Әдістемелік жұмыс

Тақырыбы : «Phyton бағдарламалау тіліндегі массивтердің түрлері»

ОРЫНДАДЫ: Ирискулов Жалолиддин

.

Түркістан 2023

**Мазмұны**

КІРІСПЕ ................................................................................................3

НЕГІЗГІ БӨЛІМ

1. Массивтер туралы жалпы түсінік ...................................................3

2. Бір өлшемді массив құру..................................................................7

3. Екі өлшемді және үш өлшемді массивтер......................................9

4. Символдық массивтер......................................................................11

5. Массивтерді тағайындау мен салыстыру.......................................12

6. Массивті сұрыптау............................................................................15

ҚОРЫТЫНДЫ.......................................................................................18

Қолданылған әдебиеттер тізімі.............................................................21

КІРІСПЕ

Бұрын талқыланған мәліметтер типтері қарапайым деп саналады. Себебі, олар тек бір объектіге, яғни бір санға немесе символға қолданылады. Python сонымен қатар бір типтегі бірнеше элементтерден тұратын нысандарды қолдана алады. Массив дегеніміз - бір типке жататын элементтерден тұратын құрылымдық тип.

Қарапайым типтер диапазонына жататын стандартты (бүтін, нақты) және қолданушы (тізбекті тип) типтерде бір айнымалыны сақтау үшін тек қана компьютер жадының бір ұяшығы қолданылады. Алайда, бағдарламалаудың көптеген мәселелерін шешкен кезде деректерді бөлек айнымалы түрінде сақтағаннан гөрі әр элементтің мәліметтерін бір жерде сақтау тиімдірек болады.

Массив дегеніміз - бір типті мәліметтер жиынтығы, оның барлық элементтері ортақ атауы бар. Массивтер құрылымдық типтер. Массив элементтері нөмірленген. Сіз массивтің әр элементінің индексін көрсете отырып жұмыс жасай аласыз. Векторларды массивтің мысалы ретінде қарастыруға болады.

Массив сипаттамалары:

· Type - массив элементтерінің жалпы түрі;

Көлем - массив көрсеткіштерінің саны;

Шектеу - әр индекстің шектеулерге сәйкестігі;

Формат - көлемдер мен шектеулер жиынтығы.

Массив элементтерімен жұмыс жасау кезінде жақшадағы индекс массив атауынан кейін көрсетілуі керек. Индекс ретінде сандар қолданылады.

Массивтің кез-келген элементтерімен жұмыс жасау кезінде бағдарламалау кезіндегі олардың индексінің мәні «Түрлер» немесе «айнымалылар» бөлімінде сипатталған шектеуден аспауы керек. Егер массив индексінің мәні көрсетілген шектен асып кетсе, синтаксистік қате тіркеліп, «Индекс типі декларациямен сәйкес келмейді» деген сөйлем экранда пайда болады.

Массив - (француздық массивтен аударылған - күшті, тұтас) біртектес заттардың немесе тұтас объектілердің жиынтығын білдіреді. Мысалы, күнделікті өмірде ол тұрғын үй кешені (тұрғын алқаптары), ормандар (орман алқаптары) сияқты ұғымдармен бірге қолданылады (1-сурет).

Массив дегеніміз біртектес емес мәліметтердің тізбегі, олардың элементтері әртүрлі атаулармен ерекшеленетін жалпы атпен біріктіріледі.

Математикада да, программалауда да сандар тізбегі массив (жиын) ретінде қарастырылады. Мысалы, 2, 4, 6, ... жұп сандар тізбегі. Математика мен бағдарламалауда массив a1, a2, ..., an - 1, an формулаларымен өрнектеледі.

Python **Javascript**

|  |  |
| --- | --- |
| my\_list = [3, 2, 10, "Hello"]  my\_list  [3, 2, 10, 'Hello'] | **var** my\_array = [3, 2, 10, "Hello"];  my\_array;  [ 3, 2, 10, 'Hello' ] |

Конструктордың көмегімен тізім / массив құру #

my\_list = list('Hello!')

my\_list

['H', 'e', 'l', 'l', 'o', '!']

Конструктордың көмегімен тізім / массив құру #

**var** my\_array = Array(1, 2, 3, "Hello");

my\_array;

[ 1, 2, 3, 'Hello' ]

**var** my\_array = **new** Array(1, 2, 3, "Hello");

my\_array;

[ 1, 2, 3, 'Hello' ]

**var** my\_array = **new** Array(4);

my\_array;

[ , , , ]

**var** my\_array = **new** Array("4");

my\_array;

[ '4' ]

**var** my\_array = **new** Array("Hello!");

my\_array;

[ 'Hello!' ]

Индекске қол жеткізу

|  |  |
| --- | --- |
| my\_list = [3, 2, 10, "Hello"]  my\_list[0]  3  my\_list[1]  2  my\_list[-1]  'Hello' | **var** my\_array = [3, 2, 10, "Hello"];  my\_array[0];  3  my\_array[1];  2  my\_array[my\_array.length-1];  Hello |

Тізім элементінің / массивтің мәнін өзгерту

|  |  |
| --- | --- |
| my\_list = [3, 2, 10, "Hello"]  my\_list[0] = 100  my\_list  [100, 2, 10, 'Hello'] | **var** my\_array = [3, 2, 10, "Hello"];  my\_array[0] = 100;  my\_array;  [ 100, 2, 10, 'Hello' ] |

Тізім / массив ұзындығы #

|  |  |
| --- | --- |
| my\_list = [3, 2, 10, "Hello"]  len(my\_list)  4 | **var** my\_array = [3, 2, 10, "Hello"];  my\_array.length;  4 |

Элементті тізім / массивке қосу

|  |  |
| --- | --- |
| my\_list = [3, 2, 10, "Hello"]  my\_list.append(5)  my\_list  [3, 2, 10, 'Hello', 5] | **var** my\_array = [3, 2, 10, "Hello"];  my\_array.push(5);  my\_array;  [ 3, 2, 10, 'Hello', 5 ] |

