

# Урок 11

## Цикли

## та їх застосування

Поняття циклу. Цикли for та while.

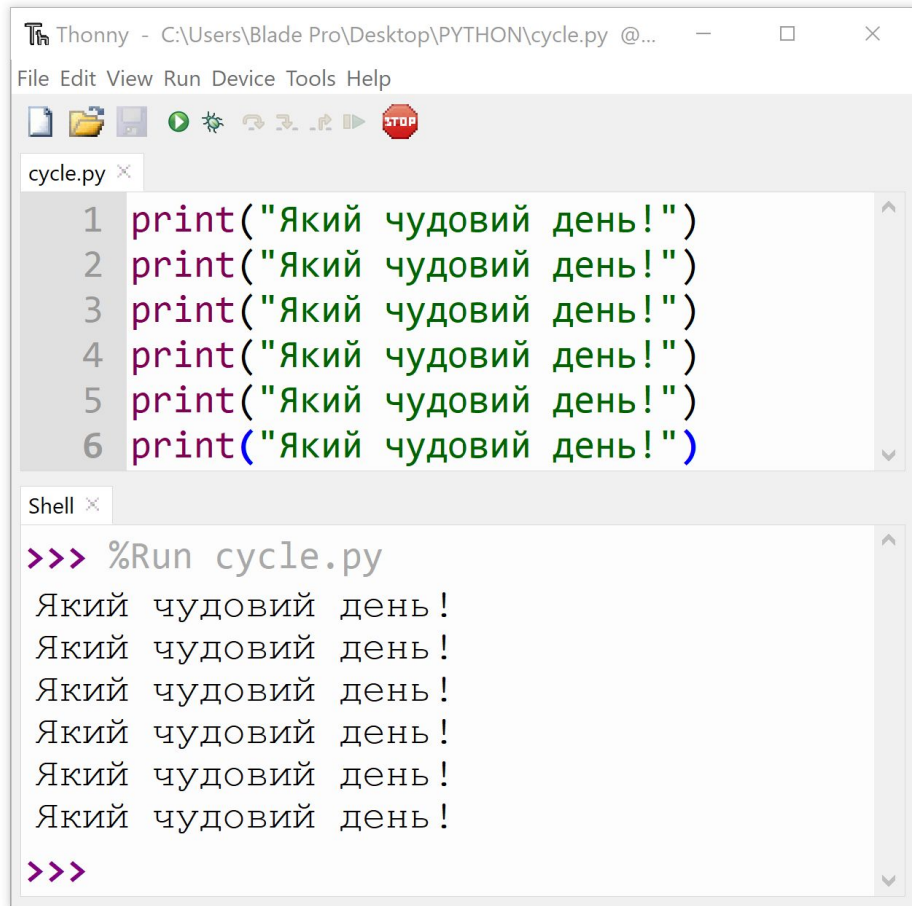


# Цикли

Іноді виникає необхідність повторного виконання певних команд.

Щоб не засмічувати код однаковими інструкціями, використовують **цикли**.

Це **команди**, які дозволяють багато разів повторювати певні інструкції.



The screenshot shows the Thonny Python IDE interface. The main editor window displays a file named `cycle.py` with the following code:

```
1 print("Який чудовий день!")
2 print("Який чудовий день!")
3 print("Який чудовий день!")
4 print("Який чудовий день!")
5 print("Який чудовий день!")
6 print("Який чудовий день!")
```

Below the editor is the Shell window, which shows the output of running the script:

