ЖИТОМИРСЬКА ПОЛІТЕХНІКА
ДЕРЖАВНИЙ УНІВЕРСИТЕТ

@ZTUEDUUA

АЛГОРИТМІЧНЕ ЗАБЕЗПЕЧЕННЯ КОМП'ЮТЕРИЗОВАНИХ ІНФОРМАЦІЙНО-ВИМІРЮВАЛЬНИХ СИСТЕМ

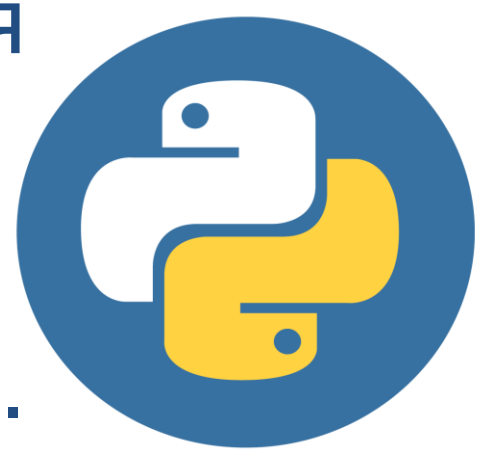
Лекція 12
Тема: Масиви в Python

1. Вступ до масивів в Python
2. Як працює масив у Python?
3. Створення масиву в Python?
4. Методи масиву в Python

1. Введення в списки.

- Масиви можна розглядати як списки в Python. Масив - це тип структури даних, який зберігає дані у суміжній пам'яті. Дані, що зберігаються, мають той самий тип даних. Оскільки змінні можуть зберігати одне значення одночасно, масив може зберігати декілька значень одночасно. Для обробки даних масиву ми повинні зрозуміти, що таке індекс, що таке елемент, як обчислити довжину масиву, як отримати елемент у масиві, як додати або видалити елемент у масиві, як циклічно масив тощо.

- **Індекс:** це число, яке представляє значення масиву, і завжди починається з 0.
- **елемент:** це значення в масиві.
- **len ():** загальна кількість елементів у масиві.
- **append ()** : це спосіб додати елемент до масиву.
- **remove ():** метод видалення елемента з масиву.



2.Як працює масив у Python?

Масив зберігається в суміжних місцях пам'яті, де індекс - це послідовність чисел, яка представляє значення, що зберігаються в кожному конкретному індексі. Щоб отримати доступ до або вказати значення для певного індексу в масиві, ми використовуємо набір квадратних дужок (), також ми можемо використовувати цикл для введення для ітерації через масив. Масив має індекси (множинна форма індексу) та значення.

У кожному індексі зберігається значення. Чому ми використовуємо масиви, це тому, що важко зберігати і запам'ятовувати сотні чисел одночасно, було б простіше і простіше використовувати масиви в такому випадку, скажімо, цілий масив виглядає наступним чином. масив ('i', (1, 2, 3, 4, 5, 6, 7, 8, 9, 10)), то для доступу до цих значень ми будемо використовувати наступний формат.

a (0,1,2…10) => 1,2,3…10

Пам'ятайте, що індекс починається з 0. Ми будемо використовувати цикл for-in також для переходу через даний масив:

```
for i in a
print a(i)
```

Що дасть нам значення від 1 до 10.

3.Як створити масиви в Python?

- Для створення масиву в Python нам потрібно спочатку імпортувати модуль масиву.

```
import array as arr
```

де,
arr => - псевдонім

Інший спосіб імпорту модуля полягає в наступному:

```
from array import *
```

Синтаксис для створення масиву:

```
array(typecode (, intializer))
```

де, typecode => int, float або double або тип значення, яке має масив.

inicijalizer => - необов'язкове значення і може мати будь-який тип, наприклад список, рядок або будь-які ітерабельні елементи певного типу.

Код типу та його значення згадується нижче у форматі таблиць.

TypeCode	C Тип	Тип Python	Значення
i	підписано int	int	2
Я	Без підпису int	довго	2
б	підписав чар	int	1
Б	Непідписаний char	int	1
год	короткий підпис	int	2
Н	Непідписаний короткий	int	2
л	підписаний довго	int	4
L	Без підпису довго	int	4
f	плавати	плавати	8
г	подвійний	плавати	4

Тип коду : i

Код:

```
import array as arr
a=arr.array('i', (10, 20, 30) )
print("Element at 0th index: ", a(0))
print("Element at 1st index: ", a(1))
print("Element at 2nd index: ", a(2))
```

Вихід:

```
Element at 0th index:  10
Element at 1st  index:  20
Element at 2nd index:  30
```

Тип коду: I

Код:

```
import array as arr
a=arr.array('I', (10, 20, 30) )
print("Element at 0th index: ", a(0))
print("Element at 1st index: ", a(1))
print("Element at 2nd index: ", a(2))
```

