ҚАЗАҚСТАН РЕСПУБЛИКАСЫ БІЛІМ ЖӘНЕ ҒЫЛЫМ МИНИСТРЛІГІ
«ӨРЛЕУ» БАҰО» АҚФ ТҮРКІСТАН ОБЛЫСЫ ЖӘНЕ
ШЫМКЕНТ ҚАЛАСЫ БОЙЫНША КӘСІБИ ДАМУ ИНСТИТУТЫ

PYTHON

ПРОГРАММАЛАУ ТІЛІНДЕ ТАПСЫРМАЛАРДЫ ОРЫНДАУ ОПЕРАТОРЛАРЫ

Оқу-әдістемелік құрал
Мектеп мұғалімдері мен
6-9 сынып оқушыларына арналған

Шымкент 2022

ҚАЗАҚСТАН РЕСПУБЛИКАСЫ БІЛІМ ЖӘНЕ ҒЫЛЫМ МИНИСТРЛІГІ

«ӨРЛЕУ» БАҰО» АҚФ ТҮРКІСТАН ОБЛЫСЫ ЖӘНЕ ШЫМКЕНТ ҚАЛАСЫ БОЙЫНША КӘСІБИ ДАМУ ИНСТИТУТЫ

PYTHON

ПРОГРАММАЛАУ ТІЛІНДЕ ТАПСЫРМАЛАРДЫ ОРЫНДАУ ОПЕРАТОРЛАРЫ

Оқу-әдістемелік құрал

Мектеп мұғалімдері мен 6-9 сынып оқушыларына арналған

Шымкент
2022

ТҮСІНДІРМЕ ЖАЗБА

Оқу әдістемелік құрал жалпы және негізгі орта білім беру деңгейінің 6-9 сыныптарына арналған "Информатика" пәнінен жаңартылған мазмұны бойынша үлгілік оқу бағдарламасын оқушыларға қосымша білім беруге Python программалау тілін практикалық сабақтарда үйретуге негізделген. Python программалау тілі балалардың танымдық қызығушылықтары мен шығармашылық қабілеттерін дамытуға, олардың интеллектуалдық, моральдық, физикалық қажеттіліктерін қанағаттандыруға ықпал етеді.
Жақсарту: «Python программалау тілі» оқу бағдарламасында оқушыларға АКТ құзыреттілігін дамытуға практикалық бағдарланған.
Python программалау тілінің мақсаты мен міндеттері: Мақсаты дайындықтың кез-келген деңгейіндегі оқушыларға алгоритмдеу мен бағдарламалау саласындағы теориялық және практикалық білімнің жеткілікті көлемін беру, әлеуметтік, мәдени және кәсіби өзін-өзі анықтауға, жеке тұлғаның шығармашылық өзін-өзі жүзеге асыруына жағдай жасау.

Бұл Python программалау тілінің негізгі мақсатын бөліп көрсетуге болады: алгоритмдік және логикалық ойлауды дамыту. Python программалау тілін оқудағы мақсаттылығы мұғалім немесе оқушы бағдарламалау операторлары мен компиляторлық деректерді сақтау мүмкіндіктерінде жинақтар мен жиынтықтарды тиімді жұмыс істеуге нақтылайды.

Бағдарламалау жалпы интеллекті дамытуға ерекше күшті серпін береді және сонымен бірге жұмысқа жағымды эмоционалды әсер береді. Ерекшеліктер, бұл логикалық ойлау дағдыларын дамытуға көмектеседі, сонымен қатар дәл және жүйелі жұмыс дағдысын дамытуға көмектеседі, сондай-ақ дарынды балаларға бағдарламалауды шешудің логикасын жақсырақ қалыптастыруға көмектеседі.

КІРІСПЕ

Мемлекеттік білім стандарттары мен бағдарламаларында информатика пәнінде оқушыларға бағдарламалау және ақпараттық технологияларды үйрету үшін жаңа технологиялық білім беру бағыты ұсынылады. Бағдарлама оқушыларды бағдарламалау негіздерімен таныстырады, сонымен қатар олардың шығармашылық деректерін, логикасын және ойлауын дамытуға көмектеседі.

Кітаптың бұл бөлімінде Python программалау тілінің негізгі тұжырымдамаларымен таныстырады. Бұл тұжырымдамаларда мұғалім немесе оқушы кез келген Python программалау тілін меңгерудің бастапқы айырмашылығын үйрену болып табылады. Мұнда, Сіз әртүрлі типтегі мәліметтерді, есептер жиынтығын, бағдарламалау операторларын, компиляторлық деректерді сақтау мүмкіндіктерімен таныса аласыз. Сонымен қатар, Сіз мәліметтер жиынтығын құруды және сол жиынтықтармен тиімді жұмыс істеуді үйренесіз. Атап айтқанда, егер, және операторлық циклдарда қандай-да бір шарт дұрыс болса, белгілі бір код бөліктерін орындауға мүмкіндік береді, әйтпесе басқа бөліктерді орындайды - бұл құрылымдар процестерді автоматтандыруға өте пайдалы.

Сіз өзіңіздің бағдарламаларыңызды интерактивті ету үшін пайдаланушыдан қалай жауап алу керектігін және пайдаланушы белсенді болғанша оларды орындауды үйренесіз және оқытуды үйрете аласыз. Сонымен қатар, сіз кейбір әрекеттерді бір рет бағдарламалап, содан кейін оны қанша рет қолданатын болсаңыз, бағдарламаларыңыздың кейбір бөліктерін бірнеше рет орындайтын функциялар жазуды үйренесіз. Бұл бағдарламаларда күрделі есептер мен зертханалық жұмыстарды орындай отырып, қарапайым жеңіл бағдарламаларды балаға үйретуге мүмкіндік бересіз. Сіз көптеген жалпы қателіктерді дұрыс өңдейтін бағдарламаларды қалай жазуды үйретесіз.

Python программалау тілі синтаксистері сонымен қатар «таза» код жазуға мүмкіндік береді. Сіздің жазған кодыңызды оңай оқуға болады, және басқа тілдермен салыстырғанда бағдарламаларды жөндеу және кеңейту проблемалары аз болады.

Python программалау тілінде жазылған бағдарламалар әр түрлі мақсаттарда қолданылады: ойындар құру, веб-қосымшалар құру, бизнес мәселелерін шешу және барлық қызықты жобаларға арналған ішкі құралдар. Python программалау тілінде теориялық зерттеулер мен қолданбалы мәселелерді шешу үшін ғылыми салада кеңінен қолданылады.

Бағдарламаның мақсаттары:

PYTHON БАҒДАРЛАМАСЫ

Бағдарламалау ортасын дайындау

Python бағдарламасын қолдану әр түрлі операциялық жүйелерде аздап ерекшеленеді, сондықтан бірнеше нәрсені ескеру қажет. Бұл тарау Python бағдарламасының қазіргі кезде қолданылып жүрген екі негізгі нұсқасымен таныстырады және сіздің жүйеңізде Python бағдарламасын орнатудың негізгі қадамдарын сипаттайды.