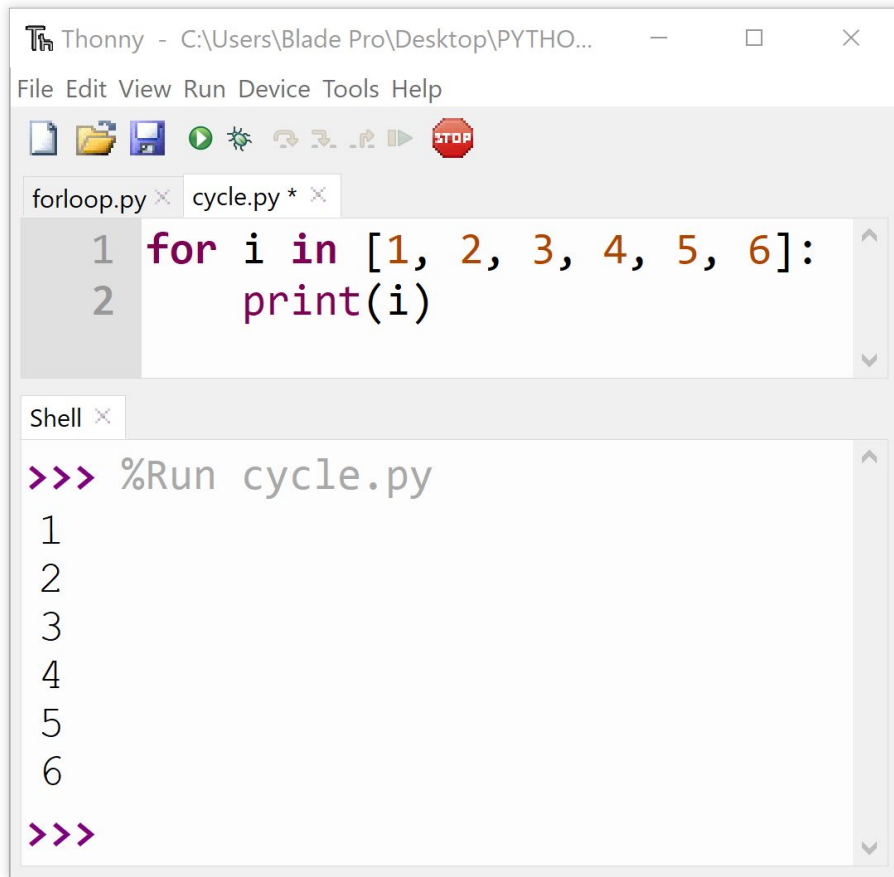
```
>>> %Run cycle.py
Який чудовий день !
Який чудовий день !
Який чудовий день !
Який чудовий день !
Який чудовий день !
Який чудовий день !
>>>
```

# Цикл for

Цикл з оператором **for** використовується тоді, коли якусь команду треба виконати певну кількість разів.

Для цього використовують конструкцію:

**for** назва змінної **in** набір даних  
(послідовність чисел, діапазон, список тощо) : (двокрапка)  
**команди, які треба повторити**



The screenshot shows the Thonny Python IDE interface. The title bar indicates the file path: C:\Users\Blade Pro\Desktop\PYTHO... The menu bar includes File, Edit, View, Run, Device, Tools, and Help. The toolbar contains icons for file operations and execution. The editor window shows a file named 'cycle.py' with the following code:

```
1 for i in [1, 2, 3, 4, 5, 6]:
2     print(i)
```

