Информатика. Введение в Python

Берленко Татьяна Андреевна,
Шевская Наталья Владимировна,
СПбГЭТУ “ЛЭТИ”, ФКТИ, МОЭВМ

Некоторые базовые типы и структуры данных

Типы данных

- Числа
- Строки
- None
- Логические значения

Структуры данных

- Списки
- Кортежи
- Словари
- Множества

Как можно запустить программу

Также вы можете написать программу в файле, который называется модуль. Модуль - это файл с расширением **.py**.

```
  GNU nano 2.5.3                Файл: hello.py

print('Hello, world!')
```

После того, как вы написали программу в модуле hello.py, вы можете ее запустить.

```
tatyana@tatyana-ThinkPad-T480s:~$ python3 hello.py
Hello, world!
```

DEMO

Функции. Основные сведения

- Функция - это подпрограмма, к которой можно обратиться из другого места программы.

Плюсы:

- Многократное использование кода ⇒ отсутствие избыточности
- Декомпозиция программы

Функции. Основные сведения

Функцию можно:

- вызвать, т.е. использовать;
- определить, т.е. написать последовательность действий, которую будет выполнять функция.

Функции. Синтаксис вызова

- Общий вид вызова функции:

<имя_функции>**(**<аргумент_1>, <аргумент_2>, … <аргумент_n>**)**

Функции. Примеры вызова

- input()

Функция может принимать на вход текст-сообщение для пользователя, например:

```
name = input("Hello! Please, type your name!\n")
```

Функция возвращает **строку**.

Функции. Примеры вызова

- print() # вывод информации на консоль.

Примеры:

```
user@user-pc:~$ python3
>>> print('aaa')
aaa
>>> print('aaa', 'bbb')
aaa bbb          <- отделены пробелами
>>> print('age:', 13)
age: 13          <- различные типы данных
```

Введение в ООП. Объект

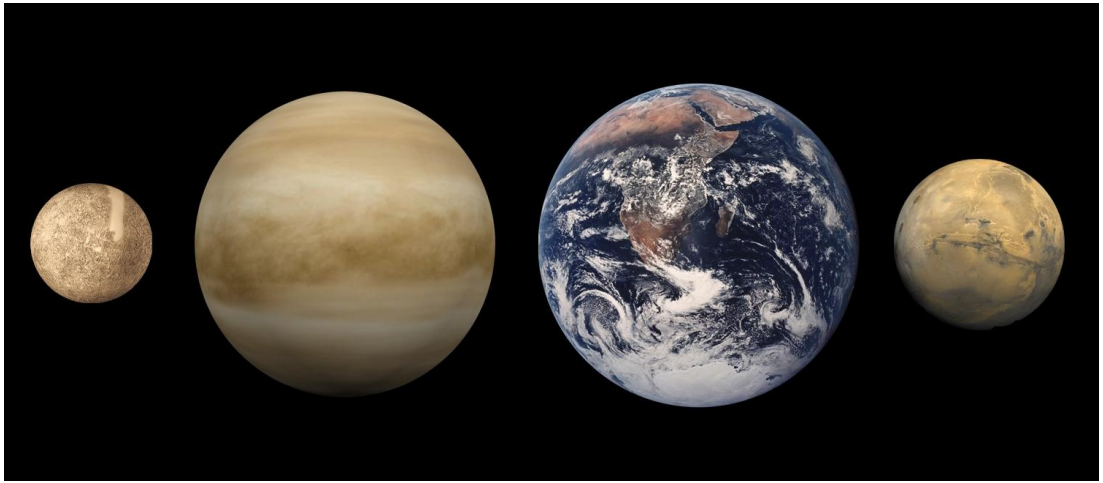
Объект - конкретная сущность предметной области.

Введение в ООП. Класс

Класс - это **тип** объекта. Или говорят, что объект - это экземпляр класса.

Класс: **Планета**.

Объекты: **Меркурий, Венера, Земля, Марс**.



Введение в ООП. Метод

Метод - функция, принадлежащая классу.

Например: класс **Планета** содержит метод

`получитьСреднийРадиус()`

- Для объекта **Венера** метод вернет 6051,8 км
- Для объекта **Земля** метод вернет 6371,0 км
- Для объекта **Марс** метод вернет 3389,5 км

Введение в ООП. Метод

Синтаксис вызова метода:

<Объект>.<Имя_метода>(аргумент_1, аргумент_2, ...)

Пример вызова:

Венера.получитьСреднийРадиус()

После вызова такого метода мы получим значение 6051,8 км.

Немного о числах

int	Целые числа неограниченной точности
float	Числа с плавающей точкой
complex	Комплексные числа

DEMO

Немного о числах

Представление	Описание
111111111111111	Целое число
2.1e-1, 2E3, 0.123	Вещественное число
0o157, 0x9f, 0b10111	Восьмеричное, шестнадцатеричное и двоичное число
3+4.8j, 22j	Комплексное число

Операции
над целыми и вещественными числами

Оператор	Описание
+, -	Сложение, Вычитание
*, /, //, %	Умножение, Деление, Деление с округлением вниз, Остаток от деления
x ** y	Возведение в степень: x^y

Логический тип

- Логический тип bool
- Значения True, False
- True == 1
- False == 0

Логические операторы

Оператор	Описание
or	Логическое ИЛИ
and	Логическое И
not	Логическое отрицание

Операторы сравнения

Оператор	Описание
x < y, x <= y, x > y, x >= y	Операторы сравнения
x == y, x != y	Операторы проверки на равенство