Басқа тізім / массив элементтерімен тізімді / массивті кеңейту

**var** my\_array = [3, 2, 10, "Hello"];

**var** other\_array = [1, 2, 3];

my\_array.push.apply(my\_array, other\_array);

my\_array;

[ 3, 2, 10, 'Hello', 1, 2, 3 ]

**var** my\_array = [3, 2, 10, "Hello"];

**var** other = [1, 2, 3];

Array.prototype.push.apply(my\_array, other);

my\_array;

[ 3, 2, 10, 'Hello', 1, 2, 3 ]

Python

my\_list = [3, 2, 10, "Hello"]

other\_list = [1, 2, 3]

my\_list.extend(other\_list)

my\_list

[3, 2, 10, 'Hello', 1, 2, 3]

Оператор in

my\_list = [1, 7, 10]

1 **in** my\_list

True

7 **in** my\_list

True

10 **in** my\_list

True

2 **in** my\_list

False

"1" **in** my\_list

Python-да массивтер мен матрицаларды қосу және өзгерту NumPy кітапханасының көмегімен жасалады. Осылайша сіз бір өлшемді, екі өлшемді және көп өлшемді массив жасай аласыз. Кітапханада әр түрлі математикалық есептерді шешуге қажет кең пакеттер бар. Ол екіөлшемді және көпөлшемді массивтерді құруды қолдап қана қоймайды, сонымен қатар біртекті көпөлшемді матрицалардың жұмысын қамтамасыз етеді.

Осы пакеттің функцияларына қол жеткізу және оны пайдалану үшін ол импортталады:

**import** numpy **as** np

Массив () функциясы Python-да бір және екі өлшемді массивті динамикалық түрде орнатудың ең қарапайым әдістерінің бірі болып табылады. Ол ndarray типінің нысанын жасайды:

массив = np.array (/ \* көптеген элементтер \* /)

Тексеру үшін array.type () функциясы қолданылады - ол дәлел ретінде жасалған массивтің атын алады.

Егер сіз массив түрін қайта анықтағыңыз келсе, құру кезеңінде dtype = np.complex қолданыңыз:

массив2 = np. массив ([/ \* элементтер \* /, dtype = np.complex)

Тапсырма Python-да белгілі бір ұзындықтың бір өлшемді немесе екі өлшемді массивін көрсету керек болса және оның мәндері осы кезеңде белгісіз болса, ол нөлдер () функциясы арқылы нөлдермен толтырылады. Одан басқа, бірліктер матрицасын солардың () функциясы арқылы алуға болады. Сонымен қатар элементтер саны мен ішіндегі массивтердің саны дәлел ретінде қабылданады:

np.zeros (2, 2, 2)

Мысалы, Python ұзындығы екі элементі бар екі массивті ішке қалай орнатады:

массив ([

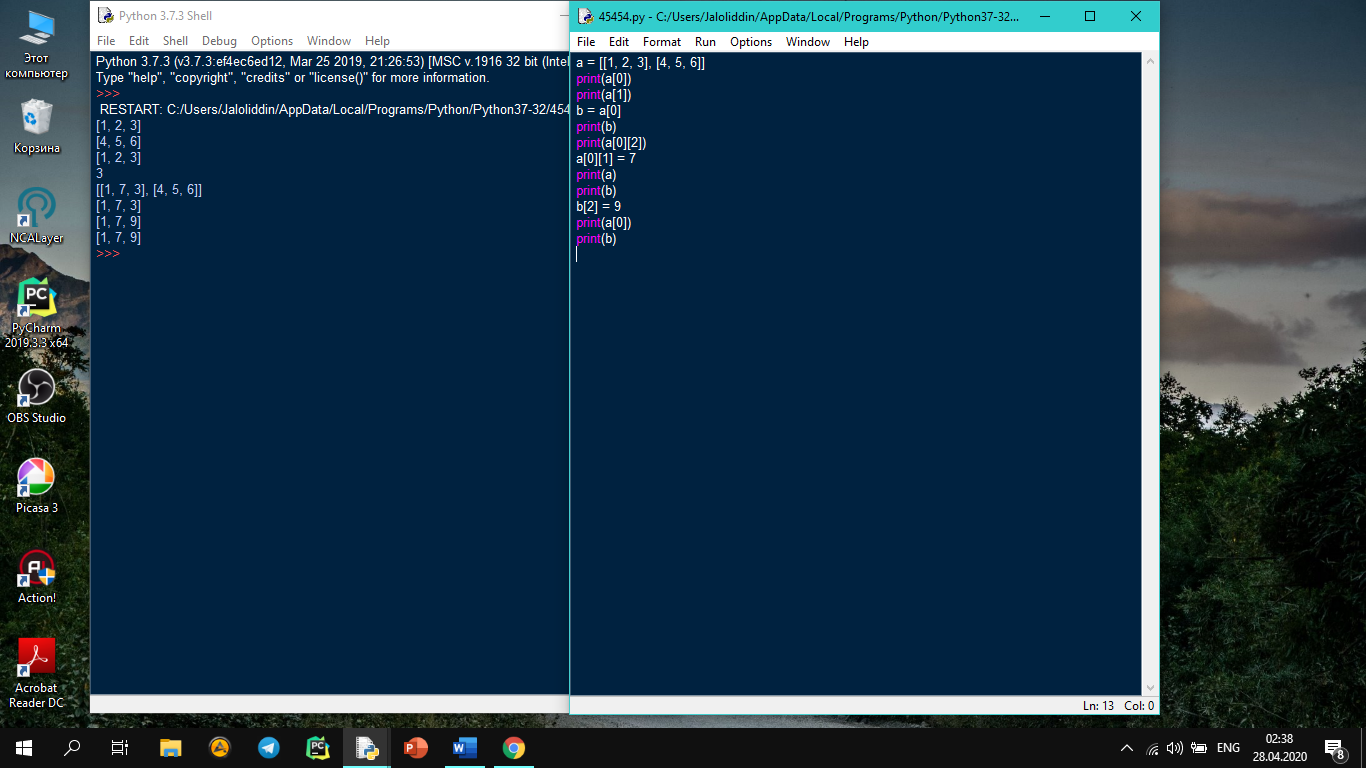
[[0, 0]]