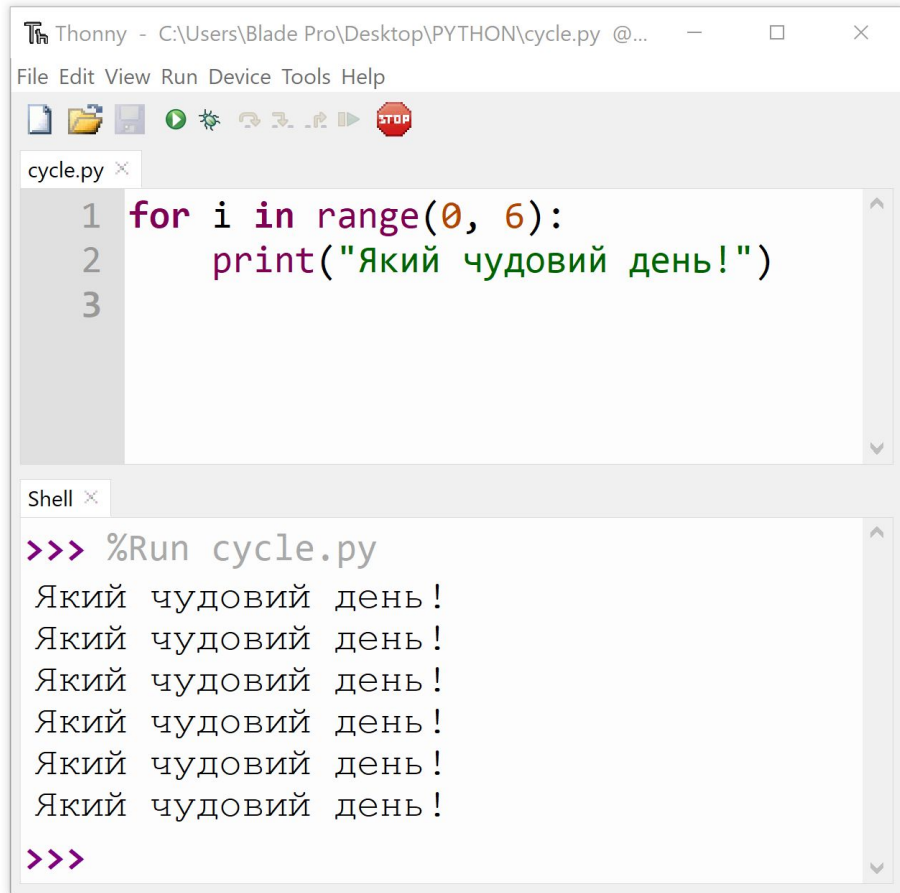
The Shell window shows the command to run the file and the output:

```
>>> %Run cycle.py
1
2
3
4
5
6
>>>
```

# Цикл for

Потрібну кількість повторів можна визначити за допомогою функції **range()**, де вказується діапазон від першого до числа перед останнім.

Наприклад у циклі **for i in range(0, 6)** фактично задано діапазон від 0 до 5 (6-1), тому буде лише шість повторів: 0,1,2,3,4,5.



The screenshot shows the Thonny Python IDE interface. The top window, titled 'Thonny - C:\Users\Blade Pro\Desktop\PYTHON\cycle.py @...', contains a Python script in 'cycle.py':

```
1 for i in range(0, 6):
2     print("Який чудовий день!")
3
```


The bottom window, titled 'Shell', shows the execution of the script:

```
>>> %Run cycle.py
Який чудовий день!
Який чудовий день!
Який чудовий день!
Який чудовий день!
Який чудовий день!
Який чудовий день!
>>>
```



# Практична робота

- 1) Створіть новий файл (**New**)
- 2) Створіть програму з використанням оператора **for**, щоб вивести у вікно **Shell**  
стовпчик чисел від 1 до 100. Збережіть файл (**Save**) під назвою **forloop.py**



The image shows a screenshot of the Thonny Python IDE. The title bar indicates the file path is C:\Users\Blade Pro\Desktop\PYTHON\forloop.py. The menu bar includes File, Edit, View, Run, Device, Tools, and Help. The toolbar contains icons for file operations and execution. The editor window, titled 'forloop.py', contains the following Python code:

