

# АЛГОРИТМІЧНЕ ЗАБЕЗПЕЧЕННЯ КОМП'ЮТЕРИЗОВАНИХ ІНФОРМАЦІЙНО-ВИМІРЮВАЛЬНИХ СИСТЕМ



# Лекція 5

## Тема: Цикли в мові Python

1. Цикли.
2. Оператори переривання циклу.
3. Вкладені цикли.



# 1.Цикли

Кожна мова програмування містить конструкцію циклу. Більшість мов є більше однієї такої конструкції. У світі Python є два типи циклів:

- Цикл *for*
- Цикл *while*



Ми використовуємо «цикл for» для повторення елементів колекції, таких як рядок. Оскільки рядок є послідовністю символів, він виглядає як колекція. Таким чином, ми можемо використовувати цикл for для повторення кожного символу в рядку, а потім щось робити з ним.

Діапазон — це ряд значень між двома числовими інтервалами. У Python використовується вбудована функція range() для визначення діапазону значень.

## Цикл - це багатократне виконання однакових дій.

Приклад циклу FOR:

```
for i in range(5):  
    print("Привіт!")
```

Змінна **i** приймає значення 0, 1, 2, 3, 4:

```
for i in range(5):  
    print(i)
```

**for** – «для»

**in range** – «у діапазоні» в перекладі з англійського

Цикл `while` є одним з найпоширеніших у Python. Він дозволяє відтворювати елемент коду знову і знову, до тих пір, поки вказана умова залишається вірною. Щоб цикл працював належним чином, вам потрібен фрагмент коду, який ви хочете використовувати повторно, умова `true` or `false` і ключове слово `while`.

Головна відмінність циклу `while` від `for` полягає в тому, що `while` може повторювати код нескінченну кількість разів, доки виконується умова його застосування.

Цикл `while` також використовується для повторення частин коду, але замість зациклювання на `n` кількість разів він виконує роботу до тих пір, поки не досягне певної умови.

```
i = 0
while i < 3:
    print("Hello")
    i = i+1
```

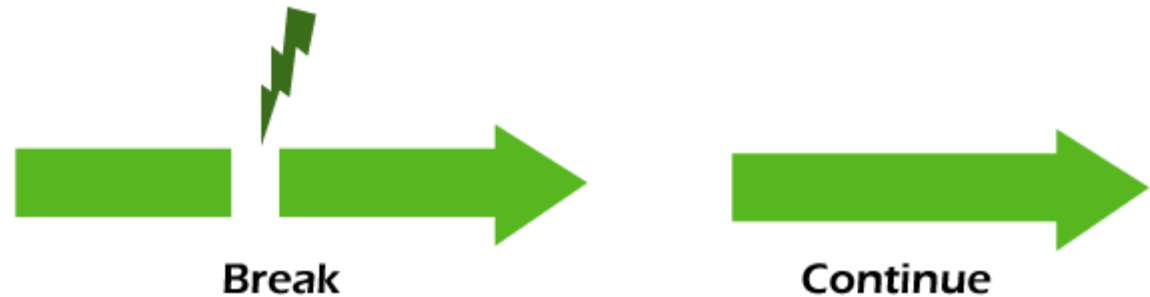
## 2.Оператори переривання циклу.

Оператори переривання циклу поділяються на два види:

- `break`;
- `continue`;

Але іноді роботу програми може вплинути зовнішній чинник. У таких випадках програма може припинити виконання циклу, пропустити цикл або проігнорувати цей зовнішній фактор. Для цього використовують оператори `break`, `continue`.

Оператори `break`, `continue` в Python дозволяють оптимізувати роботу циклів `for` та `while` та спростити код



# Оператор «break»

У Python оператор break дозволяє перервати цикл у разі виникнення зовнішнього чинника. Оператор break потрібно помістити відразу після оператора циклу (зазвичай після if).

Давайте розглянемо такий цикл for з оператором break:

```
1  number = 0
2  for number in range(10):
3      number = number + 1
4      if number == 5:
5          break
6      print('Number is ' + str(number))
7
8  print('Out of loop')
```

У цій невеликій програмі змінна `number` має значення 0. Далі йде оператор `for`, який запускає цикл, якщо змінна `number` менше 10.

Кожен цикл `for` збільшує значення змінної `number` на 1 (`number = number + 1`).