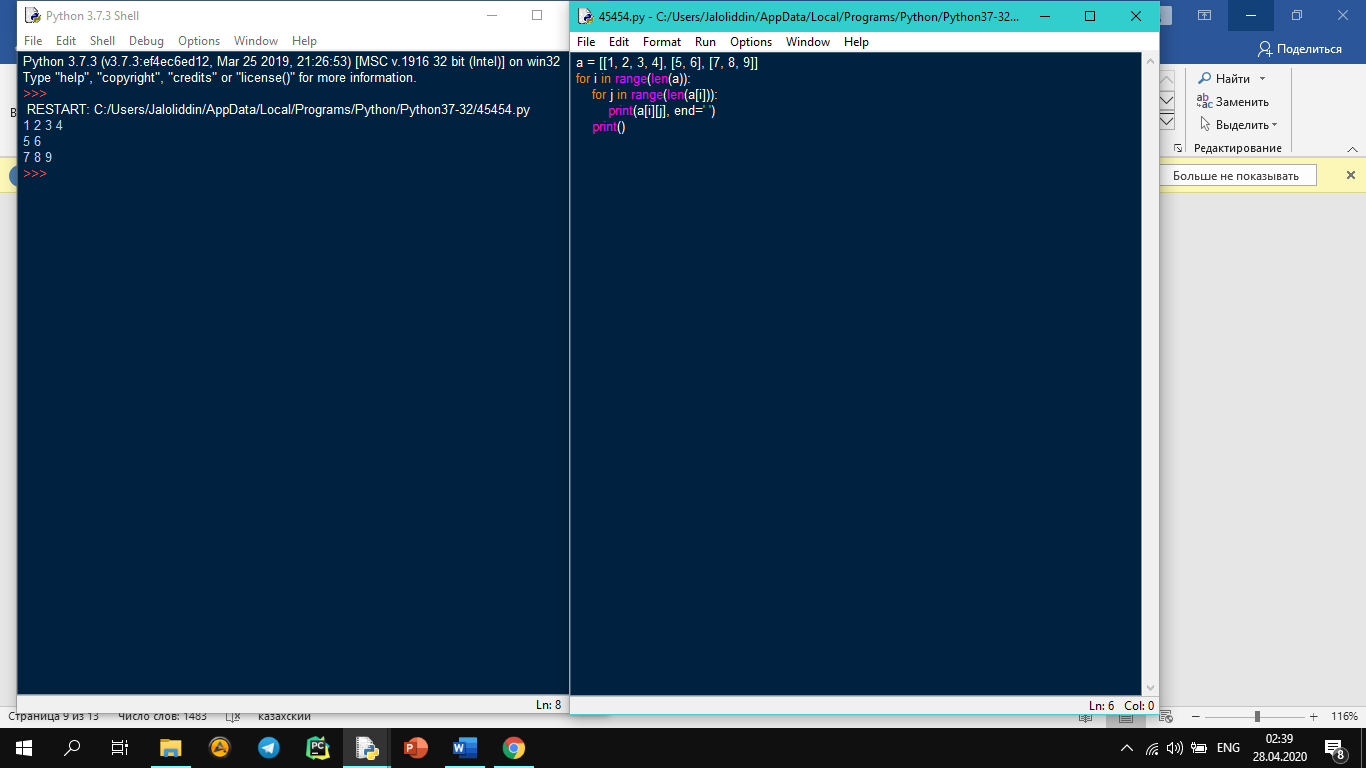
[[0, 0]]]

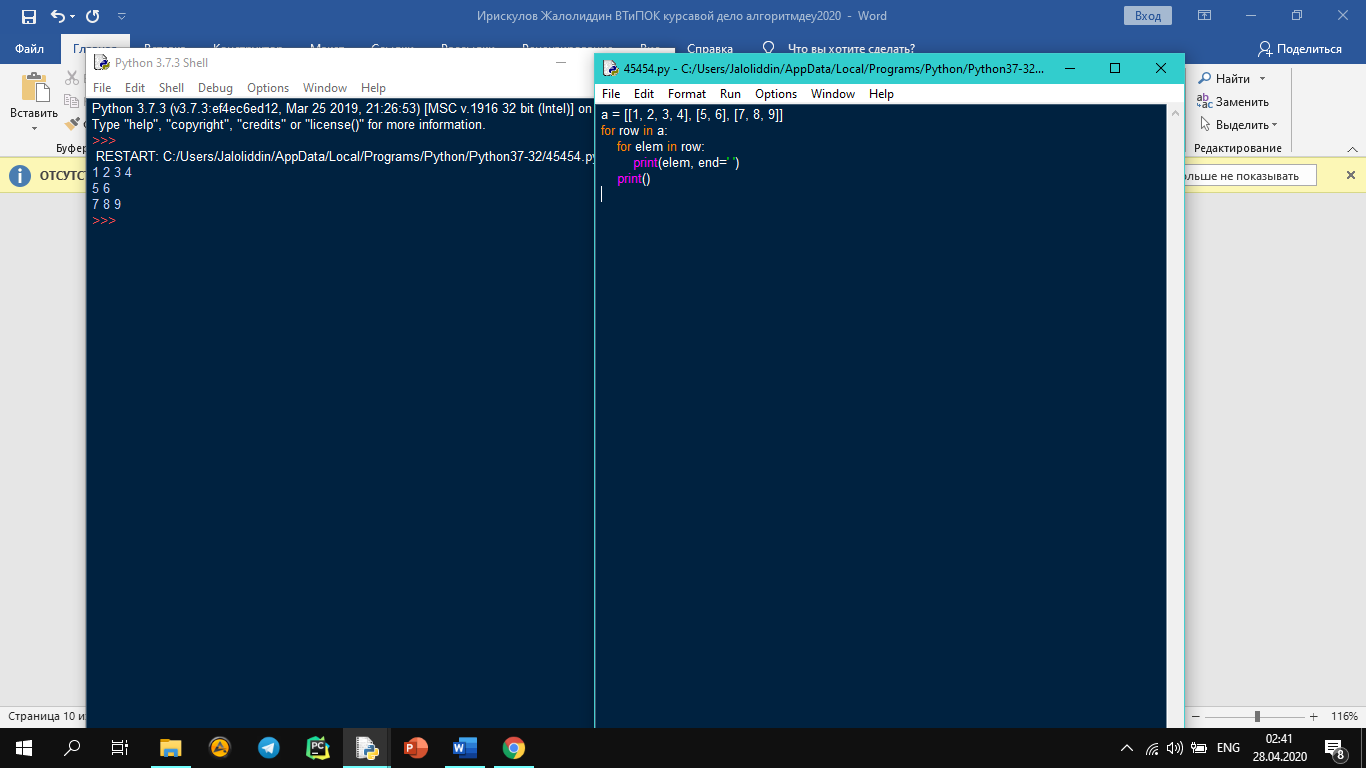
)