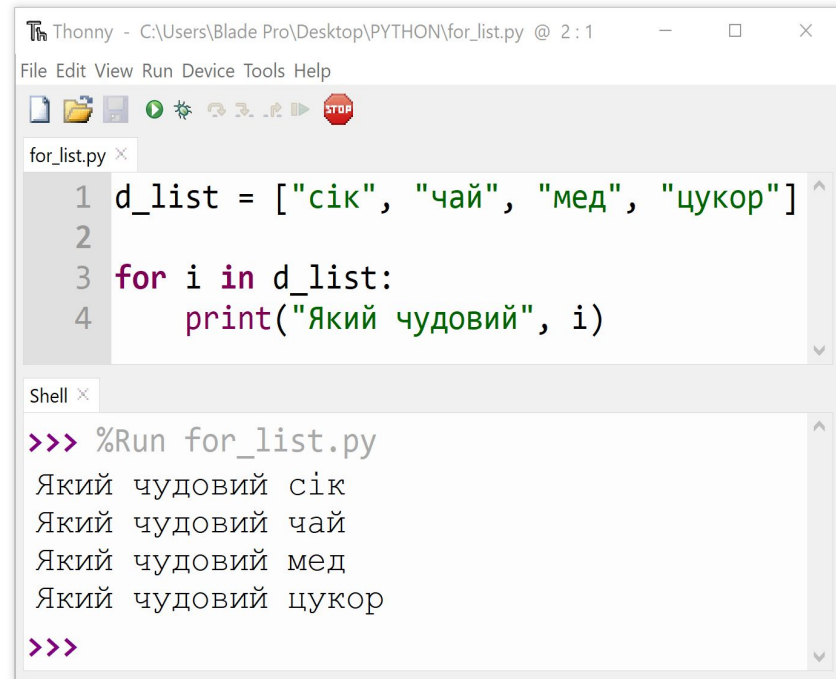
```
1 for i in range(1, 101):  
2     print(i)
```

Below the editor is a 'Shell' window showing the output of the script, which is the numbers 90 through 100, each on a new line. At the bottom of the shell window, the prompt '>>>' is visible.

Якщо все зроблено правильно - ви побачите такий результат

# Цикл for та списки

Цикл з оператором **for** можна використовувати для виведення значень, що зберігаються у списках.



The screenshot shows the Thonny IDE interface. The top window, titled 'for\_list.py', contains the following Python code:

```
1 d_list = ["сік", "чай", "мед", "цукор"]
2
3 for i in d_list:
4     print("Який чудовий", i)
```

Below the code editor is a 'Shell' window showing the output of running the script:

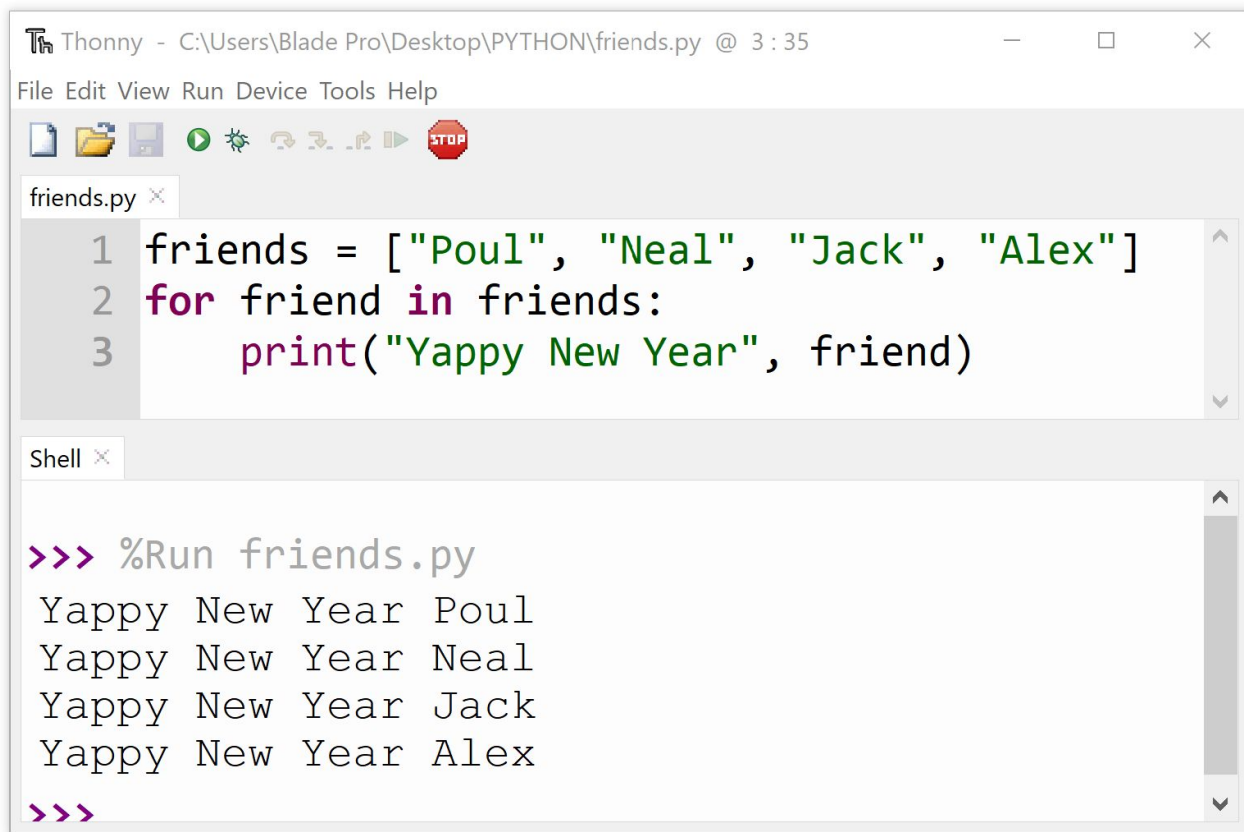
```
>>> %Run for_list.py
Який чудовий сік
Який чудовий чай
Який чудовий мед
Який чудовий цукор
>>>
```



