, vizuallashtirish, savol qo‘yish
 - B) Faqat vizuallashtirish
 - C) Savol qo‘yish, ma’lumot yig‘ish, tozalash, tahlil, vizuallashtirish, xulosa
 - D) Tozalash, xulosa, ma’lumot yig‘ish, savol
190. Maktab o‘quvchilari bilan ma’lumot tahlilida shaxsiy ma’lumotlarga munosabat qanday bo‘lishi kerak?
- A) Barcha o‘quvchilarning telefon raqamlarini yig‘ish
 - B) Anonim yoki o‘ylab topilgan ma’lumotlardan foydalanish
 - C) Parollarni to‘plash
 - D) Ma’lumotlarni ochiq e’lon qilish

M20. Git/GitHub, testlash, kod sifati, SI yordamchilari

191. Git nima?
- A) Versiyalarni boshqarish tizimi
 - B) Grafik muharrir
 - C) Dasturlash tili
 - D) Brauzer
192. `git commit` nima qiladi?
- A) O‘zgarishlarni tarixga saqlaydi
 - B) Faylni o‘chiradi
 - C) Repozitoriyni o‘chiradi
 - D) Internetdan fayl yuklaydi
193. `git add` buyrug‘i nima qiladi?

- A) Faylni o‘chiradi
 - B) Commit tarixini o‘chiradi
 - C) Dasturni ishga tushiradi
 - D) Faylni keyingi commit uchun tayyorlov (staging) maydoniga qo‘shadi
194. GitHub nima?
- A) Dasturlash tili
 - B) Antivirus
 - C) Git repozitoriylarni onlayn saqlash va hamkorlikda ishlash platformasi
 - D) Operatsion tizim
195. Pull request nima?
- A) Faylni o‘chirish buyrug‘i
 - B) O‘zgarishni asosiy tarmoqqa qo‘shishni taklif qilish va ko‘rib chiqish
 - C) Dasturni ishga tushirish
 - D) Parolni tiklash
196. Repozitoriyga nimani joylamaslik kerak?
- A) README faylini
 - B) Parollar va maxfiy kalitlarni
 - C) Dastur kodini
 - D) Testlarni
197. Birlik testi (unit test) nima?
- A) Dastur kichik qismining (funksiyaning) to‘g‘riligini tekshiruvchi kod
 - B) Dastur hajmini o‘lchaydi
 - C) Dastur nusxasini saqlaydi
 - D) Dasturni shifrlaydi
198. Python uslub qo‘llanmasi qanday ataladi?
- A) PEP 1000
 - B) ISO 9000
 - C) GOST 7
 - D) PEP 8
199. Sun‘iy intellekt yordamchisi bergan koddan foydalanishda nima qilish kerak?
- A) Tekshirmasdan ishonish
 - B) Tushunmasdan nusxalash
 - C) Kodni tushunish, tekshirish va sinash
 - D) Maxfiy ma‘lumotlarni yuborish
200. Akademik halollik nuqtayi nazaridan to‘g‘ri yo‘l qaysi?
- A) SI yozgan yechimni o‘zinikidek topshirish

B) SI dan foydalanganini yashirish

C) SI dan foydalanish qoidalariga rioya qilish va foydalanilganini ochiq ko'rsatish

D) Boshqalarning kodini ko'chirish

Javoblar kaliti

Jadvalda har bir mavzu bo'yicha (M1–M20) savollarning tartib raqami va to'g'ri javob harfi keltirilgan.

Savollar	1	2	3	4	5	6	7	8	9	10
M1 (1–10)	D	C	B	B	A	D	C	C	B	A
M2 (11–20)	D	D	C	B	A	A	D	C	B	B
M3 (21–30)	A	D	C	C	B	A	D	D	C	B
M4 (31–40)	A	A	D	C	B	B	A	D	C	C
M5 (41–50)	B	A	D	D	C	B	A	A	D	C
M6 (51–60)	B	B	A	D	C	C	B	A	D	D
M7 (61–70)	C	B	A	A	D	C	B	B	A	D
M8 (71–80)	C	C	B	A	D	D	C	B	A	A
M9 (81–90)	D	C	B	B	A	D	C	C	B	A
M10 (91–100)	D	D	C	B	A	A	D	C	B	B
M11 (101–110)	A	D	C	C	B	A	D	D	C	B
M12 (111–120)	A	A	D	C	B	B	A	D	C	C
M13 (121–130)	B	A	D	D	C	B	A	A	D	C
M14 (131–140)	B	B	A	D	C	C	B	A	D	D
M15 (141–150)	C	B	A	A	D	C	B	B	A	D
M16 (151–160)	C	C	B	A	D	D	C	B	A	A
M17 (161–170)	D	C	B	B	A	D	C	C	B	A
M18 (171–180)	D	D	C	B	A	A	D	C	B	B
M19 (181–190)	A	D	C	C	B	A	D	D	C	B
M20 (191–200)	A	A	D	C	B	B	A	D	C	C