Python 2 және Python 3

Қазір Python-нің екі нұсқасы бар: Python 2 және Python 3-тің жаңа бағдарламалау нұсқасы. Әрбір бағдарламалау тілі жаңа идеялармен және технологиялармен дамиды, ал Python бағдарламасын әзірлеушілері тілді қуатты және икемді ету үшін аянбай жұмыс істейді. Python 2-де жазылған код Python 3-тің қолдануымен орнатылған жүйелерде дұрыс жұмыс істемейді. Бұл оқулықта мен Python 2 мен Python 3 бағдарламаларының арасындағы айырмашылықтарды көрсетемін, сондықтан сіз қолданып отырған нұсқаға қарамастан берілген нұсқауларды орындай аласыз.

Егер сіздің жүйеңізде екі нұсқа да орнатылған болса немесе сіз әлі Python орнатпаған болсаңыз, онда Python 3 бағдарламасын қолданыңыз. Егер сіздің жүйеде тек Python 2 орнатылған болса және сіз кодты қондырғыға кедергі келтірмей бірден жазуды жөн көрсеңіз, Python 2-де бастаңыз. Бірақ Python 3-ке қаншалықты тезірек ауыссаңыз, соғұрлым жақсы - ең жаңа нұсқасын пайдалану пайдалы.

Python тілінің ерекшеліктері

Python бірнеше нәрсені орындай алады:

• xml / html файлдарымен жұмыс
• http сұрауларымен жұмыс
• GUI (графикалық интерфейс)
• веб-сценарийлер • FTP-мен жұмыс
• Кескіндермен, аудио және видео файлдармен жұмыс
• Робототехника
• Математикалық және ғылыми есептеулерді бағдарламалау

Басқа көптеген ...

Осылайша, python-да резервтік көшіру, электрондық поштаны оқу мүмкіндіктерін де қолдана аласыз. Python бағдарламалау тілі іс жүзінде шектеусіз, сондықтан оны ірі жобаларда да қолдануға болады. Мысалы, Google және Yandex сияқты IT- мамандар python -ды көп қолданады. Сонымен қатар, python -ның қарапайымдылығы мен әмбебаптығы оны ең жақсы бағдарламалау тілдерінің біріне айналдырады.

Бүгін біз python 3-ті компьютерге қалай жүктеу және орнату туралы әңгімелесеміз.

Бағдарламалауға байланысты мамандықты меңгеруге қызығушылықты қалыптастыруға алгоритмдік мәдениетті қалыптастыруға бағыттайды. Оқушыларға оның құрылымдық нұсқасында алгоритмдеудің білімі мен дағдысын меңгертуді үйретеді.

Python программалау тілінде оқушыға енгізілген есептерді шығарудың барлық әдістерін меңгертуді, алгоритмдік ойлауын дамытуды, бағдарламаны сауатты құрастыру дағдысын қалыптастыруды, бағдарламалау мен алгоритмдік есептерді шешуде білімді, дағды мен дағдыларды тереңдетіп оқытуды қарастырады.

WINDOWS-ка PYTHON ОРНАТУ

Python 3 программасын ресми сайттан жүктейміз, https://python.org/downloads/windows/ сайтына өтіп, «python-нің соңғы нұсқасы» мен python 3 таңдаңыз.

Кейбір бағдарламалар python 2-де жұмыс істемеуі мүмкін. Жазу кезінде бұл python 3.4.1.

Python » Downloads » Windows

Python Releases for Windows

- Latest Python 2 Release - Python 2.7.7
- Latest Python 3 Release - Python 3.4.1

- Python 2.7.7 - June 1, 2014
- Python 3.4.1 - May 19, 2014
- Python 2.7.7rc2 - May 17, 2014
- Python 3.4.1rc1 - May 5, 2014

Python 3 бағдарламасының осы нұсқасын сипаттайтын бет пайда болады. Содан кейін біз парақтың төменгі жағына өтіп, «жүктеу парағын» "download page" ашамыз.

More resources

- Change log for this release.
- Online Documentation
- What's new in 3.4?
- 3.4 Release Schedule
- Report bugs at http://bugs.python.org.
- Help fund Python and its community.

Download

Please proceed to the download page for the download.

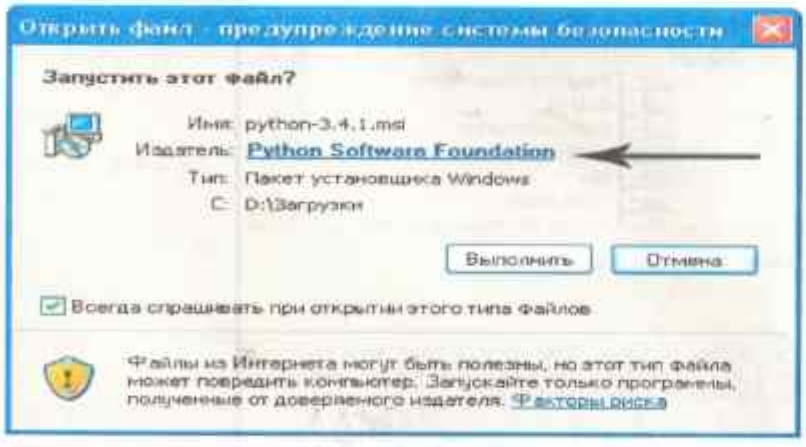
Notes on this release:

- The binaries for AMD64 will also work on processors that implement the Intel 64 architecture. (Also known as the "x64" architecture, and formerly known as both "EM64T" and "x86-64".) They will not work on Intel Itanium Processors (formerly "IA-64").
- There is important information about IDLE, Tkinter, and Tcl/Tk on Mac OS X here.

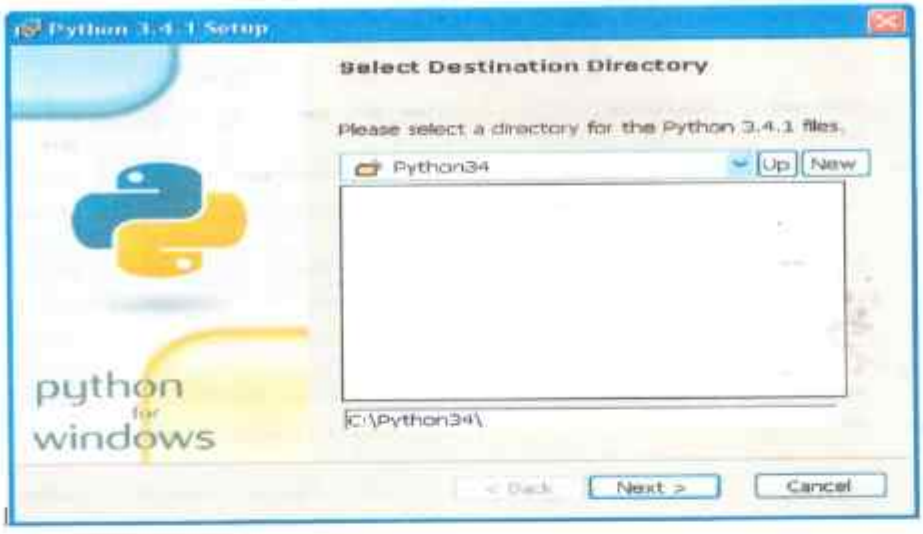
Сіз жүктеуге болатын файлдардың тізімін көресіз. Бізге Windows x86 MSI инсталляторы қажет (егер жүйе 32 бит болса) немесе Windows x86-64 MSI инсталляторы (егер жүйе 64 бит болса). Бізге файлдардан басқа ештеңе қажет емес.