# Практична робота

- 1) Створіть новий файл (**New**)
- 2) Створіть список з іменами друзів та збережіть файл (**Save**) під назвою **friends.py**
- 3) За допомогою оператора **for** привітайте усіх друзів зі списку з Новим роком.
- 4) Виведіть результат роботи програми у вікно **Shell** за допомогою функції **print()**.





The image shows a screenshot of the Thonny Python IDE. The title bar indicates the file path is C:\Users\Blade Pro\Desktop\PYTHON\friends.py. The menu bar includes File, Edit, View, Run, Device, Tools, and Help. The toolbar contains icons for opening files, saving, running, and stopping. The main editor window shows a Python script named friends.py with the following code:

```
1 friends = ["Poul", "Neal", "Jack", "Alex"]
2 for friend in friends:
3     print("Yappy New Year", friend)
```

Below the editor is a Shell window showing the output of running the script:

```
>>> %Run friends.py
Yappy New Year Poul
Yappy New Year Neal
Yappy New Year Jack
Yappy New Year Alex
>>>
```

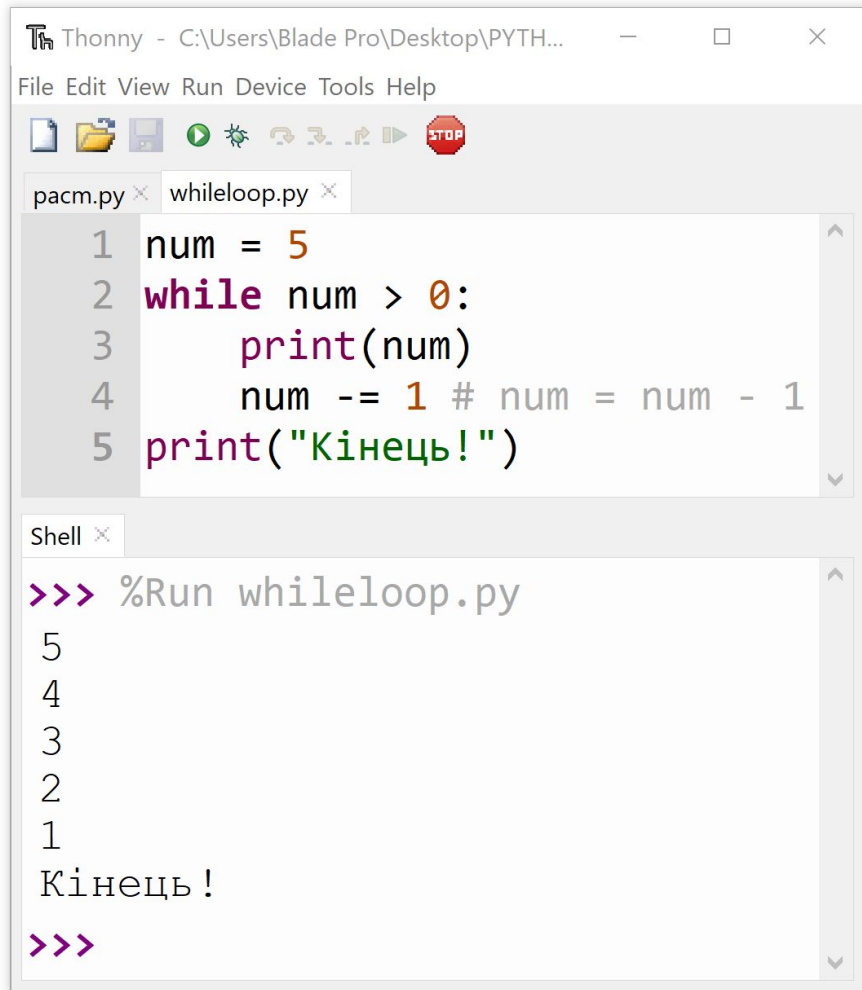
Якщо все зроблено правильно - ви побачите такий результат