XI. YAKUNIY NAZORAT (YOZMA ISH) MATERIALLARI VA BAHOLASH MEZONLARI

Yakuniy nazorat yozma ish shaklida o'tkaziladi. Har bir variant 3 ta topshiriqdan iborat: 1–2-savollar – nazariy, 3-topshiriq – amaliy (dasturlash). Nazariy savollar fanning mazmuni bo'yicha bilimni, amaliy topshiriq esa dastur yozish, uni sinash va izohlash ko'nikmasini tekshiradi. Yozma ish uchun 80 daqiqa ajratiladi. Yakuniy nazorat jami bahoning 60 foizini tashkil etadi (qolgan 40 foiz – joriy nazorat).

1. Yakuniy nazorat uchun nazariy savollar

1. Dasturlash tillari qanday belgilar bo'yicha tasniflanadi? Paradigma tushunchasini misollar bilan tushuntiring.
2. Dasturlash tillarining rivojlanish bosqichlarini (mashina kodidan zamonaviy tillargacha) bayon qiling.
3. Kompilyatsiya va interpretatsiyani taqqoslang; Python dasturi qanday bajarilishini (bayt-kod, virtual mashina) tushuntiring.
4. Python tilining asosiy xususiyatlari va qo'llanish sohalarini bayon qiling; Python ni o'rnatish va muhit tanlash masalalarini yoritng.
5. IDE, REPL va Jupyter Notebook tushunchalarini tushuntiring; maktab uchun dasturlash muhitini tanlash mezonlarini keltiring.
6. Xato xabarini (traceback) o'qish va boshlovchilarning tipik xatolari haqida gapiring; ularni o'rgatish metodikasini bayon qiling.
7. O'zgaruvchi va ma'lumot turlari (int, float, str, bool) tushunchalarini tushuntiring; turlarni o'zgartirish va tipik xatolarni sanang.
8. Python da operatorlar (arifmetik, taqqoslash, mantiqiy) va ularning ustuvorligini misollar bilan tushuntiring.
9. input() va print() funksiyalarining imkoniyatlarini, f-string formatlashni tushuntiring.
10. Shartli operatorlar (if, elif, else) va ularning ishlash tamoyilini bayon qiling; elif zanjirida shartlar tartibining ahamiyatini tushuntiring.
11. match operatori va chekka holatlarni tekshirish tushunchalarini tushuntiring; shartli operatorni o'qitish metodikasini yoritng.
12. Mantiqiy ifodalar, haqiqiylik jadvallari va murakkab shartlarni tuzish qoidalarini bayon qiling.
13. for va while sikllarini taqqoslang; range() funksiyasi va akkumulyator hamda hisoblagich tushunchalarini tushuntiring.
14. break, continue va cheksiz sikl tushunchalarini bayon qiling; kuzatuv jadvalining ahamiyatini tushuntiring.

15. Ichma-ich sikllar va ularning qo'llanishini (naqshlar, jadvallar) tushuntiring; takrorlar sonini baholang.
16. Satrlar, indekslash va kesimlarni bayon qiling; satrning o'zgarmasligini tushuntiring.
17. Satr metodlarini (split, join, replace, find va boshqalar) sanab, misollar bilan tushuntiring.
18. Unicode va belgilarni kodlash (ord, chr) haqida gapiring; Sezar shifrini tushuntiring.
19. Ro'yxatlar, ularning metodlari va kesimlarini bayon qiling; $b = a$ va nusxalash farqini tushuntiring.
20. List comprehension va ikki o'lchovli ro'yxatlar (matritsalar) haqida gapiring; tipik xatolarni sanang.
21. Kortejlar va ularning o'zgarmasligi, ochish (unpacking) tushunchalarini tushuntiring.
22. Lug'atlar tuzilishi, metodlari va chastota lug'atini tuzish usulini bayon qiling.
23. To'plamlar va to'plam amallarini (birlashma, kesishma, ayirma) tushuntiring; tuzilma tanlash mezonlarini keltiring.
24. Funksiya yaratish, parametrlar, return va standart qiymatlarni bayon qiling; print va return farqini tushuntiring.
25. Lokal va global ko'rinish sohasi hamda lambda funksiyalarni tushuntiring.
26. Rekursiya g'oyasi, bazaviy holat va chaqiruvlar stekini tushuntiring; faktorial va Fibonachchi misollarida rekursiya va siklni solishtiring.
27. Modul va paket tushunchalari, import usullarini bayon qiling; math, random va datetime modullarini tushuntiring.
28. pip, PyPI va virtual muhit (venv) tushunchalarini tushuntiring; o'z modulini yaratishni bayon qiling.
29. Standart kutubxonaning qo'shimcha modullari (collections, statistics, time) va ularning qo'llanishini misollar bilan bayon qiling.
30. Python tilida qiymat berish, ob'yektga havola va o'zgaruvchan hamda o'zgarmas turlar farqini tushuntiring.
31. Dasturlashni o'rganishda PRIMM va juft dasturlash metodlarining mohiyatini tushuntiring.
32. Fayllar bilan ishlash (open, rejimlar, with, encoding) va CSV formati bilan ishlashni bayon qiling.
33. JSON formati va json moduli bilan ishlashni tushuntiring; JSON va CSV ni solishtiring.