Version	Operating System	Description	Date	MD5 Sum	File Size
Mac OS X 64-bit/32-bit installer	Mac OS X	for Mac OS X 10.6 and later		316a2f81ed6f73bb0cb2c84380e4e2db	22776248
Mac OS X 32-bit i386/PPC installer	Mac OS X	for Mac OS X 10.5 and later		134f84c2f5ad0533f9165b31250b4e959	22692757
XZ compressed source tarball	Source release			6cafc193b410947fdd73d573fdf7f616e	14125708
Gzipped source tarball	Source release			2088540008f78587b26b0b06a63844af6	19113124
Windows debug information files	Windows			9ce29e8356cf13f8ee41f7595c2d7399	367e4364
Windows x86 MSI installer	Windows			4949c3fad31ffa2ca7f9cc43a005b89a	24408064
Windows debug information files for 64-bit binaries	Windows			44a2d4d3c62a147f5a9f7310b1049001	24126218
Windows help file	Windows			6f47ff9368b15d2f30f3c7311ab629e5	7297786
Windows x86-64 MSI installer	Windows	for AMD64/EM64T/x64, not Itanium processors		2544b53f27ee1597fd6b3e15eee1151	25134184

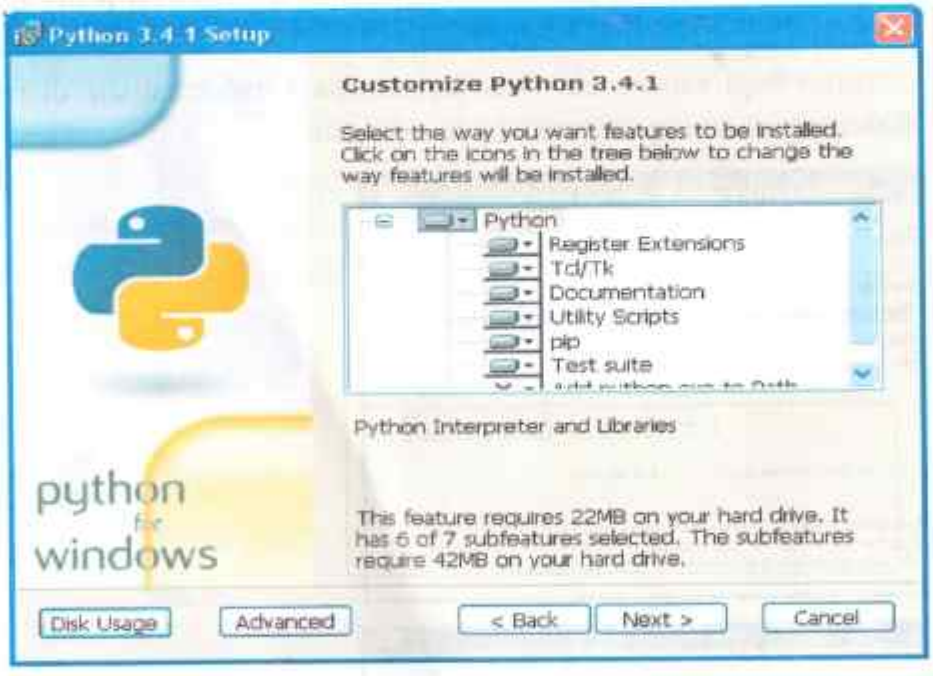
Біз python - ның жүктелуін күтеміз. Содан кейін жүктелген файлды ашыңыз. Файлға *Python Software Foundation* бағдарламалық жасақтама қоры қол қойған, сондықтан бәрі тәртіппен. Осы мүмкіндікті пайдаланып, сізге таныс емес exe файлдарын ашпау керектігін ескертемін.



Орнату үшін біз (папка) қалтаны таңдаймыз. Мен әдепкі (папка) қалтаны қалдырамын. Дискінің кез-келген (папка) қалтасын таңдауға болады.



Орнатылатын компоненттерді таңдаймыз. Егер сенімді болмасаңыз, стандартты компоненттерді қалдырыңыз.



Біз python бағдарламасының орнатылуын күтеміз ...
Finish - Аяқтау. Құттықтаймыз, сіз Python бағдарламасын орнаттыңыз! IDLE ортасы Windows үшін python инсталляторына орнатылған. Дәл қазір сіз өзіңіздің алғашқы бағдарламаңызды жаза аласыз.

САНДАР

Python программалау тілінде сандарды бағдарламалауда ойындарда ұпай ұстау, визуалдау кезінде мәліметтерді ұсыну, ақпаратты веб-қосымшаларда сақтау және тағы басқалар үшін өте жиі қолданылады. Python-да сандық мәліметтер қолданылуына қарай бірнеше санаттарға бөлінеді. Алдымен, Python бүтін сандармен қалай жұмыс істейтінін көрейік, өйткені олар ең аз проблемалы болып табылады.

Сандар: бүтін, нақты, күрделі

Python бағдарламалау тілінде сандар: бүтін сандар, нақты, күрделі болып бөлінеді.

Бүтін сандар

Python-да бүтін сандарға қосу (+), азайту (-), көбейту (*) және бөлуді (/) орындауға болады.

```
>>> 2 + 3
5
>>> 3 - 2
1
>>> 2 * 3
6
>>> 3 / 2
1.5
```

терминал сеансында python жай операцияның нәтижесін қайтарады, python -да дәрежелеуді көрсету үшін қос көбейту белгісін қолданады:

```
>>> 3 ** 2
9
>>> 3 ** 3
27
>>> 10 ** 6
1000000
```

python-да бірнеше операцияларды бір өрнекте қолдануға мүмкіндік беретін белгілі бір жұмыс тәртібі бар. Жақшалар амалдар ретін өзгерту үшін пайдаланылады, осылайша өрнек сіз қалаған тәртіппен бағаланады. мысал:

```
>>> 2 + 3*4
14
>>> (2 + 3) * 4
  20
```

осы мысалдардағы бос орындар python бағдарламасындағы өрнектерді қалай бағалайтынына әсер етпейді, олар сізге кодты оқу кезінде жылдамдықты, операцияларды жылдам табуға көмектеседі.

Python -да тұтынылатын жад көлеміне қарамастан кез келген бүтін сан **int** түрімен ұсынылған. 22 немесе 22222222222222222222222222222222 жазсаңыз да, ол әлі де **int** ретінде анықталады, тек бұл мән сіздің

құрылғыңыздың жадында басқа жадты алады. Басқа программалау тілдерінде бүтін сандар әр түрлі мәліметтер типіне жатады.

Айтпақшы, Python -да түсінікті болу үшін үлкен сандардың астын сызықпен бөлуге болады және олар әлі де **int** деп түсіндіріледі. Мысал: сіз 200000000 жазып, нөлмен шатастыра аласыз, бірақ мұндай опция да бар: 200_000_000 (келісемін, бұл ыңғайлы).

Python сандары компьютердің жадында екілік түрінде бейнеленген және әр түрлі көлемді жадты қабылдай алады. Түсіну маңызды: сан неғұрлым көп болса, онымен операциялар соғұрлым көп уақытты алады және компьютер жады қажет болады.