## Цикл while

Іноді необхідно повторювати команду, доки не буде досягнуто певної умови, незважаючи на кількість повторів.

Для таких випадків використовують цикл з оператором **while**.

Програма у циклі буде виконуватись допоки твердження у циклі **while** **вірне**. якщо твердження буде **невірне** - цикл **припиняється** і переходить до наступних команд.



The screenshot shows the Thonny Python IDE interface. The main editor window displays a Python script named `whileloop.py` with the following code:

```
1 num = 5
2 while num > 0:
3     print(num)
4     num -= 1 # num = num - 1
5 print("Кінець!")
```

Below the editor is a Shell window showing the execution output:

```
>>> %Run whileloop.py
5
4
3
2
1
Кінець !
>>>
```

З циклом **while** треба бути дуже обережним.

Якщо твердження у циклі **while** завжди буде **вірним**, команди у циклі будуть виконуватись **нескінченно**, допоки вистачить потужностей комп'ютера для обробки та зберігання згенерованих даних.

The screenshot shows the Thonny IDE interface. The main editor window displays a Python script named `whileloop.py` with the following code:

```
1 num = 5
2 while num > 0:
3     print(num)
4     print("Кінець!")
```

Below the editor, there are three separate Shell windows. The first Shell window shows the command `>>> %Run whileloop.py` and the output of the program, which is the number 5 printed 10 times, followed by the text "Кінець!". The second and third Shell windows also show the output of the program, which is the number 5 printed 10 times, followed by the text "Кінець!".