34. Istisnolar va try/except/else/finally konstruksiyasini bayon qiling; kiritilgan ma'lumotni tekshirish usullarini keltiring.
35. OYD ning g'oyasi, sinf va ob'yekt tushunchalari, __init__ va self ni tushuntiring.
36. Atribut va metodlar, sinf va ob'yekt atributlari, maxsus (dunder) metodlarni misollar bilan bayon qiling.
37. Inkapsulyatsiya va @property ning ahamiyatini tushuntiring; OYD ni maktabda o'qitish xususiyatlarini yorit.
38. Vorislik, super() va metodni qayta aniqlashni bayon qiling; "is-a" va "has-a" munosabatlarini solishtiring.
39. Polimorfizm va abstraksiya tushunchalarini misollar bilan tushuntiring.
40. Algoritmalar murakkabligi va O-notatsiyani tushuntiring; asosiy murakkabliklarga misollar keltiring.
41. Chiziqli va ikkilik qidiruv algoritmlarini taqqoslang; ikkilik qidiruv uchun talablarni tushuntiring.
42. Oddiy saralash usullarini (pufakchali, tanlash, qo'shish) bayon qiling va murakkabligini baholang.
43. Tez saralash va birlashtirib saralash algoritmlarining g'oyasini tushuntiring; "bo'lib tashla va hukmronlik qil" tamoyilini bayon qiling.
44. Stek va navbat tuzilmalari, ularning amallari va qo'llanishlarini misollar bilan tushuntiring.
45. Bog'langan ro'yxat va daraxt tuzilmalarini bayon qiling; ikkilik qidiruv daraxtining xususiyatlarini keltiring.
46. Graf tushunchasi, uni ifodalash usullari hamda BFS va DFS algoritmlari g'oyasini bayon qiling.
47. C++ ning xususiyatlari, dastur tuzilishi va kompilyatsiya jarayonini bayon qiling.
48. C++ da statik turlar, int to'lib ketishi va kiritish-chiqarishni tushuntiring; Python bilan solishtiring.
49. C++ da massivlar va funksiyalar (qiymat va havola bo'yicha uzatish) ni bayon qiling.
50. STL konteynerlari (vector, map, set) va sort ni tushuntiring; ularni Python tuzilmalari bilan solishtiring.
51. Olimpiada masalalarini yechish strategiyasi: shartni tahlil qilish, chegaralar, chekka holatlar va hukmlar (WA, TLE, RE) ni bayon qiling.
52. Hodisaga asoslangan dasturlash va Tkinter asoslarini (vidjetlar, joylashtirish, hodisalar) bayon qiling.

53. Turtle grafikasi va uning yordamida sikl, funksiya va rekursiyani o'rgatish metodikasini yoritish.
54. NumPy va vektorlashtirish g'oyasini tushuntiring; ro'yxat bilan farqini ko'rsating.
55. pandas (Series, DataFrame) va matplotlib bilan ma'lumotlarni qayta ishlash va vizuallashtirishni bayon qiling.
56. Ma'lumotlar tahlilining asosiy bosqichlarini tushuntiring; maktab o'quvchilari bilan tahlilda maxfiylikka rioya qilish talablarini keltiring.
57. Versiyalarni boshqarish, Git ning asosiy buyruqlari, GitHub, branch, pull request va README tushunchalarini bayon qiling.
58. Testlash (assert, unittest, pytest) va debugging bosqichlarini bayon qiling.
59. Kod sifati: PEP 8, DRY, refaktoring va kod sharhi (code review) tushunchalarini tushuntiring.
60. Sun'iy intellekt yordamchilaridan dasturlashda foydalanishning imkoniyat va xavflarini, akademik halollik qoidalarini bayon qiling.