БҮТІН САНДАР (int) операторы

Python программалау тілінде сандар қарапайым, басқа сандардан айырмашылығы жоқ. Олар ең кең таралған математикалық амалдар жиынтығын қолданады. Кез келген арифметикалық операцияны бүтін сандармен шектеусіз орындауға болады. Олардың 7 түрі бар:

x + y	қосу
x - y	азайту
x * y	көбейту
x / y	бөлімі
x // y	Бөлуден бүтін бөлікті алыңыз
x% y	Бөлудің қалдықтары -x Санның өзгеру белгісі
abs (x)	Сандардың модулі
divmod (x, y)	жұп (x // y, x% y)
x ** y	Көрсеткіш
pow (x, y [, z])	x^y модулі (егер модуль көрсетілген болса)

«Бүтін сандар» тақырыбы бойынша сұрақтар

1. *Бүтін сандар бойынша қандай амал ешқашан бүтін санға әкелмейді?*

Жауап: Қарапайым бөлу әрқашан float операторы қалдықсыз түрін қайтарады, тіпті егерде бөлшек қалдықтары болмаса да (яғни, егер сіз 4/2 бөлсеңіз, сіз 2 емес, 2,0 аласыз).

2. *Бүтін санды жариялау параметрлері бір нәрседен ерекшеленеді ме: 1230 немесе int (1230)?*

Жауап: Олардың айырмашылығы жоқ. Шындығында, егер біз 10 деп жазсақ, int (10) дегенді білдіреді.

3. *Int () функциясы арқылы бүтін санға түрлендіруге болатын барлық деректер түрлерінің тізімін жасаңыз?*

Жауап: Бүтін сандарды қысқартуға болады:

- бүтін сандар: int (555) = 555;
- өзгермелі нүкте сандары: int (-32.45) = -32;

- логикалық мәндер: int (False) = 0;
- ондық бөлшектер: int (Ондық ('17 .7 ')) = 17;
- бөлшек сандар: int (Бөлшек (10, 7)) = 1;
- жолдар: int ('31 ') = 31.

Биттік операциялар түсінігі

Разрядты операцияларды бүтін сандарда да орындауға болады

x | y биттік немесе

x ^ y Bitwise эксклюзивті немесе

x & y биттік және

x << n бит жылжуы солға

x >> y биттің оңға жылжуы

~ x бит инверсиясы

қосымша әдістер

int.bit_length () - таңбаны және жетекші нөлдерді қоспағанда, санды екілік санау түрінде ұсынуға қажетті биттер саны.

```
>>> n = -37 >>> bin(n)
'-0b100101'
>>> n.bit_length ()
```

САНАУ ЖҮЙЕЛЕРІ

Мектепте информатика пәнінде сандарды ондық санау жүйесінде ғана емес кодтық жүйені де ұсынуға болатындығын біледі. Мысалы, компьютерде екілік код қолданылады, мысалы, екілік жүйеде 19 саны 10011 сияқты болады.

Сондай-ақ, кейде сізге сандарды бір санау жүйесінен екіншісіне ауыстыру қажет болады. Python бұл үшін бірнеше функцияларды ұсынады:

• int ([объект], [негіз]) - ондық санау жүйесінде бүтін санға айналдыру. Әдепкі негіз ондық санға тең, бірақ сіз 2-ден 36-ға дейінгі кез-келген негізді қоса аласыз.

• bin (x) - бүтін санды екілік жолға айналдырады.

• hex (x) - бүтін санды он алтылық жолға айналдыру.

• oct (x) - бүтін санды сегіздік жолға айналдыру.

Мысалдар:

```
>>> a = int ('19 ') # жолды санға түрлендіру >>> b = int ('19 .5') # жол бүтін сан емес
Бақылау (соңғы қоңырау соңғы):
File "", line 1, in
ValueError: invalid literal for int() with base 10: '19.5'
>>> c = int (19.5) # өзгермелі нүкте санына қолданылады, бөлшек бөлігін алып тастайды
>>> print(a, c)
19 19
>>> bin(19)
```

```
'0b10011'
>>> oct(19)
'0o23'
>>> hex(19)
'0x13'
>>> 0b10011 # Мұны сандық тұрақтыларды жазу үшін де қолдануға болады
19
>>> int ('10011', 2)
19
>>> int ('0b10011', 2)
19
```

НАҚТЫ САНДАР

Python-да бөлшек бөлігі бар сандар нақты (немесе «өзгермелі нүкте») сандары деп аталады. Әдетте, әзірлеуші бөлшек мәндерді көп ойланбай-ақ қолдана алады. Қажетті сандарды енгізіңіз, сонда Python бағдарламасы сіздің қалағаныңызды орындай алады:

```
>>> 0.1 + 0.1
0.2
>>> 0.2 + 0.2
0.4
>>> 2 * 0.1
0.2
>>> 2 * 0.2
  0.4
```

Алайда, кейбір жағдайларда, күтпеген жерден нәтижеде бөлшек бөлікте күтпеген жерден үлкен цифрлар болады:

```
>>> 0.2 + 0.1
0.30000000000000004
>>> 3 * 0.1
0.30000000000000004
```

Мұндай нәрсе кез-келген тілде болуы мүмкін; алаңдауға негіз жоқ. Python нәтижені дәл көрсетуге тырысады, бұл кейде компьютерлердегі сандардың ішкі көрінісі сипатына байланысты қиынға соғады. Әзірге тек «қосымша» цифрларды елемеңіз; Осындай жағдайларды қалай шешуге болатынын бөлім жобаларынан білетін боласыз.

Нақты сандар (float) операторы

Нақты сандар бүтін сандармен бірдей амалдарды қолдайды. Алайда (сандардың компьютерде көрсетілуіне байланысты) нақты сандар нақты емес, және бұл қателіктерге әкелуі мүмкін:

```
>>> 0,1 + 0,1 + 0,1 + 0,1 + 0,1 + 0,1 + 0,1 + 0,1 + 0,1 + 0,1
```

0.9999999999999999
Жоғары дәлдік үшін басқа заттарды қолданыңыз (мысалы, Decimal и Fraction)) – (ондық және бөлшек).
Сондай-ақ, нақты сандар ұзақ арифметиканы қолдамайды:
>>> a = 3 ** 1000
>>> a + 0,1
Traceback (most recent call last): - Бақылау (соңғы қоңырау соңғы):
File "", line 1, in
OverflowError: int too large to convert to float
Сандармен жұмыс істеудің қарапайым мысалдары:
>>> c = 150
>>> d = 12.9
>>> c + d
162.9
>>> p = abs (d - c) # Санның модулі
>>> print(p)
137.1
>>> round(p) # дөңгелектеу
137

Қосымша әдістер
float.as_integer_ratio() - қатынасы осы санға тең бүтін сандар жұбы.
float.is_integer() - мән бүтін бола ма.
float.hex() - at сұлы алтылыққа айналдырады (он алтылық белгі). classmethod
float.fromhex(s) - алты қырлы жіптен алынған сұлы.
>>> (10.5).hex()
'0x1.5000000000000p + 3'
>>> float.fromhex ('0x1.5000000000000p + 3')
10.5

Сандармен жұмыс істеуге арналған стандартты өрнектерден басқа (Python-да олардың саны онша көп емес), Python-да бірнеше пайдалы модульдер бар.
Модуль *math* - күрделі математикалық функцияларды ұсынады.
>>> import math - математиканы импорттау
>>> math.pi
3.141592653589793
>>> math.sqrt (85)
9.219544457292887
Модуль *random* кездейсоқ сандар генераторы мен кездейсоқ таңдау функцияларын жүзеге асырады.
>>> import random - кездейсоқ импорттау
>>> random.random()
0.15651968855132303

«Нақты сандар» тақырыбына тапсырмалар.