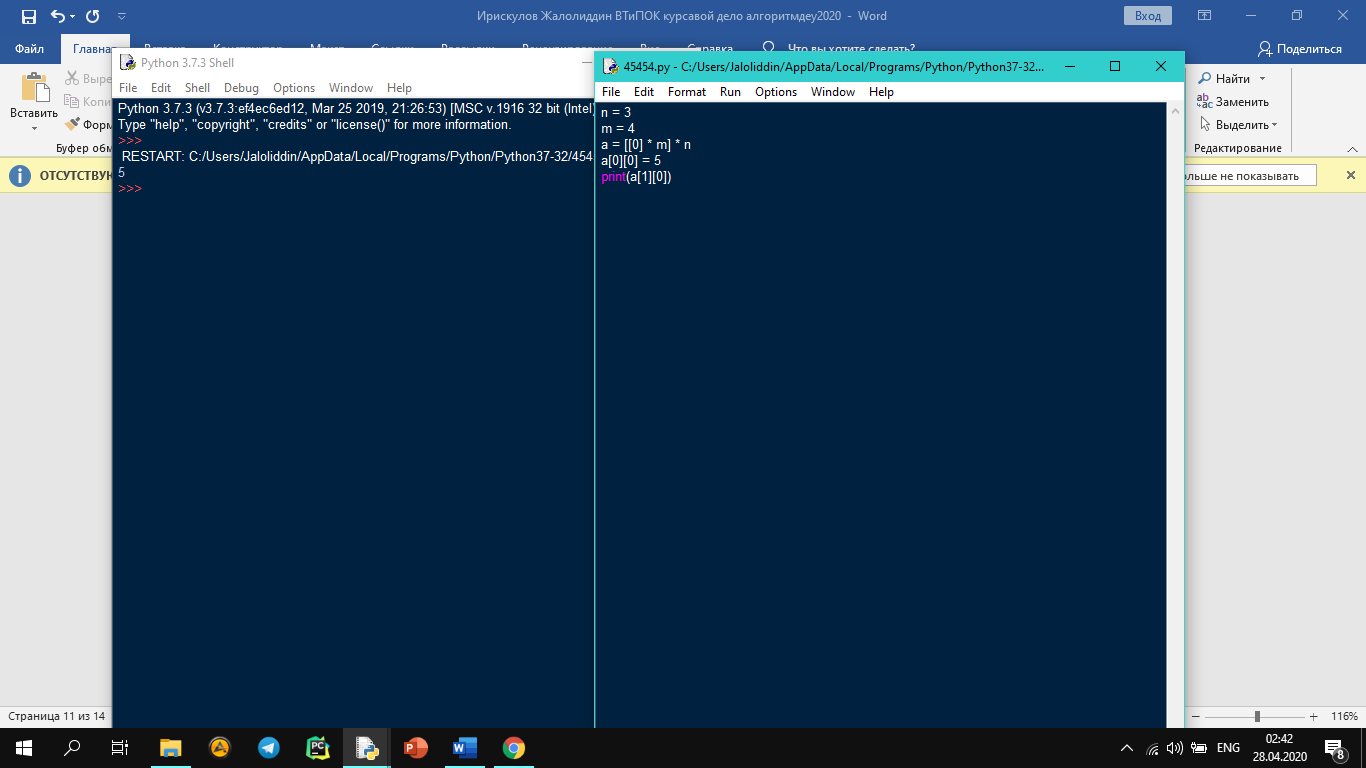
1. Кірістірілген тізімдерді өңдеу және тізімдеу

Көбіне тапсырмаларда тіктөртбұрышты кестелерді мәліметтермен сақтау керек. Мұндай кестелер матрицалар немесе екі өлшемді массивтер деп аталады. Python бағдарламалау тілінде кесте жолдардың тізімі түрінде ұсынылуы мүмкін, оның әр элементі өз кезегінде, мысалы, сандар тізімінен тұрады. Мысалы, біз екі жолдан және үш бағандардан тұратын, әр түрлі әрекеттер орындалатын сандық кесте құратын бағдарламаны береміз.