2. Yakuniy nazorat uchun amaliy (dasturlash) topshiriqlari

1. Foydalanuvchidan ikki butun son olib, ularning yig'indisi, ayirmasi, ko'paytmasi va butun bo'linmasi hamda qoldig'ini chiqaruvchi Python dasturini yozing va uni 5 ta test bilan tekshiring.
2. Ball (0–100) bo'yicha 5/4/3/2 baho chiqaruvchi dastur yozing; chekka holatlar test jadvalini tuzing.
3. 1 dan n gacha sonlar orasida 3 ga va 5 ga bir vaqtda bo'linadiganlari yig'indisi va sonini hisoblovchi dastur yozing; $n = 30$ uchun kuzatuv jadvalini to'ldiring.
4. Sonning raqamlari yig'indisini while sikli bilan hisoblovchi dastur yozing; 4075 uchun kuzatuv jadvalini tuzing.
5. Ichma-ich sikllar yordamida n qatorli piramida (yulduzchalardan) chizuvchi dastur yozing.
6. Matn kiritilganda undagi so'zlar soni, unli harflar soni va eng uzun so'zni topuvchi dastur yozing; palindromni tekshiruvchi funksiya qo'shing.
7. Ro'yxatdagi sonlarning eng katta, eng kichik elementi va o'rtachasini topuvchi funksiyalar yozing va sorted() bilan solishtiring.
8. Matndagi so'zlar chastotasini lug'at yordamida hisoblab, eng ko'p uchragan 3 so'zni chiqaruvchi dastur yozing.
9. Sonning tubligini tekshiruvchi funksiya yozing va undan foydalanib n gacha tub sonlarni chiqaruvchi dastur tuzing; murakkablikni baholang.

10. Rekursiya yordamida sonning faktorialini va Fibonachchi sonini hisoblovchi funksiyalar yozing; ularning siklli variantlarini yozib, vaqtini solishtiring.
11. Fayldan sonlarni o'qib, ularning yig'indisi va o'rtachasini hisoblab, natijani boshqa faylga yozuvchi dastur yozing; faylning mavjud emasligini try/except bilan qayta ishlang.
12. Talabalar ismi va ballari yozilgan CSV faylni o'qib, o'rtacha ball va eng yuqori ball egasini aniqlovchi, natijani JSON ga saqlovchi dastur yozing.
13. Foydalanuvchi to'g'ri butun son kiritguncha so'raydigan funksiya yozing (try/except bilan) va undan "Sonni top" o'yinida foydalaning.
14. "Hisob" sinfini (egasi, balans; qo'shish va yechish metodlari, mablag' yetmaganda xabar) yozing va uni sinab ko'ring.
15. Shakl asosiy sinfi va Doira hamda Kvadrat hosila sinflarini yozing; yuza() metodi polimorfizmini ko'rsating.
16. Pufakchali saralash algoritmini yozing va 1000 hamda 2000 elementli ro'yxatlarda vaqtini sorted() bilan solishtiring; natijani izohlang.
17. Ikkilik qidiruv funksiyasini yozing va 1000 elementli tartiblangan ro'yxatda qadamlar sonini sanab chiqing.
18. Stek yordamida qavslar balansini (uch turdagi qavs) tekshiruvchi funksiya yozing.
19. Graf qo'shnilik ro'yxati bilan berilgan; BFS yordamida boshlang'ich uchdan boshqa uchlargacha masofalarni hisoblovchi dastur yozing.
20. C++ da n ta sonni o'qib, ularning yig'indisi (long long) va eng katta elementini chiqaruvchi dastur yozing; int to'lib ketishi mumkinligini izohlang.
21. C++ da vector va sort yordamida n ta sonni saralab, takrorlanmaydigan elementlar sonini chiqaruvchi dastur yozing.
22. C++ da map yordamida so'zlar chastotasini hisoblovchi dastur yozing va Python dagi variant bilan solishtiring.
23. Olimpiada masalasi: n ta butun sondan eng katta va eng kichik elementlar farqini toping ($n \leq 10^5$). Chegaralarni tahlil qiling va Python hamda C++ da yechim yozing.
24. Tkinter yordamida ikki son kiritib, yig'indisini chiqaradigan oyna yozing; noto'g'ri kiritishni qayta ishlang.
25. Turtle yordamida n burchakli muntazam ko'pburchak chizuvchi funksiya yozing va u orqali mandala naqshi hosil qiling.
26. pandas yordamida CSV jadvaldan ballari 80 dan yuqori qatorlarni tanlang, sinf bo'yicha guruhlab o'rtacha ballni toping va ustunli diagramma chizing.