Тапсырма 1. Негізгі деңгей

Шарт

Өрнектің амалдармен бағалану ретін көрсетіңіз:

11 * 2 ** 2 - 13/4 + 7.

Соңғы бүтін сан дегеніміз не?

Біріншіден, біз дәрежеге деін басымдық береміз (2 ** 2). Содан кейін біз көбейтеміз (11 * 4). Содан кейін біз бөлеміз (13/4). Содан кейін алып тастаймыз (44 - 3.25). Соңғы қадамда (40.75 + 7) қосыңыз. Біз өзгермелі нүкте нөмірін аламыз 47.75. Егер сіз оны int түріне шығарсаңыз, нәтиже 47 болады.

Шешім - интерактивті режим

```
>>> 11 * 2 ** 2 - 13 / 4 + 7

47.75
```

Тапсырма 2. Негізгі деңгей

Шарт

3 ** 9090001 нөмірі қанша мегабайт жадыға ие?

Бұл мәселені шешу үшін sys модулінен getizeof() функциясын қолданыңыз. Асықпаңыз. Өрнекті бағалау үшін шамамен 10 секунд күтуге тура келеді (компьютердің қуатына байланысты). Өздеріңіз білетіндей, бір килобайтта - 1024 байт, ал бір мегабайтта - 1024 Кбайт. Жад көлемін есептеу үшін біз sys модулінен getizeof() функциясын қолданамыз. Біз аламыз: sys.getsizeof (3 ** 9090001) / (1024 * 1024) = шамамен 1,83 МБ.

IDE- шешім

```
# 1. Функцияны модульден импорттау
from sys import getsizeof
# 2. Мегабайтта орын алатын жад көлемін табу
getsizeof(3 ** 9090001) / (1024 * 1024)
# Нәтиже: 1.8319969177246094
```

Тапсырма 3. Негізгі деңгей

Шарт

Стандартты арифметикалық амалдарды қолдана отырып, 4, 4, 4 цифрларының ең үлкен бүтін санын көрсетіңіз және оның мәнін беріңіз.

Экспоненциалдау операциясы өте үлкен сан береді, оның мәні белгілі ғаламдағы атомдардан үлкен.

Шешім - интерактивті режим

```
>>> 4 ** 4 ** 4
13407807929942597099574024998205846127479365820592393377723561443721764030073546976801874298166903427690031858186486050853753882811946569946433649006084096
```

Тапсырма 4. Негізгі деңгей

Шарт

Ұсынылған өрнектердің қайсысын бір амалмен бүтін ондық санға түрлендіруге болады:

A) '123e';

B) '91 .4 ';

B) 524.345 ** 4353453453111145345;

D) '7.1 + 4';

E) '4' - 2;

E) '4 - 2';

F) '42'

H) -12.12?

Жауап алу үшін int() функциясындағы өрнектердің әрқайсысын ойша орындау керек. Іс жүзінде бұлай жасау ұсынылмайды, себебі «В» нүктесінде сіздің дербес компьютеріңіз тығыз ілінеді.

А) Жолды түрлендіру мүмкін емес, себебі онда ондық санау жүйесінде сандық өрнегі жоқ алфавиттік таңба бар.

Б) int түріне өзгермелі нүкте санына еліктейтін жолды түрлендіру мүмкін емес.

Шешім - интерактивті режим

```
>>> int ('91 .4 ')
```

Құндылық қатесін ValueError алыңыз

В) Санның ақылға сыймайтын өлшемдері болса да, жады жеткілікті, ол int түріне айналады;

Шешім - интерактивті режим

```
>>> int (524.345 ** 4353453453111145345)
```

Дәреже алдымен есептеледі, содан кейін бүтін санға түрлендіреді

Г) Бұл бүтін санға азайтылмайтын жол.

Шешім - интерактивті режим

```
>>> int ('7.1 + 4')
```

Құндылық қатесін ValueError алыңыз

Д, Е) Жол мен санды немесе жол ішіндегі өрнекті бүтін санға айналдыру мүмкін емес.

Шешім - интерактивті режим

```
>>> int ('4' - 2)
```

TypeError қатесін алу

```
>>> int ('4 - 2')
```

ValueError қатесін алыңыз

Ж) Егер жолдың ішінде тек сандар болса, Python олардан оңай int жасайды

Шешім - интерактивті режим

```
>>> int ('42 ')
42
```

З) Жылжымалы нүкте саны бөлшек қалдығын кесу арқылы бүтін санға айналады.

Шешім - интерактивті режим
>>> int (-12.12)
-12
Сондықтан дұрыс жауаптар: В, Ж, З.

Тапсырма 5. Негізгі деңгей
Шарт
Екі бүтін санды қосудың оң мәнін қайтаратын pos_add(a, b) функциясын жазыңыз.
Біз аргументтердің дұрыстығын тексермейміз, себебі шарт тек бүтін сандарды беретінімізді көрсетеді.
Ең жақсы нұсқа-оң санды қайтаратын abs() функциясын қолдану.
IDE - шешім
```
def pos_add(a, b):
  return abs(a + b)
# Тесты
print(pos_add(7, -3))
print(pos_add(7, -7))
print(pos_add(-2, -3))
```
Орындау нәтижесі
```
4
0
5
```

Тапсырма 6. Негізгі деңгей
Шарт
-11 бойынша кез келген бүтін санның бүтін және модульдік бөлінісінің нәтижесін қайтаратын foo(a) функциясын анықтаңыз.
Бүтін бөлу - «//» операциясының нәтижесі, ал модульдік бөлу - «%» операциясы. Python-да divmod() функциясы бар, ол сан мен бөлгішті қабылдайды. Ол жұп сандарды қайтарады: бүтін санның нәтижесі және бөлудің қалған бөлігі. Бізге «-11» бөлгіш берілген, сондықтан нәтиже келесідей:
IDE - шешім
```
def foo(a):
  return divmod(a, -11)
# Тесттер
print(foo(22))
print(foo(-77))
print(foo(1))
```
Орындау нәтижесі
```
(-2, 0)
(7, 0)
(-1, -10)
```

КҮРДЕЛІ САНДАР (COMPLEX)

Python-да күрделі сандар енгізілген:

```
>>> x = complex (1, 2)
>>> print(x)
 (1 + 2j)
>>> y = complex(3, 4)
>>> print (y)
(3 + 4j)
>>> z = x + y >>> print (x)
(1 + 2j)
>>> print (z)
(4 + 6j)
>>> z = x * y >>> print (z)
(-5 + 10j)
>>> z = x / y >>> print (z)
(0,44 + 0,08j)
>>> print(x.conjugate()) # Біріктірілген сан
(1-2j)
>>> print(x.imag) # Елестететін бөлім
2.0
>>> print(x.real) # Нақты бөлігі
1.0
>>> print(x > y) # Күрделі сандарды салыстыру мүмкін емес
Traceback (most recent call last):   - Бақылау (соңғы қоңырау):
File "", line 1, in
TypeError: реттелмейтін типтер: complex ()> complex ()
>>> print(x == y) # Бірақ сіз теңдікті тексере аласыз
False  - Жалған
>>> abs(3 + 4j) # Күрделі сан модулі
5.0
>>> pow(3 + 4j, 2) #Көрсеткіш
(-7 + 24j)
```

Cmath модулі күрделі сандармен жұмыс істеу үшін де қолданылады.