Питондағы массив | 1-жинақ

Кейбір жалпы контейнерлер [тізімінен басқа](http://quiz.geeksforgeeks.org/python-set-3-strings-lists-tuples-iterations/) , Python өзінің анықтамасы бойынша белгілі бір типтегі контейнерлермен жұмыс жасай алады. Массивті « **массив** » деп аталатын модуль арқылы питон арқылы өңдеуге болады . Олар бізге деректер типінің белгілі бір мәндерін ғана қолданған кезде пайдалы болуы мүмкін

**Array амалдары:**

**1. массив (мәліметтер түрі, құндылықтар тізімі)** : - Бұл функция мәліметтер типімен және оның дәлелдерінде көрсетілген мәндер тізімімен массив **құру** үшін қолданылады . Деректердің кейбір түрлері төмендегі кестеде көрсетілген.

**2. append ()** : - Бұл функция массивтің **соңында** өзінің дәлелдерінде көрсетілген **мәнді қосу үшін** қолданылады .

**3. Insert (i, x)** : - Бұл функция оның дәлелінде көрсетілген **позицияға мән қосу үшін** қолданылады .

# Python code to demonstrate the working of

# array(), append(), insert()

# importing "array" for array operations

import array

# initializing array with array values

# initializes array with signed integers

arr = array.array('i', [1, 2, 3])

# printing original array

print ("The new created array is : ",end=" ")

for i in range (0, 3):

    print (arr[i], end=" ")

print("\r")

# using append() to insert new value at end

arr.append(4);

# printing appended array

print("The appended array is : ", end="")

for i in range (0, 4):

    print (arr[i], end=" ")

# using insert() to insert value at specific position

# inserts 5 at 2nd position

arr.insert(2, 5)

print("\r")

# printing array after insertion

print ("The array after insertion is : ", end="")

for i in range (0, 5):

    print (arr[i], end=" ")

Қорытынды:

Жаңа жасалған массив: 1 2 3

Қосылған массив: 1 2 3 4

Кірістіруден кейінгі массив: 1 2 5 3 4

**4. поп ()** : - Бұл функция **нүктесінде элементін жояды** оны өз дәлел айтылған, және қайтару.

**5. Remove ()** : - Бұл функция өзінің дәлелдерінде аталған мәннің **бірінші пайда болуын жою үшін** қолданылады .

|  |
| --- |
| # Python code to demonstrate the working of  # pop() and remove()    # importing "array" for array operations  import array    # initializing array with array values  # initializes array with signed integers  arr= array.array('i',[1, 2, 3, 1, 5])    # printing original array  print ("The new created array is : ",end="")  for i in range (0,5):      print (arr[i],end=" ")    print ("\r")    # using pop() to remove element at 2nd position  print ("The popped element is : ",end="")  print (arr.pop(2));    # printing array after popping  print ("The array after popping is : ",end="")  for i in range (0,4):      print (arr[i],end=" ")    print("\r")    # using remove() to remove 1st occurrence of 1  arr.remove(1)    # printing array after removing  print ("The array after removing is : ",end="")  for i in range (0,3):      print (arr[i],end=" ") |

Қорытынды:

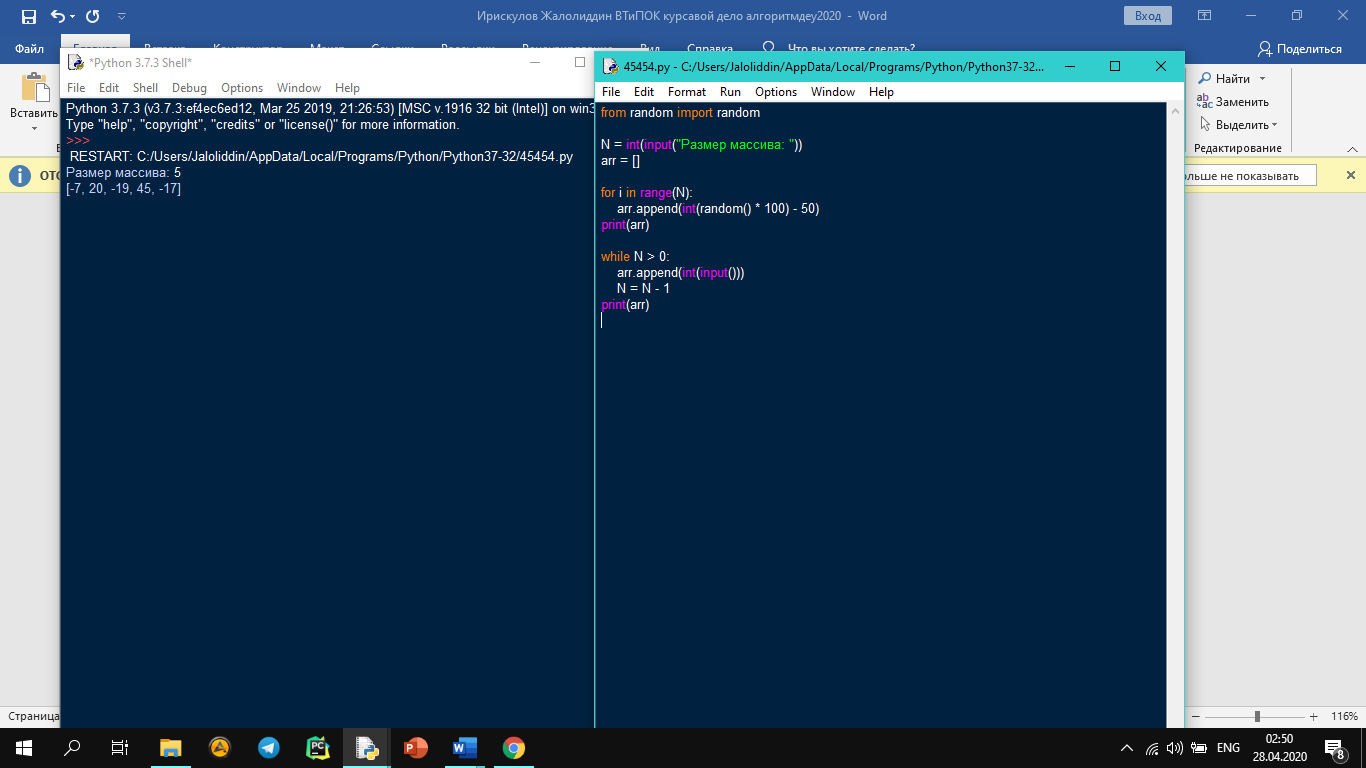
Жаңа жасалған массив: 1 2 3 1 5

Пайда болған элемент: 3

Пайда болғаннан кейінгі массив: 1 2 1 5

Алынғаннан кейінгі массив: 2 1 5

Python-дағы бір өлшемді массивтер - бұл элементтер тізімі. Мәндер тік жақша ішінде көрсетіледі, онда олар үтірмен жазылады. Әдетте, кез-келген элемент индексті шақырып, оған жаңа мән бере алады.