27. NumPy massivida (1 dan 20 gacha sonlar) juft sonlarni ajratib, ularning o'rtachasini toping va vektorlashtirishning ro'yxatdan afzalligini izohlang.
28. Git da yangi repozitoriy yaratish, 2 ta fayl qo'shish, commit qilish, branch ochish va birlashtirish buyruqlari ketma-ketligini yozing va har bir qadamni izohlang.
29. Berilgan funksiya uchun unittest (yoki assert) yordamida kamida 5 ta test yozing, xatoli variantni toping va tuzating.
30. Maktab informatika darsi uchun "Sikllar" mavzusida 3 darajali (A, B, C) masalalar to'plamini (har darajada 2 tadan) tuzing, yechimlari va tipik xatolarni izohlang.

3. Yozma ish variantlari

Variantlar shunday tuzilganki, har bir variantdagi ikki nazariy savol fanning turli bo'limlariga tegishli bo'ladi va tinglovchi amaliy topshiriqni mavzu bo'yicha tanlay olmaydi. Jami 30 ta variant tuzilgan; variant raqami o'qituvchi tomonidan jurnal tartibiga ko'ra yoki qur'a bo'yicha beriladi.

Variant	1-nazariy savol	2-nazariy savol	Amaliy topshiriq
1	1. Dasturlash tillari qanday belgilar bo'yicha tasniflanadi? Paradigma tushunchasini misollar bilan tushuntiring.	38. Vorislik, super() va metodni qayta aniqlashni bayon qiling; "is-a" va "has-a" munosabatlarini solishtiring.	12-amaliy topshiriq
2	2. Dasturlash tillarining rivojlanish bosqichlarini (mashina kodidan zamonaviy tillargacha) bayon qiling.	45. Bog'langan ro'yxat va daraxt tuzilmalarini bayon qiling; ikkilik qidiruv daraxtining xususiyatlarini keltiring.	23-amaliy topshiriq
3	3. Kompilyatsiya va interpretatsiyani taqqoslang; Python dasturi qanday bajarilishini (bayt-kod, virtual mashina) tushuntiring.	52. Hodisaga asoslangan dasturlash va Tkinter asoslarini (vidjetlar, joylashtirish, hodisalar) bayon qiling.	4-amaliy topshiriq
4	4. Python tilining asosiy xususiyatlari va qo'llanish sohalarini bayon qiling; Python ni o'rnatish va muhit tanlash masalalarini yoritng.	59. Kod sifati: PEP 8, DRY, refaktoring va kod sharhi (code review) tushunchalarini tushuntiring.	15-amaliy topshiriq
5	5. IDE, REPL va Jupyter Notebook tushunchalarini tushuntiring; maktab uchun dasturlash muhitini tanlash mezonlarini keltiring.	36. Atribut va metodlar, sinf va ob'yekt atributlari, maxsus (dunder) metodlarni misollar bilan bayon qiling.	26-amaliy topshiriq

Variant	1-nazariy savol	2-nazariy savol	Amaliy topshiriq
6	6. Xato xabarini (traceback) o'qish va boshlovchilarning tipik xatolari haqida gapiring; ularni o'rgatish metodikasini bayon qiling.	43. Tez saralash va birlashtirib saralash algoritmlarining g'oyasini tushuntiring; "bo'lib tashla va hukmronlik qil" tamoyilini bayon qiling.	7-amaliy topshiriq
7	7. O'zgaruvchi va ma'lumot turlari (int, float, str, bool) tushunchalarini tushuntiring; turlarni o'zgartirish va tipik xatolarni sanang.	50. STL konteynerlari (vector, map, set) va sort ni tushuntiring; ularni Python tuzilmalari bilan solishtiring.	18-amaliy topshiriq
8	8. Python da operatorlar (arifmetik, taqqoslash, mantiqiy) va ularning ustuvorligini misollar bilan tushuntiring.	57. Versiyalarni boshqarish, Git ning asosiy buyruqlari, GitHub, branch, pull request va README tushunchalarini bayon qiling.	29-amaliy topshiriq
9	9. input() va print() funksiyalarining imkoniyatlarini, f-string formatlashni tushuntiring.	34. Istisnolar va try/except/else/finally konstruksiyasini bayon qiling; kiritilgan ma'lumotni tekshirish usullarini keltiring.	10-amaliy topshiriq
10	10. Shartli operatorlar (if, elif, else) va ularning ishlash tamoyilini bayon qiling; elif zanjirida shartlar tartibining ahamiyatini tushuntiring.	41. Chiziqli va ikkilik qidiruv algoritmlarini taqqoslang; ikkilik qidiruv uchun talablarni tushuntiring.	21-amaliy topshiriq
11	11. match operatori va chekka holatlarni tekshirish tushunchalarini tushuntiring; shartli operatorni o'qitish metodikasini yoritng.	48. C++ da statik turlar, int to'lib ketishi va kiritish-chiqarishni tushuntiring; Python bilan solishtiring.	2-amaliy topshiriq
12	12. Mantiqiy ifodalar, haqiqiylik jadvallari va murakkab shartlarni tuzish qoidalarini bayon qiling.	55. pandas (Series, DataFrame) va matplotlib bilan ma'lumotlarni qayta ishlash va vizuallashtirishni bayon qiling.	13-amaliy topshiriq
13	13. for va while sikllarini taqqoslang; range() funksiyasi va akkumulyator hamda hisoblagich tushunchalarini tushuntiring.	32. Fayllar bilan ishlash (open, rejimlar, with, encoding) va CSV formati bilan ishlashni bayon qiling.	24-amaliy topshiriq
14	14. break, continue va cheksiz sikl tushunchalarini bayon qiling; kuzatuv jadvalining ahamiyatini tushuntiring.	39. Polimorfizm va abstraksiya tushunchalarini misollar bilan tushuntiring.	5-amaliy topshiriq