«Күрделі сандар» тақырыбына тапсырмалар.

1 -тапсырма. * Жетілдірілген

Шарт

Кез-келген мәнді қабылдай алатын num_sum (a) функциясын жазыңыз.
Егер ол бүтін сан болса, онда оның сандарының қосындысын қайтар.
Әйтпесе, «Бұл бүтін сан емес» деген тіркес қайтарылады.
Шешім: теріс санды оңға айналдыратын және аргумент int екенін тексеретін abs () функциясын қажет етеді.

Негізгі мәселе - бұл логикалық мәндер, олар (бәрі біле бермейді) Python ішіндегі int класына жатады. Мәселенің шарты бойынша бізге тек бүтін сандар сәйкес келеді, ал True немесе False сәйкес келмейді. Бұл тексеру де жүргізілуі керек.

IDE - шешім

```
def num_sum(a):
  # 1. 'a' мәнінің бүтін сан екенін, бірақ логикалық емес екенін анықтаңыз
if isinstance(a, int) and not isinstance(a, bool):
  # 2. Санды оңға айналдырыңыз, содан кейін - жол
a_to_str = str(abs(a))
  # 3. Бастапқы қосындыны 0 -ге орнатыңыз
     s = 0
  # 4. Барлық сандарды циклге қосыңыз
     for i in a_to_str:
        s += int(i)
     return s
  return 'Бұл бүтін сан емес'
# Тесттер
print(num_sum(-146))
print(num_sum('-11'))
print(num_sum(True))
```

Орындау нәтижесі:

```
11
Бұл бүтін сан емес
Бұл бүтін сан емес
```

2 -тапсырма. Жетілдірілген

Шарт

Сізге кез келген ұзындықтағы кездейсоқ сандар тізбегі және нөлден үлкен «сиқырлы» оң сан беріледі.

Осы аргументтерді алатын сиқырлы () функциясын жазыңыз және тізбектің квадраттарының қосындысын сиқырлы санға қалдықсыз бөлуге болатынын көріңіз.

Жауап сәттілікке «Сиқырлық болады» немесе бөлуге болмайтын жағдайда «Сиқыр болмайды» қайтарады.

Тізбектің өлшемі белгісіз болғандықтан, біз «args» параметрін «бірінші» параметрі ретінде, сондай -ақ қажетті k параметрі k параметрімен жібереміз.

Цикл, map() функциясын немесе тізімді қосу арқылы квадраттардың қосындысын табу және оның қалдықсыз k -ге бөлінетінін анықтау қажет.

IDE - шешім

```
def magic(*args, k):
  # 1. Бастапқы қосынды нөлге тең
  sq_sum = 0
  # 2. Барлық аргументтердің квадраттарын циклге қосыңыз
```

```
    for i in args:
        sq_sum += i ** 2
    # 3. Sq_sum -ды k -ге бөлудің қалдықтары нолге тең екенін анықтаңыз
    if sq_sum % k == 0:
        return «Сиқырлық болады»
    return 'Сиқыр жоқ'
# Тесттер
print(magic(2, 5, 7, k=5))
print(magic(2, 5, 7, k=39))
print(magic(2, 5, 7, k=2))
```

Орндау нәтижесі:

'Сиқыр жоқ'
«Сиқырлық болады»
«Сиқырлық болады»

ЖЫЛЖЫМАЛЫ НҮКТЕ САНДАРЫ

Жылжымалы нүкте операциялары қате тудыруы және күтпеген нәтиже беруі мүмкін. **float** түрімен жұмыс істеу ерекше назар аударуды қажет етеді.

Сондай -ақ, математикада иррационал сандар болғанына қарамастан (мысалы, Pi, екеуінің квадрат түбірі) мұндай сандардың көрсетілуі ақырлы. Тақырыпты зерттегенде мыналарға назар аудару қажет:

Өзгермелі нүкте санының құрылымы (оның қандай бөліктері жадыда сақталады және әрқайсысы қанша бит алады);

Float түрі конструкторы;

Бүтін сандарға түрлендіру әдістері (**trunc**, **floor**, **ceiling**, **round**);

Дөңгелектеу ережелері.

Егер сіз өзгермелі нүктелік сандарды қолданудан аулақ бола алсаңыз, олардың артықшылықтарын пайдаланған жөн, өйткені олар есептеулерде математикалық тұрғыдан жалған нәтиже бере алады. Егер сізге нақты жиынтық қажет болса, кіріктірілген модульді ондық **decimal** (ондық сандар) немесе бөлшектерді **fractions** (бөлшектер) қолданыңыз.

«Жылжымалы нүкте сандары» тақырыбы бойынша сұрақтар

1. Өрнекті бағалау нәтижесінде қандай деректер түрі алынады:

3 + 3.0 + 4?

Жауап: Алдымен біз өрнекті бағалаймыз, содан кейін type() функциясының көмегімен оның түрін анықтаймыз.

Шешім - интерактивті режим

```
>>> 3 + 3.0 + 4
10.0
>>> type(10.0)
```

<class 'float'>

Көріп отырғаныңыздай, алынған түрі float, яғни. өзгермелі нүкте нөмірі.

2. float (қалқымалы) операторы деректер түрі қандай үш элементтен тұрады?

Жауап: float түрдегі сандар мыналардан тұрады:

- белгісі (+ немесе -);
- ондық дәрежені білдіретін көрсеткіш (мысалы, e + 5);
- елеулі сандар (ондық бөлшекке дейін және кейін).

-11.03e + 5 -нақты санның типтік белгісі. Ол -11.03 * 10 ** 5 немесе -1103000.0 опциясына ұқсас.

3. Кез келген «a» өзгермелі санына int(a) операциясын қолданудың нәтижесі қандай?

Жауап: Нәтиже әрқашан бүтін сан болады. Белгі сақталады, бүтін бөлігі қалады, ал бөлшектің барлық бөлігі кесіледі.

Мысалдар - Интерактивті режим

```
>>> int (3.93)
3
>>> type(int(3.93))
<class 'int'>
>>> int(-11.328)
-11
>>> type(int(-11.328))
<class 'int'>
```

«Жылжымалы нүкте сандары» тақырыбына тапсырмалар.

Тапсырма 1. Негізгі деңгей

Шарт

Кез келген санды өзгермелі нүктеге түрлендіретін to_float (num) функциясын жазамыз.

Егер аргумент ретінде басқа деректер түрі берілсе, ол «Түрлендіру мүмкін емес» дегенді қайтарады.

Int немесе float түріне жататындығын дәлелдеу үшін тексеру қажет. Егер солай болса, онда біз трансформацияны жүзеге асырамыз. Әйтпесе, «Түрлендіру мүмкін емес» дегенді қайтарамыз. Логикалық мәндерді өзгермеліге де float түрлендіруге болатынын ескерген жөн.

Шешім – IDE

```
def to_float(num):
    if isinstance(num, (int, float)):
        return float(num)
    return «Түрлендіру мүмкін емес»
# Тесттер
print(to_float(12))
print(to_float(-1.762))
```

```
print(to_float(True))
print(to_float('Сан емес'))
print(to_float('2.2'))
```
Орындау нәтижесі
```
12.0
-1.762
1.0
«Түрлендіру мүмкін емес»
«Түрлендіру мүмкін емес»
```

Тапсырма 2. Негізгі деңгей

Шарт

4 сан беріледі.