Программа құрылымы:

Кез-келген программа комментарийден басталады. Комментарийді Си тілінде блок түрінде беруге болады.

Содан кейін препроцессор дерективасы орналасады. Препроцессор директивасы берілгенде кітапханалық файлда - тақырыптық файлда орналасқан кодты программаға қою .h файлдар деп аталады. Тақырыптық файлдарда әртүрлі тұрақты мен кейін main функциясы сипатталған бөлігі орналасады.

Main функциясы... одан орындалуы басталып, Return инструкциясы шақырылғанда аяқталады.

Main функциясының денесі фигуралық жақшада орналасады.

Си тілінде жазылған программа текстінде кездесетін инструкциялар (нұсқаулар) препроцессор директивасы (ПД) деп аталады. Олар орындалады және программаның текстін өзгертуге мүмкіндік береді. Мысалы текстегі кейбір лексемдерді өзгерту, басқа файлдан мәтінді да бір бөлігін орындауға, трансляциялауға шектеу қою т.б. Барлық препроцессор директивасы таңбасымен басталады және үтір (;) #include директивасы.

Бұл программаға көрсетілген файлдағы мәліметтерді қосады. Бұл директиваның екі формасы бар:

#include «файл аты»

# include

#include директивасы іштестірілген болуы мүмкін, яғни қосылатын файлда тағы кездесуі мүмкін. бұл қосу үшін қолданады, яғни стандартты кітапханалық функциялар прототиптерінен тұратын файлдарды қосуға мүмкіндік береді. Сондықтан С тіліндегі программа осы директивадан #define - жиі қызметші сөздерді, операторларды және өрнектерді кейбір идентификаторлармен ауыстыруға мүмкіндік береді

Программа орындалу барысында мәндер өзгеретін шамалар айнымалылар деп аталады. Айнымалылар да, тұрақтылар сияқты программаның var деген қызметші сөзінен басталатын арнайы бөлімінде сипатталынуы тиіс

Әр адам программаны әр түрлі жазады(мен өзім программаны қысқа жазғанды жақсы көрем). Ең бастысы программа логикалық түрде дұрыс болу керек.

Типэкодтар - бұл массив мәндерінің түрін немесе массив түрін анықтау үшін қолданылатын кодтар. Қол жетімді код кодтарының тізімі:

* 'b' -> 1 байт көлеміндегі қол қойылған бүтін санды ұсынады
* 'B' -> 1 байт өлшемдегі қол қойылмаған бүтін сандарды ұсынады
* 'c' -> 1 байт көлеміндегі кейіпкерді ұсынады
* 'u' -> 2 байттан тұратын кодтың символын ұсынады
* 'h' -> 2 байт көлеміндегі қол қойылған бүтін санды ұсынады
* 'H' -> 2 байт көлеміндегі қол қойылмаған бүтін санды ұсынады
* 'i' -> 2 байт көлеміндегі қол қойылған бүтін санды ұсынады
* 'I' -> 2 байт көлеміндегі қол қойылмаған бүтін санды ұсынады
* 'w' -> 4 байттан тұратын кодтың символын ұсынады
* 'l' -> 4 байтқа қол қойылған бүтін санды ұсынады
* 'L' -> 4 байт көлеміндегі қол қойылмаған бүтін санды ұсынады
* 'f' -> 4 байт көлеміндегі өзгермелі нүктені ұсынады
* 'd' -> 8 байт көлеміндегі өзгермелі нүктені ұсынады

Қатысты тақырыпта сіз [Python тізімдерін](https://www.thegeekstuff.com/2013/06/python-list/) қалай тиімді пайдалану керектігін де білуіңіз керек .

1. Негізгі мысал

Мұнда 5 бүтін сандардан тұратын массивтің қарапайым мысалы келтірілген

~ $ питон

Python 2.7.4 (әдепкі, 19 сәуір 2013, 18:28:01)

[GCC 4.7.3] linux2 бойынша

Қосымша ақпарат алу үшін «анықтама», «авторлық құқық», «несие» немесе «лицензия» деп теріңіз.

**Массив импорттауынан**

**>>>**

**\* үшін my\_array = массив ('i', [1,2,3,4,5]) >>> i үшін my\_ массивте:**

**... басып шығару (i)**

**...**

**1**

**2**

**3**

**4**

**5**

Осылайша біз қарапайым питон массивін құрып, оны басып шығарамыз.

2. Жеке элементтерге индекстер арқылы қол жеткізіңіз

Жеке элементтерге индекстер арқылы қол жеткізуге болады. Мұнда мысал келтірілген:

>>> my\_rray [1]

2

>>> my\_array [2]

3

>>> my\_array [0]

1

Массивке мәнді кірістіру () әдісі арқылы енгізіңіз

Массивтің кез-келген индексіне мән енгізу үшін біз енгізу () әдісін қолдана аламыз. Мұнда мысал келтірілген:

>>> **my\_array.insert (0,0)**

>>> менің\_жарым

массив ('i', [ **0** , 1, 2, 3, 4, 5, 6])

Жоғарыда келтірілген мысалда, enter () әдісін қолдана отырып, 0 мәні 0 индексіне енгізілді. Бірінші дәлел - индекс, ал екінші аргумент - бұл мән.