Вихід:

```
Element at 0th index:  10
Element at 1st  index:  20
Element at 2nd index:  30
```

4.Методи масиву в Python

На даному масиві ми побачимо наступні методи:

1. Введіть код ()

Синтаксис: `array.typecode()`

Данна функція повертає значення типу коду, що використовується в даному масиві.

```
#importing array module
import array as arr
#creating array
a1 = arr.array('i', (100, 200, 300) )
#printing array with method
print(a1.typecode)
```

Вихід:

```
i
```

2. Вставити ()

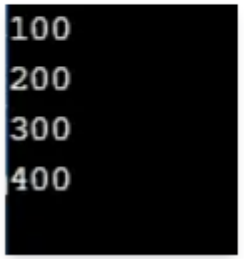
Синтаксис: `array.insert(index, element )`

Він додає елемент до масиву перед індексом

Код:

```
#importing array module
import array as arr
#creating array
a = arr.array('i', (100, 200, 300))
#inserting a value of 400 to after index 2
a.insert(3, 400);
#looping through array a
for i in a:
print(i)
```

Вихід:



3. Оновити ()

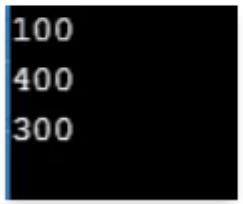
Синтаксис: `arrayname(index) = value`

Він оновлює певне значення в індексі до нового значення.

Код:

```
#importing array module
import array as arr
#creating array
a = arr.array('i', (100, 200, 300) )
#updating a value of 200 to 400
a(1) = 400
#looping through array a
for i in a:
print(i)
```

Вихід:



4. Видалити ()

Синтаксис: `array.remove(element)`

Ця функція видаляє елемент з масиву.

Код:

```
#importing array module
import array as arr
#creating array
a = arr.array('i', (100, 200, 300) )
#deleting a value 100
a.remove(100)
#looping through array a
for i in a:
print(i)
```

Вихід:

```
200
300
```

5. Додавати ()

Синтаксис: `array.append(element)`

Ця функція додає елемент до кінця масиву.

Код:

```
#importing array module
import array as arr
#creating array
a = arr.array('i', (100, 200, 300) )
#appending 400 to the end
a.append(400)
#printing array
for i in a:
print(i)
```

Вихід:

```
100
200
300
400
```

6. Зворотний ()

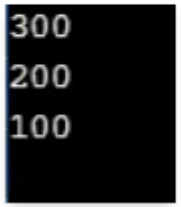
Синтаксис: `array.reverse()`

Ця функція повертає порядок елементів у даному масиві.

Код:

```
#importing array module
import array as arr
#creating array
a = arr.array('i', (100, 200, 300) )
//applying the reverse method to the array
a.reverse()
//printing the array
for i in a:
print(i)
```

Вихід:



7. count ()

Синтаксис: `array.count(element)`

Ця функція повертає, скільки разів елемент траплявся в даному масиві.

Код:

```
#importing array module
import array as arr
#creating array
a3 = arr.array('i', (100, 200, 300, 100, 400, 100, 500) )
#printing the array count
print(a3.count(100))
```

Вихід:

```
3
```

8. Індекс()

Синтаксис: `array.index(x)`

Цей метод повертає "i", що є індексом і найменшим значенням першого появи x у масиві.

Код:

```
#importing array module
import array as arr
#creating array
a3 = arr.array('i', (700, 200, 300, 100, 400, 100, 500) )
#search the value 100 and return its index
print(a3.index(100))
```

Вихід:

9. pop ()

Синтаксис: `array.pop(( i ))`

Ця функція видаляє та повертає елемент, що має індекс і даного масиву. За замовчуванням він видаляє та повертає останній елемент.

Код:

```
#importing array module
import array as arr
#creating array
a3 = arr.array('i', (100, 200, 300) )
# removing 100 and printing
print(a3.pop(0))
```

Вихід:

```
100
```

10. Розмір елементів ()

Синтаксис: `array.itemsize()`

Код:

```
#importing array module
import array as arr
#creating array
a3 = arr.array('i', (100, 200, 300) )
#printing the itemsize
print(a3.itemsize)
```

Вихід :

11. метод len ()

Синтаксис: `len(arrayname)`

Цей метод дає довжину масиву.

Код:

```
#importing array module
import array as arr
#creating array
a3 = arr.array('i', (100, 200, 300, 400, 500) )
#printing the length of method
print(len(a3))
```

Вихід:

@ZTUEDUUA

ЖИТОМИРСЬКА ПОЛІТЕХНІКА

ДЕРЖАВНИЙ УНІВЕРСИТЕТ

- **Розвиваємо лідерів**
- **Створюємо інновації**
- **Змінюємо світ на краще**