Сізге аргументтердің арифметикалық ортасын қайтаратын және оны 5 ондық бөлшекке айналдыратын avg_5 (a, b, c, d) функциясын жазу керек.

Дөңгелектеу үшін біз кіріктірілген round() функциясын қолданамыз. Сандарды қосу және оларды 4 -ке бөлу қажет.

IDE - шешім
```
def avg_5(a, b, c, d):
    return round((a + b + c + d) / 4, 5)
# Тесттер
print(avg_5(1, 6, 7, 4))
print(avg_5(1.7, 6.2, 2, 6))
print(avg_5(3, -3.143223442, -4.76, 1.3902))
```
Орындау нәтижесі
```
4.5
3.975
-0.87826
```

Тапсырма 3. Негізгі деңгей

Шарт

Mul_to_int (a, b) функциясы бүтін сандарды немесе нақты сандарды қабылдай алады.

Егер аргументтерді көбейту нәтижесінде ондық бөлшектен кейін маңызды үтір сандар болмаса, онда ол оны бүтін сан ретінде қайтарады.

Әйтпесе, float қалқымалы ретінде.

Көбейтудің нәтижесін int түріне тексеру керек. Сонымен қатар, int класында is_integer () әдісі жоқ екенін ескеру қажет.

IDE - шешім
```
def mul_to_int(a, b):
    res = a * b
    if float(res).is_integer():
        return int(res)
    return res
```

```
# Тесттер
print(mul_to_int(2, 4))
print(mul_to_int(2.5, 4))
print(mul_to_int(2.2, 2))
```
Орындау нәтижесі
```
8
10
4.4
```

Тапсырма 4. Негізгі деңгей

Шарт

Шардың көлемі X текше метрді құрайды. бірліктер

Пішіннің радиусын табыңыз.

Доптың көлемінің формуласын қолданайық:

V = 4/3 * r ** 3 * pi

Осыдан:

r = ((3 * V / (4 * pi)) ** (1/3))

Интерактивті режим - шешімі

```
>>> from math import pi
>>> (3 * 36 / (4 * pi)) ** (1/3)
2.048352189765887
>>> (3 * 1 / (4 * pi)) ** (1/3)
0.6203504908994001
>>> (3 * 19.32 / (4 * pi)) ** (1/3)
1.6645857441456702
```

Тапсырма 5. Қосымша деңгей

Шарт

Айналмалы нүкте санын алатын және оны орта мектеп математикасының ережелеріне сәйкес бүтін санға дейін дөңгелектейтін round_standard (num) дөңгелектеу функциясын жазыңыз.

Мәселені шешкенде санның белгісін жіберіп алмау маңызды. Тұтастай алғанда, мұндай тапсырма банкті дөңгелектеу ережесін болдырмау үшін бастапқы параметрге 0,5 қосу арқылы шешіледі.

IDE – шешім

```
def round_standard(num):
    if num >= 0:
        sign = 1
    else:
        sign = -1
    return sign * int((abs(num) + 0.5))
# Тесттер
print(round_standart(1.5))
print(round_standart(-2.5))
```

```
print(round_standart(1.6))
print(round_standart(5.11))
```
Орындау нәтижесі
```
2
-3
2
5
```
Тапсырма 6. Жоғары деңгей

Шарт

Python -да нақты сандармен жасалатын операциялар күтпеген нәтиже бере алатындықтан (атап айтқанда 0,1 + 0,2 дәл 0,3 -ке тең болмайды), міндет - бұны қандай да бір жолмен жеңу.

Сіз 3 саннан тұратын eqv (a, b, c) функциясын жазғыңыз келеді.

a және b сандары қосылады.

Бұл сома белгілі бір дәлдікпен «c» санымен салыстырылады.

Дәлдік a және b сандарының үлкенінен 0,01% құрайды.

Егер теңдік қанағаттандырылса, функция True қайтарады, әйтпесе False.

Біз Python -да 0.1 + 0.2! = 0.3, математика ережелері бойынша өрнектің екі бөлігі де бірдей екенін білдік. Демек, жартысын белгілі бір минималды ауытқумен салыстыру қажет «e», оның мәні шартта 0,01%ретінде анықталды. Егер бөліктер «e» -ден кіші мәнмен ерекшеленсе, онда олар тең деп саналады.

IDE - шешім
```
def eqv(a, b, c):
    res = a + b
    e = 0.01 / 100  # Пайыздарды аудару түрлендіру
    tolerance = e * max(abs(a), abs(b))  # Ауытқу мәнін табыңыз
    return abs(res - c) <= tolerance  # Айырмашылық ауытқудан аз екенін анықтаңыз
# Тесттер
print(eqv(0.12, 0.31, 0.43))
print(eqv(0.1, 0.2, 0.3))
print(eqv(0.1, 0.2, 0.4))
print(eqv(-0.1, -0.2, -0.3))
```
Орындау нәтижесі
```
True
True
False
True
```

Кірістірілген **math** математикалық модульді қолдану арқылы жетілдірілген шешім. Isclose() әдісі салыстырмалы айырмашылықты ғана емес (бөлшек түрінде) қабылдай алады, сонымен қатар абсолютті (кейбір жағдайларда екі мән де қажет болуы мүмкін) қабылдай алады.

IDE -шешім

```
def eqv(a, b, c):
    from math import isclose
    return isclose(a + b, c, rel_tol=0.01 / 100, abs_tol=0)
# Тест
print(eqv(0.12, 0.31, 0.43))
print(eqv(0.1, 0.2, 0.3))
print(eqv(0.1, 0.2, 0.4))
print(eqv(-0.1, -0.2, -0.3))
```

Орындау нәтижесі

```
True
True
False
True
```

ЛОГИКАЛЫҚ МӘЛІМЕТТЕР ТИПІ

Логикалық деректер түрінің 2 мәні бар: True (ақиқат) және False (жалған) операторлары. Бұл түрдің толық сипаттамасы PEP-285 стандартында берілген.

Bool класы - int түрінің ішкі сыныбы (яғни бүтін сандар). True және False - бұл бағдарламаның орындалуы кезінде жадыдағы орнын ешқашан өзгертпейтін жеке объектілер.

Математикалық амалдарды логикалық объектілерге қолдануға болады. Логикалық операторлардың 3 түрі бар: not («емес»), or («немесе»), and («және»).

Мәліметтердің логикалық типі бар операциялардың келесі қасиеттері бар: коммутативтілік, үлестіру, ассоциативтілік; де Морган ережесі оларға қатысты.

«Логикалық мәліметтер типі» тақырыбы бойынша сұрақтар.

1. Өрнектердің айырмашылығы неде: True == 1 және True 1? Оларды есептеу кезінде біз қандай нәтиже аламыз?

Жауабы: True == 1 деп жазғанда, біз мәндерді салыстырамыз. Bool класы бүтін сандардың ішкі сыныбы болғандықтан, бұл жағдайда True бірлік болады.

True is 1 өрнегі берілгенде, жадтағы адрестер салыстырылады. Логикалық деректер синглондық объектілер болғандықтан, олардың жадында өздерінің адрестері болады, ол сценарийді орындау кезінде өзгермейді. Сондықтан 1 саны мен True - есте сақтаудың әр түрлі аймақтарын алады.