Variant	1-nazariy savol	2-nazariy savol	Amaliy topshiriq
15	15. Ichma-ich sikllar va ularning qo'llanishini (naqshlar, jadvallar) tushuntiring; takrorlar sonini baholang.	46. Graf tushunchasi, uni ifodalash usullari hamda BFS va DFS algoritmlari g'oyasini bayon qiling.	16-amaliy topshiriq
16	16. Satrlar, indekslash va kesimlarni bayon qiling; satrning o'zgarmasligini tushuntiring.	53. Turtle grafikasi va uning yordamida sikl, funksiya va rekursiyani o'rgatish metodikasini yoritng.	27-amaliy topshiriq
17	17. Satr metodlarini (split, join, replace, find va boshqalar) sanab, misollar bilan tushuntiring.	60. Sun'iy intellekt yordamchilaridan dasturlashda foydalanishning imkoniyat va xavflarini, akademik halollik qoidalarini bayon qiling.	8-amaliy topshiriq
18	18. Unicode va belgilarni kodlash (ord, chr) haqida gapiring; Sezar shifrini tushuntiring.	37. Inkapsulyatsiya va @property ning ahamiyatini tushuntiring; OYD ni maktabda o'qitish xususiyatlarini yoritng.	19-amaliy topshiriq
19	19. Ro'yxatlar, ularning metodlari va kesimlarini bayon qiling; b = a va nusxalash farqini tushuntiring.	44. Stek va navbat tuzilmalari, ularning amallari va qo'llanishlarini misollar bilan tushuntiring.	30-amaliy topshiriq
20	20. List comprehension va ikki o'lchovli ro'yxatlar (matritsalar) haqida gapiring; tipik xatolarni sanang.	51. Olimpiada masalalarini yechish strategiyasi: shartni tahlil qilish, chegaralar, chekka holatlar va hukmlar (WA, TLE, RE) ni bayon qiling.	11-amaliy topshiriq
21	21. Kortejlar va ularning o'zgarmasligi, ochish (unpacking) tushunchalarini tushuntiring.	58. Testlash (assert, unittest, pytest) va debugging bosqichlarini bayon qiling.	22-amaliy topshiriq
22	22. Lug'atlar tuzilishi, metodlari va chastota lug'atini tuzish usulini bayon qiling.	35. OYD ning g'oyasi, sinf va ob'yekt tushunchalari, __init__ va self ni tushuntiring.	3-amaliy topshiriq
23	23. To'plamlar va to'plam amallarini (birlashma, kesishma, ayirma) tushuntiring; tuzilma tanlash mezonlarini keltiring.	42. Oddiy saralash usullarini (pufakchali, tanlash, qo'shish) bayon qiling va murakkabligini baholang.	14-amaliy topshiriq
24	24. Funksiya yaratish, parametrlar, return va standart qiymatlarni bayon qiling; print va return farqini tushuntiring.	49. C++ da massivlar va funksiyalar (qiymat va havola bo'yicha uzatish) ni bayon qiling.	25-amaliy topshiriq