5. Питон массивін ext () әдісін қолданып кеңейтіңіз

Питон массивін ext () әдісін қолдана отырып, бірнеше мәннен ұзартуға болады. Мұнда мысал келтірілген:

>>> my\_extnd\_array = массив ('i', [7,8,9,10])

>>> **my\_array.extend (my\_extnd\_array)**

>>> менің\_жарым

массив ('i', [0, 1, 2, 3, 4, 5, 6, **7, 8, 9, 10** ])

Сонымен, my\_array массиві my\_extnd\_array мәндерімен кеңейтілгенін көреміз.

6. List () әдісін пайдаланып тізімнен элементтерді массивке қосыңыз

Мұнда мысал келтірілген:

>>> с = [11,12,13]

>>> **my\_array.fromlist (c)**

>>> менің\_жарым

массив ('i', [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, **11, 12, 13** ])

Сонымен, 11,12 және 13 мәндері 'c' тізімінен 'my\_array' тізіміне қосылғанын көреміз.

7. Алып тастау () әдісі арқылы кез-келген массив элементін алып тастаңыз

Мұнда мысал келтірілген:

>>> **my\_ray.remove (13)**

>>> менің\_жарым

массив ('i', [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12])

Сонымен, 13 элемент массивтен алынып тасталғанын көреміз.

8. Pop () әдісін қолдана отырып, соңғы массив элементін алып тастаңыз

Мұнда мысал келтірілген:

>>> **my\_array.pop ()**

12

>>> менің\_жарым

массив ('i', [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11])

Сонымен, соңғы элемент 12 массивтен шыққанын көреміз.

9. Кез келген элементті индекс () әдісі арқылы оның индексі арқылы алыңыз

Мұнда мысал келтірілген:

>>> **my\_array.index (5)**

5

Осылайша, 5 индексіндегі мән осы әдіс арқылы алынғанын көреміз.

10. Питон массивін кері () әдісін қолданып төңкеріңіз

Мұнда мысал келтірілген:

>>> **my\_array.reverse ()**

>>> менің\_жарым

массив ('i', [11, 10, 9, 8, 7, 6, 5, 4, 3, 2, 1, 0])

Осылайша, біз толық массивтің өзгергенін көреміз.

11. Массив буфері туралы ақпаратты buffer\_info () әдісі арқылы алыңыз

Бұл әдіс сізге жадтағы массив буферінің басталу мекенжайын және массивтегі элементтер санын қамтамасыз етеді. Мұнда мысал келтірілген:

>>> **my\_array.buffer\_info ()**

(33881712, 12)

Осылайша, буфердің бастапқы мекен-жайы және шығыс элементтерінің саны берілгенін көреміз.

12. Санау () әдісі арқылы элементтің пайда болу санын тексеріңіз

Мұнда мысал келтірілген:

>>> **my\_ray.count (11)**

1

Сонымен, 11 элемент массивте бір рет пайда болғанын көреміз.

13. tostring () әдісін қолдана отырып, массивті жолға түрлендіріңіз

Мұнда мысал келтірілген:

>>> my\_char\_array = массив ('c', ['g', 'e', ​​'e', ​​'k'])

>>> my\_char\_array

массив ('c', 'geek')

>>> **my\_char\_array.tostring ()**

'геек'

Сонымен, таңбалар массиві осы әдісті қолдана отырып, жолға түрлендірілгенін көреміз.

14. Толисті () әдісті қолдана отырып, массивті сол элементтері бар питон тізіміне түрлендіріңіз

Мұнда мысал келтірілген:

>>> **c = my\_rray.tolist ()**

>>> ә

[11, 10, 9, 8, 7, 6, 5, 4, 3, 2, 1, 0]

Сондықтан біз 'c' тізімін my\_array-да tolist () әдісі арқылы құрғанын көреміз.

15. fromstring () әдісін қолдана отырып, char массивіне жол қосыңыз

Мұнда мысал келтірілген:

>>> **my\_char\_array.fromstring («заттар»)**

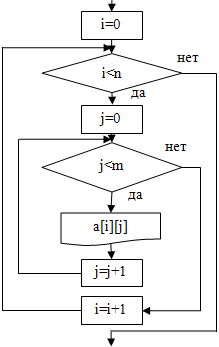
>>> my\_char\_array

массив ('c', 'geek **stuff** ')

Экранда екі өлшемді массивті көрсету

№11 сурет

блок-схемa Бағдарламаның бір бөлігі

 for(i=0; i<n; i=i+1){

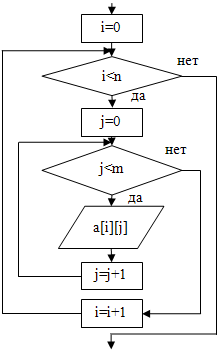
for(j=0; j<m; j=j+1){

cout<<setw(6)<<a[i][j];

}

cout<<endl;

}

double a[n][m]; №12 сурет

int i,j;

fstream file;

file.open("1.txt", ios::in);

for(i=0; i<n; i=i+1){

for(j=0; j<m; j=j+1){

file>>a[i][j];

}

}

file.close();

Python-да тізім / массивтерде қолдануға болатын бекітілген әдістер жиынтығы бар.

