Лекция 4

Здравствуйте, уважаемые слушатели!
Тема нашей лекции – Функции в Python: определение, аргументы и возврат значений

План лекции:
1. Основы функций в Python
2. Аргументы функции
3. Расширенное использование функций
4. Ключевые аргументы
5. Заключение

Функции являются одним из фундаментальных концептов программирования на Python, позволяющие организовать код более эффективно и избежать повторения. В Python функции облегчают модульность и повторное использование кода, позволяя программистам эффективно разбивать сложные задачи на более простые подзадачи. Эта лекция охватывает определение функций в Python, передачу аргументов, возвращение значений, а также дополнительные концепции, такие как аргументы по умолчанию, ключевые аргументы и использование аргументов переменной длины.

Основы функций в Python

Определение функции

Функция в Python определяется с помощью ключевого слова def, за которым следует имя функции и круглые скобки, которые могут включать аргументы или оставаться пустыми. Блок кода внутри функции начинается с двоеточия и отступа. Простейшее определение функции выглядит следующим образом:

python
Копировать код
```
def my_function():
    print("Hello, World!")
```

Вызов функции происходит по имени с круглыми скобками, что приводит к выполнению её кода:

python
Копировать код
```
my_function()  # Вывод: Hello, World!
```

Аргументы функции

Аргументы функции — это переменные, которые передаются в функцию при её вызове. Они позволяют функции получать входные данные для обработки. Определение функции с аргументами выглядит следующим образом:

python
Копировать код
```
def print_name(name):
    print(f"Hello, {name}!")
```

Эта функция принимает один аргумент и использует его внутри тела функции. Вызов функции с передачей аргумента:

python
Копировать код
```
print_name("Alice")  # Вывод: Hello, Alice!
```

Возвращение значений

Для того чтобы функция могла вернуть значение, используется ключевое слово return. Возвращаемое значение может быть любого типа и может использоваться в программе, как любое другое значение. Например:

python
Копировать код
```
def sum_two_numbers(a, b):
    return a + b

result = sum_two_numbers(5, 3)
print(result)  # Вывод: 8
```

Расширенное использование функций

Аргументы по умолчанию

Функции в Python могут иметь аргументы со значениями по умолчанию. Эти значения используются, если при вызове функции соответствующий аргумент не предоставлен:

python
Копировать код
```
def print_greeting(name, greeting="Hello"):
    print(f"{greeting}, {name}!")

print_greeting("Alice")             # Вывод: Hello, Alice!
print_greeting("Alice", "Goodbye")  # Вывод: Goodbye, Alice!
```

Ключевые аргументы

Python позволяет функциям принимать аргументы по имени. Это означает, что при вызове функции можно явно указать, какому аргументу какое значение присваивается:

python
Копировать код
```
def describe_pet(animal, name):
    print(f"I have a {animal} named {name}.")

describe_pet(name="Harry", animal="hamster")
```

Аргументы переменной длины

Иногда функция должна принимать неопределенное количество аргументов. Это достигается с помощью символов * для неименованных аргументов и ** для именованных аргументов:

python
Копировать код
```
def make_pizza(*toppings):
    print("Making a pizza with the following toppings:")
    for topping in toppings:
        print(f"- {topping}")

make_pizza('pepperoni', 'cheese', 'mushrooms')
```

Заключение

Функции в Python обеспечивают мощные возможности для организации и управления кодом, делая его более читаемым, эффективным и легко поддерживаемым. Они позволяют реализовывать различные задачи программирования, упрощая сложные процессы и повышая повторное использование кода. Понимание и правильное использование функций является ключевым аспектом создания эффективных программ на Python.

Литературы:

1. Программирование на Python для начинающих. Райтман М.А. Москва-ЭМОСКО-2015 стр 39-48
2. Python для «чайников». Джон Полль Мюллер. Диалектика-2022 стр. 31-39