Variant	1-nazariy savol	2-nazariy savol	Amaliy topshiriq
25	25. Lokal va global ko‘rinish sohasi hamda lambda funksiyalarni tushuntiring.	56. Ma’lumotlar tahlilining asosiy bosqichlarini tushuntiring; maktab o‘quvchilari bilan tahlilda maxfiylikka rioya qilish talablarini keltiring.	6-amaliy topshiriq
26	26. Rekursiya g‘oyasi, bazaviy holat va chaqiruvlar stekini tushuntiring; faktorial va Fibonachchi misollarida rekursiya va siklni solishtiring.	33. JSON formati va json moduli bilan ishlashni tushuntiring; JSON va CSV ni solishtiring.	17-amaliy topshiriq
27	27. Modul va paket tushunchalari, import usullarini bayon qiling; math, random va datetime modullarini tushuntiring.	40. Algoritmalar murakkabligi va O-notatsiyani tushuntiring; asosiy murakkabliklarga misollar keltiring.	28-amaliy topshiriq
28	28. pip, PyPI va virtual muhit (venv) tushunchalarini tushuntiring; o‘z modulini yaratishni bayon qiling.	47. C++ ning xususiyatlari, dastur tuzilishi va kompilyatsiya jarayonini bayon qiling.	9-amaliy topshiriq
29	29. Standart kutubxonaning qo‘shimcha modullari (collections, statistics, time) va ularning qo‘llanishini misollar bilan bayon qiling.	54. NumPy va vektorlashtirish g‘oyasini tushuntiring; ro‘yxat bilan farqini ko‘rsating.	20-amaliy topshiriq
30	30. Python tilida qiymat berish, ob’yektga havola va o‘zgaruvchan hamda o‘zgarmas turlar farqini tushuntiring.	31. Dasturlashni o‘rganishda PRIMM va juft dasturlash metodlarining mohiyatini tushuntiring.	1-amaliy topshiriq

4. Yozma ish variantining namunaviy ko‘rinishi

NAMANGAN DAVLAT PEDAGOGIKA INSTITUTI

“Zamonaviy dasturlash tillari” fanidan yakuniy nazorat yozma ishi

1-VARIANT

Tinglovchi F.I.Sh.: _____ Guruh: _____

Sana: _____

1. Dasturlash tillari qanday belgilar bo‘yicha tasniflanadi? Paradigma tushunchasini misollar bilan tushuntiring.
2. Vorislik, super() va metodni qayta aniqlashni bayon qiling; “is-a” va “has-a” munosabatlarini solishtiring.

3. Talabalar ismi va ballari yozilgan CSV faylni o‘qib, o‘rtacha ball va eng yuqori ball egasini aniqlovchi, natijani JSON ga saqlovchi dastur yozing.

Vaqt: 80 daqiqa. Har bir topshiriq 5 baholik shkalada baholanadi; yozma ish bahosi uchta topshiriq baholarining o‘rtacha arifmetik qiymati sifatida yaxlitlab qo‘yiladi.

5. Yozma ish topshiriqlarini baholash mezonlari

Baho	Nazariy savol	Amaliy topshiriq
5 (a‘lo)	Savol to‘liq va ilmiy to‘g‘ri yoritilgan, atamalar aniq ishlatilgan, misollar keltirilgan, xulosa mantiqiy	Dastur to‘liq va to‘g‘ri ishlaydi, chekka holatlar hisobga olingan, kod tuzilmali va tushuntirilgan, test yoki kuzatuv jadvali keltirilgan
4 (yaxshi)	Savol asosan to‘g‘ri yoritilgan, kichik noaniqliklar bor, misollar yetarli	Dastur asosan to‘g‘ri ishlaydi, ayrim chekka holatlar hisobga olinmagan yoki izoh yetarli emas
3 (qoniqarli)	Savol qisman yoritilgan, muhim jihatlar tushib qolgan, misollar yo‘q yoki noto‘g‘ri	Dastur qisman ishlaydi, mantiqiy xatolar bor, tushuntirish zaif
2 (qoniqarsiz)	Savolga javob berilmagan yoki mazmunan noto‘g‘ri	Dastur yozilmagan yoki ishlamaydi yoki fanga aloqasi yo‘q

6. Fan bo‘yicha tinglovchi bilimini baholash mezonlari (o‘quv dasturiga muvofiq)

a) “5” (a‘lo) baho olish uchun tinglovchining bilim darajasi quyidagilarga javob berishi lozim:

- fanning mohiyati va mazmunini to‘liq yorita olsa;
- fandagi mavzularni bayon qilishda ilmiylik va mantiqiylik saqlanib, ilmiy xatolik va chalkashliklarga yo‘l qo‘ymasa;
- fan bo‘yicha mavzu materiallarining nazariy yoki amaliy ahamiyati haqida aniq tasavvurga ega bo‘lsa;
- fan doirasida mustaqil erkin fikrlash qobiliyatini namoyon eta olsa;
- berilgan savollarga aniq va lo‘nda javob bera olsa;
- konspektga puxta tayyorlangan bo‘lsa;
- mustaqil topshiriqlarni to‘liq va aniq bajargan bo‘lsa;
- barcha amaliy ko‘nikma va malakalarni o‘zlashtirgan bo‘lsa;
- nazariy bilimlarni turli vaziyatda qo‘llay olsa;
- tizimli yondashish, uzviylikka amal qilsa.