|  |  |
| --- | --- |
| **Method** | **Description** |
| [append()](https://www.w3schools.com/python/ref_list_append.asp) | Adds an element at the end of the list |
| [clear()](https://www.w3schools.com/python/ref_list_clear.asp) | Removes all the elements from the list |
| [copy()](https://www.w3schools.com/python/ref_list_copy.asp) | Returns a copy of the list |
| [count()](https://www.w3schools.com/python/ref_list_count.asp) | Returns the number of elements with the specified value |
| [extend()](https://www.w3schools.com/python/ref_list_extend.asp) | Add the elements of a list (or any iterable), to the end of the current list |
| [index()](https://www.w3schools.com/python/ref_list_index.asp) | Returns the index of the first element with the specified value |
| [insert()](https://www.w3schools.com/python/ref_list_insert.asp) | Adds an element at the specified position |
| [pop()](https://www.w3schools.com/python/ref_list_pop.asp) | Removes the element at the specified position |
| [remove()](https://www.w3schools.com/python/ref_list_remove.asp) | Removes the first item with the specified value |
| [reverse()](https://www.w3schools.com/python/ref_list_reverse.asp) | Reverses the order of the list |
| [sort()](https://www.w3schools.com/python/ref_list_sort.asp) | Sorts the list |

Негізгі операциялар

Төменде массив қолдайтын негізгі операциялар берілген.

* **Траверс** - массивтің барлық элементтерін бір-бірден басып **шығарыңыз** .
* **Кірістіру** - берілген индекске элемент қосады.
* **Жою** - берілген индекстегі элементті жояды.
* **Іздеу** - берілген индекс немесе мән бойынша элементті іздейді.
* **Жаңарту** - берілген индекстегі элементті жаңартады.

Массив Python-да массив модулін python бағдарламасына импорттау арқылы жасалады. Содан кейін массив көрсетілгендей жарияланды.

Іздеу операциясы

Массив элементін оның мәніне немесе индексіне сүйене отырып іздеуге болады.

Мұнда біз деректер элементін кірістірілген индекстеу () әдісін қолдана отырып іздейміз.

from array import \*

array1 = array('i', [10,20,30,40,50])

print (array1.index(40))

Жоғарыда келтірілген бағдарламаны құрғанда және орындағанда, элементтің индексін көрсететін келесі нәтиже шығады. Егер массивте мән болмаса, онда эпрограмма қатені қайтарады.

Жою операциясы

Жою дегеніміз - массивтен бар элементті алып тастау және массивтің барлық элементтерін қайта ұйымдастыру.

Мұнда біз деректер элементін массивтің ортасында кірістірілген алып тастау () әдісін қолданып алып тастаймыз.

from array import \*

array1 = array('i', [10,20,30,40,50])

array1.remove(40)

for x in array1:

print(x)

Жоғарыда келтірілген бағдарламаны құрастырғанда және орындағанда, ол массивтен элементтің жойылғанын көрсететін келесі нәтиже шығарады.

Жаңарту әрекеті

Жаңарту әрекеті берілген индекстегі массивтен бар элементті жаңартуды білдіреді.

Мұнда біз жаңартқымыз келетін индекске жаңа мән тағайындадық.

from array import \*

array1 = array('i', [10,20,30,40,50])

array1[2] = 80

for x in array1:

print(x)

Жоғарыда келтірілген бағдарламаны құрастырғанда және орындағанда, ол индекстің 2 орнында жаңа мәнді көрсететін келесі нәтижені шығарады.

Шығу

10

20

80

40

50

ҚОРЫТЫНДЫ.

Біз жалпы Python тілінде массивтерге жоғары денгейде мән бердік.

Массивтер туралы жалпы түсінік алдық . Массивтердің түрлері , массивтерге байланысты есептер шығардық . Массивтерге байланысты есепті кез келген

проектке қарап есеп құрастыруға болатындығын үйрендік .

|  |  |
| --- | --- |
| Сонымен қатар массивтер бірөлшемдімассив; екіөлшемдімассив; көпөлшемд  массив болып 3 түргше бөлінетінін білдік.  Ал массивтің типтері сандық , мәтіндік , символдықтан туратынын білдік  Массивтің ерекшеліктері   элементтердің барлығы біртипті , элементтер бір атаумен аталады; элементтер  жадта қатар орналасатындығын білдік.  Массивтерде екіөлшемді  массивтер көп кездеседі. Олармен арифметикалық амалдарды  орындауға болады. Екі матрицаны қосуға, көбейтуге, ең үлкен, ең кіші элементтерін,  олардың орнын табуна болады.  Сонымен массивтерге байланысты курстық жұмысты аяқтаймын. |  |

Әдебиеттер тізімі

1.Омарова Н., Тұрмағамбетова К.У., Нүриденова К.Н. Паскаль тiлiнде программалау негiздерi. – Алматы: Бiлiм, 1996.- 58 б.

2.Нақысбеков Б.Қ., Балапанов Е.Қ., Халықова К.З., Дәулетқұлов А.Б.. Паскаль тiлiнiң негiздерi.- Алматы: Рауан, 1998. – 49 б.

3.Әбдиев Қ.С. Бейсик программалау тiлi.- Алматы, 2002**.** –152 б.

4.Рейнгольд Э., Нивергельт Ю., Део Н. Комбинаторные алгоритмы. М., Мир, 2000.- 256 с.

5.Шень А. Программирование: Теоремы и задачи. М., МЦНМО, 1995.- 203 с.

6.Пильщиков В.Н. Сборник упражнений по языку Паскаль. М.: Наука, 1999.- 165

7. Б. Бөрібаев. Алгоритмдеу, мәліметтер құрылымы және программалау тілдері, Алматы, Қазақ университеті, 2012

8. И.А Хахаев Практикум по алгоритмизации и программированию на Python. М. Альт Линукс, 2011

9. Г. Россум, Ф.Л. Дж. Дрейк. Д.С. Язык программирования Python. М. АльтЛинукс, 2001

10.4. Д.М. Златопольский. Основы программирования на языке Python. М. ДМК, 2017

11. Н.Безменов. Тurbo Pascal 7.0. Руководство программиста. М.2000

12.  Буч Г.,Рамбо Д.,Джекобсон А. Язык   UML. Руководство пользователя: Пер. с англ.-М.: ДМК,2000