Оператор `if` визначає таку умову: якщо змінна `number` дорівнює 5, то цикл потрібно перервати (тут використовується `break`).

Також у циклі є вираз `print()`, яке виконується при кожній ітерації циклу `for` доти, доки цикл не буде перерваний (оскільки воно йде після оператора `break`).

Після циклу `for` слід ще один вираз `print()`, який повідомляє про переривання циклу.

**Під час запуску програми вийде такий результат:**

```
Number is 1
Number is 2
Number is 3
Number is 4
Out of loop
```

# Оператор «continue»

Оператор `continue` дозволяє пропустити частину циклу у разі виникнення зовнішнього чинника і перейти до наступної ітерації циклу. Тобто поточна ітерація переривається, після чого програма повертається на початок циклу.

Оператор `continue` знаходиться в блоці коду після оператора циклу (зазвичай після `if`).

Повернемося до попереднього прикладу та спробуємо використати оператор `continue` замість `break`.

```
1 number = 0
2 for number in range(10):
3     number = number + 1
4     if number == 5:
5         continue
6     print('Number is ' + str(number))
7
8 print('Out of loop')
```

На відміну від `break`, при використанні оператора `continue` код виконуватиметься навіть після переривання циклу, коли змінна `number = 5`.

```
Number is 1  
Number is 2  
Number is 3  
Number is 4  
Number is 6  
Number is 7  
Number is 8  
Number is 9  
Number is 10  
Out of loop
```

Як бачите, навіть коли значення `number` досягло 5, цикл продовжив роботу.

Оператор `continue` можна використовувати замість вкладеного умовного коду або для оптимізації циклу.

## 3. Вкладені цикли

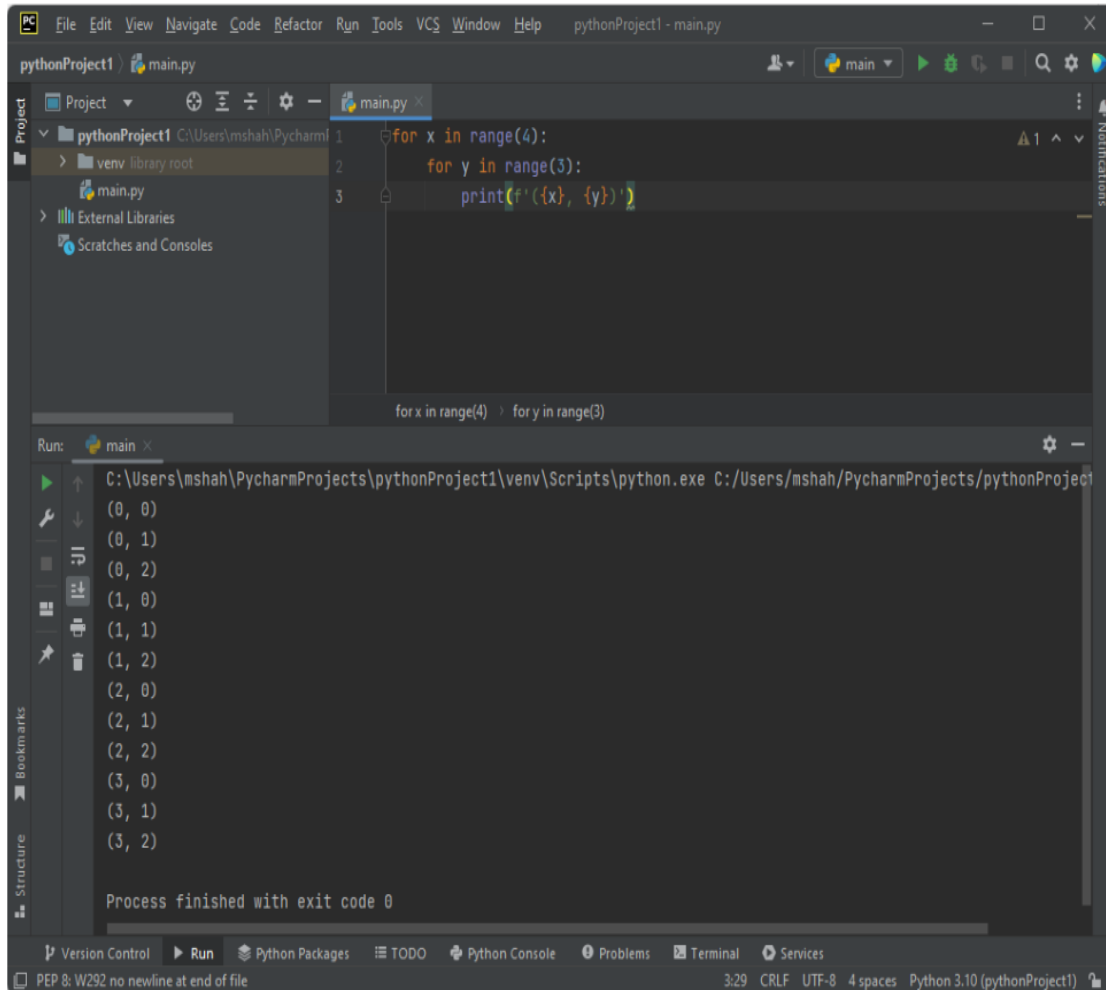
У Python використання вкладеного циклу означає додавання одного циклу в інший цикл, і за допомогою цієї техніки ми можемо робити дивовижні речі. Наприклад, ми можемо легко створити список координат.