b) “4” (yaxshi) baho olish uchun tinglovchining bilim darajasi quyidagilarga javob berishi lozim:

- fanning mohiyati va mazmunini tushungan, fandagi mavzularni bayon qilishda ilmiy va mantiqiy chalkashliklarga yo‘l qo‘ymasa;

- fanning mazmuni va amaliy ahamiyatini tushungan bo'lsa;
- fan bo'yicha berilgan vazifa va topshiriqlarni o'quv dasturi doirasida bajarsa;
- fan bo'yicha berilgan savollarga to'g'ri javob bera olsa;
- fan bo'yicha konspektini puxta shakllantirgan bo'lsa;
- fan bo'yicha mustaqil topshiriqlarni to'liq bajargan bo'lsa;
- barcha amaliy ko'nikma va malakalarni o'zlashtirishga harakat qilgan bo'lsa.

d) "3" (qoniqarli) baho olish uchun tinglovchining bilim darajasi quyidagilarga javob berishi lozim:

- fan haqida umumiy tushunchaga ega bo'lsa;
- fandagi mavzularni tor doirada yoritib, bayon qilishda ayrim chalkashliklarga yo'l qo'yilsa;
- fan bo'yicha mustaqil topshiriqlarni qisman bajargan bo'lsa;
- bayon qilish ravon bo'lmasa;
- fan bo'yicha savollarga mujmal va chalkash javoblar olinsa;
- fan bo'yicha matn puxta shakllantirilmagan bo'lsa.

e) quyidagi hollarda tinglovchining bilim darajasi qoniqarsiz "2" baho bilan baholanishi mumkin:

- fan bo'yicha mashg'ulotlarga tayyorgarlik ko'rilmagan bo'lsa;
- fan bo'yicha mashg'ulotlarga doir hech qanday tasavvurga ega bo'lmasa;
- fan bo'yicha mustaqil topshiriqlarni to'liq bajarmagan bo'lsa;
- fan bo'yicha matnlarni boshqalardan ko'chirib olganligi sezilib tursa (plagiat aniqlansa);
- fan bo'yicha matnda jiddiy xato va chalkashliklarga yo'l qo'yilgan bo'lsa;
- fanga doir berilgan savollarga javob olinmasa;
- fanni bilmasa.

XII. ADABIYOTLAR RO‘YXATI

Asosiy adabiyotlar

1. Lutz M. Learning Python. 5th Edition. – O‘Reilly Media, 2013.
2. Matthes E. Python Crash Course. 3rd Edition. – No Starch Press, 2023.
3. Sebesta R.W. Concepts of Programming Languages. 12th Edition. – Pearson, 2019.
4. Downey A.B. Think Python: How to Think Like a Computer Scientist. 2nd Edition. – O‘Reilly Media, 2015.

Qo‘shimcha adabiyotlar

5. Phillips D. Python 3 Object-Oriented Programming. 3rd Edition. – Packt, 2018.
6. Cormen T.H., Leiserson C.E., Rivest R.L., Stein C. Introduction to Algorithms. 4th Edition. – MIT Press, 2022.
7. Stroustrup B. Programming: Principles and Practice Using C++. 2nd Edition. – Addison-Wesley, 2014.
8. McKinney W. Python for Data Analysis. 3rd Edition. – O‘Reilly Media, 2022.

Axborot manbalari

9. <https://docs.python.org/3/> – Python dasturlash tilining rasmiy hujjatlari.
10. <https://en.cppreference.com> – C++ tili bo‘yicha ma’lumotnoma.
11. <https://docs.github.com> – GitHub rasmiy hujjatlari.
12. <https://visualgo.net> – Algoritmlar va ma’lumotlar tuzilmalarini vizuallashtirish platformasi.
13. <https://codeforces.com> – Dasturlash bo‘yicha olimpiada masalalari platformasi.

Xalqaro universitetlarning tegishli o‘quv manbalari

14. <https://cs50.harvard.edu/python/> – Harvard University. CS50’s Introduction to Programming with Python.
15. <https://ocw.mit.edu/courses/6-0001-introduction-to-computer-science-and-programming-in-python-fall-2016/> – Massachusetts Institute of Technology (MIT). 6.0001 Introduction to Computer Science and Programming in Python.