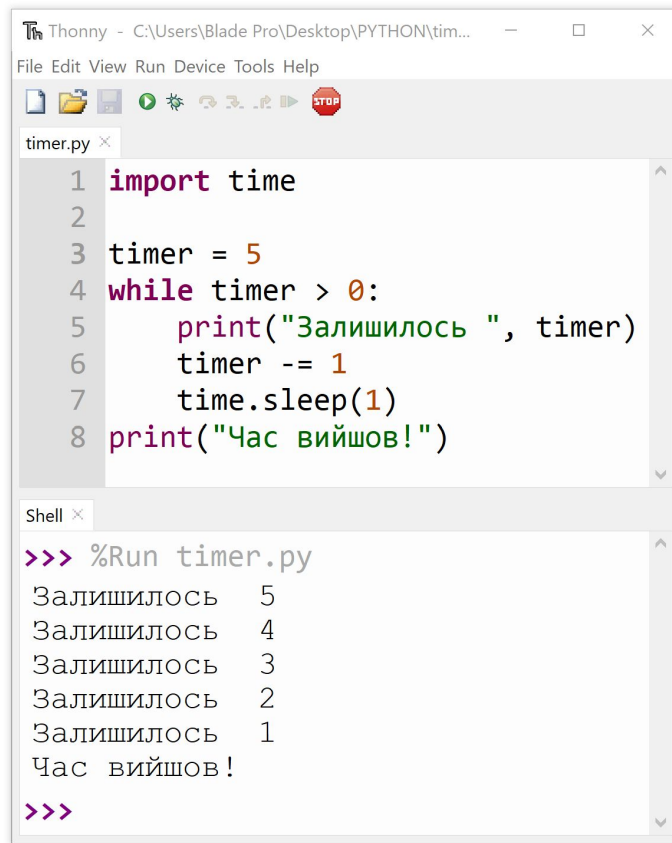


## Практична робота

- 1) Створіть новий файл (**New**)
- 2) Запишіть команди, наведені справа та збережіть файл (**Save**) під назвою **timer.py**
- 3) Виведіть результат роботи програми у вікно **Shell** за допомогою функції **print()**.

```
import time

timer = 5
while timer > 0:
    print("Залишилось ", timer)
    timer -= 1
    time.sleep(1)
print("Час вийшов!")
```



The image shows a screenshot of the Thonny Python IDE. The main window displays a Python script named `timer.py` with the following code:

```
1 import time
2
3 timer = 5
4 while timer > 0:
5     print("Залишилось ", timer)
6     timer -= 1
7     time.sleep(1)
8 print("Час вийшов!")
```

Below the code editor is a Shell window showing the execution output:

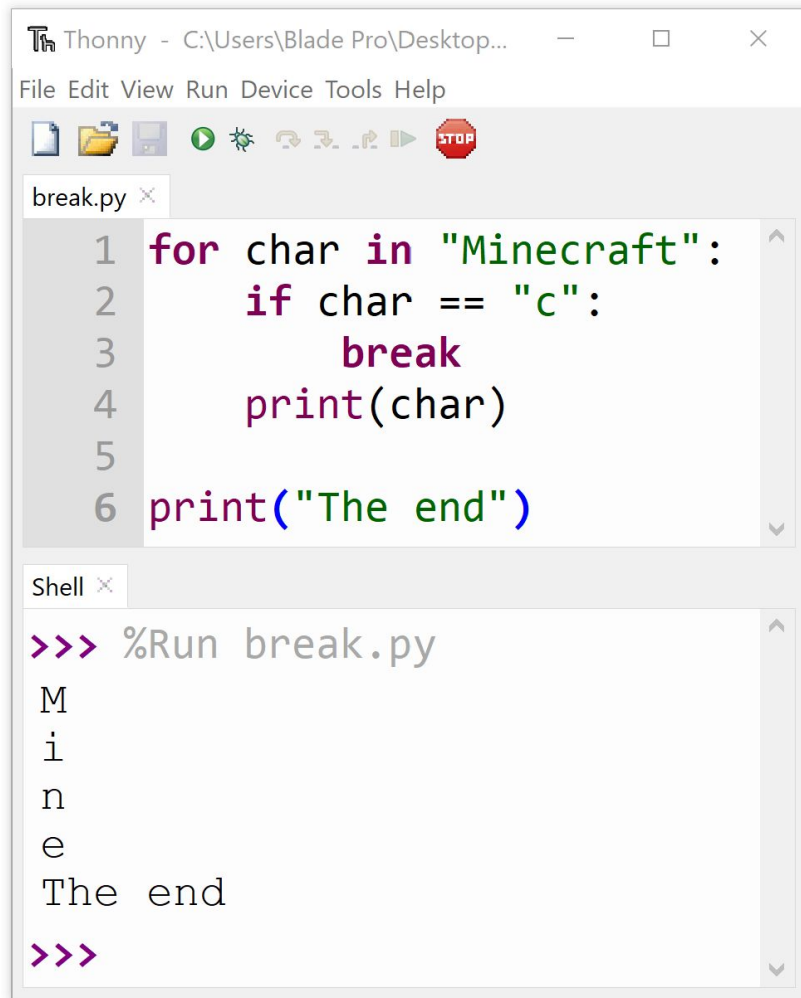
```
>>> %Run timer.py
Залишилось 5
Залишилось 4
Залишилось 3
Залишилось 2
Залишилось 1
Час вийшов!
>>>
```

Якщо все зроблено правильно - ви побачите такий результат

## Ключове слово **break**

це спосіб розірвати цикл, тобто  
МИТТЄВО ЗУПИНИТИ виконання  
програми в циклі.

