



**Python Basic**

**Заняття 6: Function.**

# Структура заняття

- Що таке функція
- Параметри функції
- Ключове слово return
- Аргументи функції
- Значення аргументів за умовчанням
- Ключові аргументи – `print(..., end="-")`
- Перемінна кількість параметрів - `print(..., ..., ...)`
- Області видимості (`local`, `nonlocal`, `global`)
- Анотації

# Функція

Функції – це фрагменти програми, що багато разів використовуються. Вони дозволяють дати ім'я певному блоку команд для того, щоб згодом запускати цей блок за вказаним ім'ям у будь-якому місці програми і скільки завгодно багато разів. Це називається викликом функції.

**Функція** - підпрограма, яка виконує будь-які операції та повертає значення.

**Процедура** - підпрограма, яка виконує операції, без повернення значення.

**Метод** - це функція чи процедура, що належить класу чи екземпляру класу.

Назва функції

Ключове слово

```
def get_max_number(a, b):
```

Тіло функції

```
    if a > b:  
        return a  
    else:  
        return b
```

# Параметри функції

Функції можуть набувати параметри, тобто, деякі значення, що передаються функції для того, щоб вона щось зробила з ними.

```
def get_max_number(a, b):  
    if a > b:  
        return a  
    else:  
        return b
```



Параметри  
функції

# return

Оператор return використовується повернення з функції, тобто, для припинення її роботи та виходу з неї. При цьому можна також повернути деяке значення функції.

```
def get_max_number(a, b):  
    if a > b:  
        return a  
    else:  
        return b
```

Вихід з функції

A diagram consisting of two red arrows pointing from the text 'Вихід з функції' to the 'return a' and 'return b' lines in the code block. Each arrow points to a red rectangular box that highlights the respective return statement.

# Функція

- Аргументи VS параметри функції - `print("Hello")`
- Значення аргументів за умовчанням
- Перемінна кількість параметрів - `print(..., ..., ...)`
- Області видимості (`local`, `nonlocal`, `global`)
- Анотації



```
def get_max_number(a, b):
```

```
    if a > b:
```

```
        return a
```

```
    else:
```

```
        return b
```

Параметри




Аргументи



```
print(get_max_number(1, 2))
```

```
def get_max_number(a, b=10):  
    if a > b:  
        return a  
    else:  
        return b
```

Значення за  
замовчуванням



```
print(get_max_number(1, 2))  
print(get_max_number(1))
```

```
def get_max_number(a, b=10):  
    if a > b:  
        return a  
    else:  
        return b
```

Ключові аргументи

```
print(get_max_number(b=1, a=2))
```



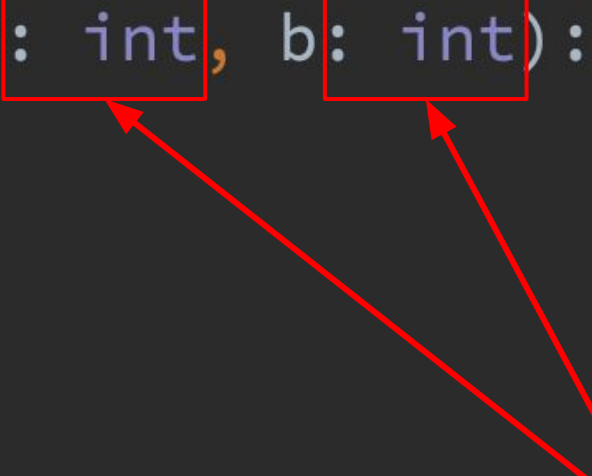
## Позиційні аргументи

```
def get_max_number(*numbers, **additional_params):  
    max_number = numbers[0]  
    for number in numbers:  
        if number > max_number:  
            max_number = number  
    return max_number  
  
print(get_max_number(1, 2, 3, 4, data='Hello'))
```

Ключові аргументи

The diagram illustrates the distinction between positional and keyword arguments in a function call. A red arrow points from the text 'Позиційні аргументи' to the list of numbers (1, 2, 3, 4) in the function call. Another red arrow points from the text 'Ключові аргументи' to the keyword argument 'data='Hello'' in the same call. In the function definition, the parameter '\*numbers' is highlighted with a red box, and a red arrow points from it to the list of numbers. The parameter '\*\*additional\_params' is also highlighted with a red box, and a red arrow points from it to the keyword argument.

```
def get_max_number(a: int, b: int):  
    if a > b:  
        return a  
    else:  
        return b
```



Анотації

```
print(get_max_number(1, 2))  
print(get_max_number('Hello', 2))
```

Expected type 'int', got 'str' instead



**lambda функції**