Мысал - интерактивті режим

```
>>> True == 1
True
```

>>> True is 1 # Жад ұяшықтары салыстырылады
False
>>> id(True), id(1)
(2078150976, 2078275504) # Көріп отырғаныңыздай - жадтағы әртүрлі мекенжайлар

2. Логикалық өрнектерді бөлудің мәні неде?
Жауабы: Бөлу мүмкіндігі логикалық өрнектердегі жақшалардың кеңеюімен байланысты. Ол келесі ережелермен анықталады:
A and (B or C) == (A and B) or (A and C)
A or (B and C) == (A or B) and (A or C)

Мысал – интерактивті режим
>>> True and (False or True) == (True and False) or (True and True)
True
>>> True or (False and True) == (True or False) and (True or True)
True

3. Егер сіз немесе логикалық операторының көмегімен 2 логикалық объектіні салыстырсаңыз, нәтижесінде қандай мәндерді алуға болады? Жауабын суреттеңіз.
Жауабы: Екі логикалық мәнді немесе операторымен салыстыру кезінде 4 вариантты жауап алуға болады. Тек бір жағдайда ол False (жалған) болады.
Мысал – интерактивті режим
>>> True or True
True
>>> False or True
True
>>> False or False
False
>>> True or False
True

4. Python -да қандай объектілер әрқашан False логикалық мәнін қайтарады?
Жауабы: Python-дағы көптеген нысандарға bool() функциясын қолдансақ, біз True аламыз. Барлық ықтимал ерекшеліктер төменде келтірілген:
1. False
Мысал – интерактивті режим
>>> bool(False)
False

2. None
Мысал – интерактивті режим
>>> bool(None)

False
3. 0
Мысал – интерактивті режим
>>> bool(0)
False

4. Бос тізбектер (жолдар, тізімдер, кортеждер және т.б.)
Мысал – интерактивті режим
>>> bool('')
False
>>> bool([])
False
>>> bool({})
False

Сондай-ақ, False қайтару мәні бар __bool__() әдісін кез келген сыныпқа енгізуге болады.

5. Неліктен «a and not b or not c and d or e»? сияқты өрнектерден аулақ болу керек? Оқуға ыңғайлы болу үшін оны қалай өзгертуге болады?
Жауабы: Компьютер одан да күрделі өрнектерді орындауға қабілетті. Бірақ мұндай кодты оқитын адам үшін мәндерді есептеу тәртібі қандай екенін түсіну өте қиын болады.
Егер программаны жазу кезінде мұндай конструкцияны қолдануға тура келсе, онда ол орындалу реті бойынша жақшаға оралуы керек. Бұл өрнекті түсінуді айтарлықтай жеңілдетеді:
((a and not b) or (not c and d)) or e.
Өрнектер тұлғасын дәлелдейтін мысал келтірейік.
Мысал – интерактивті режим
>>> 1 and not 2 or not 3 and 0 or 4
4
>>> ((1 and not 2) or (not 3 and 0)) or 4
4

6. Өрнектердің айырмашылығын түсіндіріңіз: "a and b" и "bool(a) and bool(b)".
Жауабы: bool (a) және bool (b) өрнегі әрқашан логикалық мәнді қайтарады (яғни, True немесе False).
a және b өрнегі соңғы ақиқат мәнді немесе бірінші жалған мәнді қайтарады (қалған өрнектер бұл жағдайда тіпті бағаланбайды).
Мысал – интерактивті режим
>>> bool(7) and bool('Программалау')
True
>>> bool(7) and bool('')

```
False
>>> 7 and [ ]
[ ]
>>> 7 and 44
44
>>> 1 and [ ]
[ ]
>>> None and 11
None
```

7. Логикалық операторларды қолдану кезіндегі операциялардың басымдылығын анықтаңыз.

Жауабы: Логикалық операциялар осы басымдықпен солдан оңға қарай орындалады:

- ең алдымен not операторы бар өрнектер есептеледі;
- онда and операторы бар барлық конструкциялар есептеледі;
- соңғы кезекте біз or операторымен жұмыс істейміз.

Мұнда a or b and not c: өрнегін бағалаудың мысалы келтірілген:

2 or 3 and not 1.

Біріншіден, біз 1 емес операцияның нәтижесін анықтаймыз: False.

Әрі қарай, 3 және False мәндерін салыстырыңыз: 3 and False = False.

Соңғы қадам: 2 or False = 2.

Мысал – интерактивті режим

```
>>> 2 or 3 and not 1
2
```

«Логикалық мәліметтер типі» тақырыбына тапсырмалар

1-Тапсырма. Негізгі деңгей

Шарт

dislike_6 (a) функциясын жазыңыз, ол әрқашан True мәнін қайтарады, егер сіз int немесе float мәнін бермесеңіз (бұл жағдайда ол «6 емес!» қайтарады). Біріншіден, аргументтің бүтін немесе нақты сан екенін тексереміз. Егер солай болса және ол 6 немесе 6,0 тең болса, біз «6 емес!» деп қайтарамыз. Барлық басқа жағдайларда нәтиже True ретінде көрсетіледі.

IDE - шешім

```
def dislike_6(a):
    if (type(a) is float or type(a) is int) and a == 6.0:
        return 'Тек 6 емес!'
    return True
# Тесттер
print(dislike_6(6.0))
```

print(dislike_6(6))
print(dislike_6('6'))
print(dislike_6('Жақсы'))
print(dislike_6([6, 6]))
Орндау нәтижесі:
Тек 6 емес!
Тек 6 емес!
True
True
True

2-Тапсырма. Негізгі деңгей

Шарт
Python тілін оқитын студент логикалық операциялардың қасиеттері (ассоциативтілік, дистрибутивтілік, коммутативтілік, де Морган ережесі) туралы үнемі шатастырады.
Ол 4 әріптің бірін қабылдайтын help_bool (Тапсырма 2. Негізгі деңгей
Шарт
Python тілін оқитын студент логикалық операциялардың қасиеттері (ассоциативтілік, дистрибутивтілік, коммутативтілік, де Морган ережесі) туралы үнемі шатастырады.
Ол 4 әріптің бірін қабылдайтын help_bool (letter) анықтама функциясын жазуды шешті: k, a, d, m (әр қасиетке сәйкес).
Орындалу нәтижесі: жол түріндегі белгілі бір әрекет ережесі.
Басқа бірдеңе жіберілсе, әрбір ықтимал аргументтің түсіндірмесі бар сұрау қайтарылады.
Әрбір ережені сипаттайық:
- коммутативтілік (k)) анықтама функциясын жазуды шешті: k, a, d, m (әр қасиетке сәйкес).
Орындалу нәтижесі: жол түріндегі белгілі бір әрекет ережесі.
Басқа бірдеңе жіберілсе, әрбір ықтимал аргументтің түсіндірмесі бар сұрау қайтарылады.
Әрбір ережені сипаттайық:
- коммутативтілік (k)
A or B = B or A
A and B = B and A

Бөлу мүмкіндігі (d)
A and (B or C) == (A and B) or (A and C)
A or (B and C) == (A or B) and (A or C)

Ассоциативтілік (a)
A or (B or C) == (A or B) or C == A or B or C
A and (B and C) == (A and B) and C == A and B and C