Команда **break** працює з циклами **for**  
і **while**.



The screenshot shows the Thonny Python IDE interface. The title bar reads "Thonny - C:\Users\Blade Pro\Desktop...". The menu bar includes "File", "Edit", "View", "Run", "Device", "Tools", and "Help". The toolbar contains icons for file operations and execution. The editor window, titled "break.py", contains the following Python code:

```
1 for char in "Minecraft":
2     if char == "c":
3         break
4     print(char)
5
6 print("The end")
```

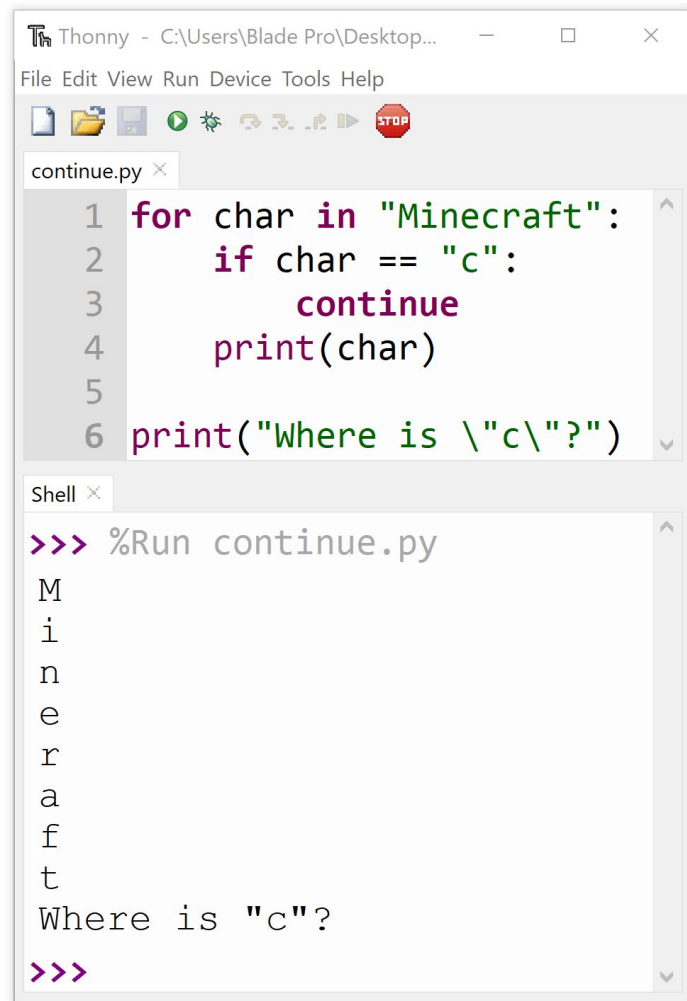
Below the editor is a "Shell" window showing the execution output:

```
>>> %Run break.py
M
i
n
e
The end
>>>
```

## Ключове слово **continue**

це спосіб розірвати цикл (пропустити ітерацію) під час виконання певної умови і після цього виконання циклу поновлюється.

Команда **continue** працює з циклами **for** і **while**.



```
Thonny - C:\Users\Blade Pro\Desktop...
File Edit View Run Device Tools Help

continue.py x
1 for char in "Minecraft":
2     if char == "c":
3         continue
4     print(char)
5
6 print("Where is \"c\"?")

Shell x
>>> %Run continue.py
M
i
n
e
r
a
f
t
Where is "c"?
>>>
```



## Підсумки

Познайомились з поняттям **цикли** у мові Python

Дізнались, як працювати з оператором **for**

Дізнались, як працювати з оператором **while**