Як відомо, **координата** - це комбінація значень «x» і «y». Скажімо, 0 і 0. Тепер припустімо, що ви хочете створити такий список координат. У нас є 0 і 0, потім буде 0 і 1, потім 0 і 2.

Далі ми збираємося змінити «x». Ми будемо використовувати 1 для 'x', і знову ми будемо використовувати ці 3 значення для координат 'y'.

Ми можемо легко згенерувати ці координати за допомогою вкладених циклів.

## Приклад програми:



```
pythonProject1 - main.py
pythonProject1
venv library root
main.py
External Libraries
Scratches and Consoles

1 for x in range(4):
2   for y in range(3):
3     print(f'({x}, {y})')

Run: main x
C:\Users\mshah\PycharmProjects\pythonProject1\venv\Scripts\python.exe C:/Users/mshah/PycharmProjects/pythonProject1
(0, 0)
(0, 1)
(0, 2)
(1, 0)
(1, 1)
(1, 2)
(2, 0)
(2, 1)
(2, 2)
(3, 0)
(3, 1)
(3, 2)

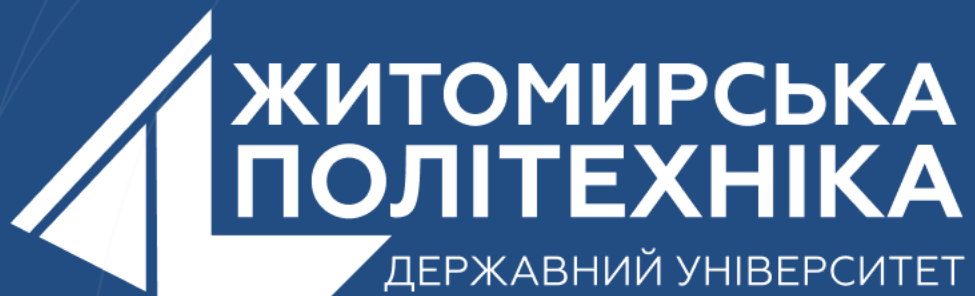
Process finished with exit code 0
```

Тож дозвольте мені пояснити, як саме виконується ця програма. У першій ітерації нашого зовнішнього циклу 'x' дорівнює 0. Тепер ми на рядку 2, тут у нас є новий цикл, який ми називаємо внутрішнім циклом. У цьому внутрішньому циклі на першій ітерації 'y' буде 0, тому 0 і 0 друкуються на терміналі.

Тепер керування повертається до рядка 2 або нашого внутрішнього циклу. У цій другій ітерації 'y' буде встановлено в 1, але ми все ще в першій ітерації нашого зовнішнього циклу. Таким чином, «x» все ще дорівнює 0, але тепер «y» збільшується до 1.

Тому ми бачимо 0 і 1 на терміналі. Це триватиме, доки не буде виконано наш внутрішній цикл, а потім інтерпретатор знову перейде до зовнішнього циклу, і цей процес триватиме до завершення зовнішнього циклу.

   @ZTUEDUUA



- Розвиваємо лідерів
- Створюємо інновації
- Змінюємо світ на краще