Выражение вида: x < y < z

Будет интерпретироваться так: x < y and y < z

Строки

- Класс str
- Неизменяемый объект
- Для инициализации можно использовать одинарные и двойные кавычки: ‘qwerty’ == “qwerty”

Доступ по индексу

- my_str = ‘AQBWCFD’
- my_str[index] # доступ по индексу

A	Q	B	W	C	F	D
0	1	2	3	4	5	6
-7	-6	-5	-4	-3	-2	-1

Извлечение срезов

- my_str = ‘AQBWCFD’
- my_str[i:j:k] - извлечение среза из my_str. Извлекаются символы от **i** до **j-1** с шагом **k**.
- По умолчанию i = 0, j - длина строки, k = 1.
- Пример: my_str[2:4] = “BW“

A	Q	B	W	C	F	D
0	1	2	3	4	5	6

Извлечение срезов

- my_str = ‘AQBWCFD’
- При **k<0** порядок использования i и j должен быть изменен на противоположный: mystr[j:i:k]
- Пример: my_str[::-1] извлечет все элементы my_str в обратном порядке.

A	Q	B	W	C	F	D
-7	-6	-5	-4	-3	-2	-1

Операторы сравнения

Оператор	Описание
in, not in	Проверка вхождения
x < y, x <= y, x > y, x >= y	Операторы сравнения
x == y, x != y	Операторы проверки на равенство

DEMO

Приведение типов

int()	Приведение к целому числу
float()	Приведение к числу с плавающей точкой
bool()	Приведение к объекту логического типа
str()	Приведение к строке

Списки

- Список - набор упорядоченных разнородных объектов
  - Пример: example_list = [‘hello’, 1, 2, [‘world’, ‘!’]]
- Изменяемый объект
- Произвольное число уровней вложенности:

matrix = [

[0, 1, [2], 3, 4], [0, [1, 2], 3, 4],

[0, 1, [2, 3], 4], [0, 1, 2, 3, 4]]

DEMO списки

Словари

- Словарь - неупорядоченный набор элементов, доступ в котором осуществляется по ключу.

```
countries =
{
    "Russia": "Moscow",
    "Finland": "Helsinki",
    "USA": "New York",
    "USA": "Washington, D.C."

 }
```

```
countries["Russia"]
# "Moscow"
```

DEMO словари

Многомерные словари

```
subjects = {
        "groups": {
                "303": ["Computer Science", "Programming"],
                "310": ["History"]
                },
        "classrooms": {
                        "Computer Science": "10",
                        "Programming": "20",
                        "History": "30",
                        }
    }
```

Оператор ветвления

```
if <выражение_1>:
    <инструкции>
elif <выражение_2>:
    <инструкции>
else:
    <инструкции>
```

Базовые конструкции языка

- **break** - прекращает выполнение цикла
- **continue** - прекращает выполнение итерации цикла

Цикл while


```
while <тест1>:
    <инструкции>
    if <тест2>: break # необязательная часть
    if <тест3>: continue # необязательная часть
else: # необязательная часть
    <инструкции>
```

Цикл for


```
for <переменная> in <объект>:
    <инструкции>
    if <усл2>: break # необязательная часть
    if <усл3>: continue # необязательная часть
else:  # необязательная часть
    <инструкции>
```

DEMO for со “счетчиком”

Возможные ошибки

```
groups = {"901":"Иванов", "902":"Петрова"}
```

Нельзя менять размер словаря во время итерирования словаря с помощью цикла.

```
for i in groups:
    groups["903"] = "Антонова"
```

```
for i in groups:
    groups.pop("901")
```

Определение функции

- def <имя_функции>(<аргументы>):

  <инструкции>

- return
  - можно возвращать несколько переменных
- pass или ...

Варианты определения функции

- def func($name_1$, $name_2$, … $name_n$):
- Определение функции с значениями по умолчанию:

  def func($name_1$=val_1, $name_2$=val_2, … $name_n$=val_n):
- Сначала должны следовать аргументы функции **без** значения по умолчанию.

Вызов функции

- Вызов функции с позиционными аргументами:

  $func(val_1, val_2, \dots val_n)$

- Вызов функции с именованными аргументами:

  $func(name_1=val_1, name_2=val_2, \dots name_n=val_n)$

- Сначала должны следовать **позиционные** аргументы.

Вывод данных

- print([arg$_1$, ...][, sep=' '][, end='\n'][, file=sys.stdout])
- [arg$_1$, ...] # позиционные аргументы
- [, sep=' '] # именованный аргумент
- [, end='\n']# именованный аргумент
- [, file=sys.stdout] # именованный аргумент
- Функция возвращает **None.**

DEMO print()

Модули стандартной библиотеки

- Модуль - файл с программой.
- Модуль может импортировать другие модули для доступа к функциям и переменным.
- Импорт всего модуля:

  `import <имя модуля>`
- Импорт некоторых имен из модуля:

  `from <имя_модуля> import <имена>`

DEMO

Импорт собственных модулей

- Код модуля выполняется при импорте.
- У любого модуля python есть атрибут __name__.
- Если модуль запускается как главный модуль программы, атрибут __name__ хранит значение “__main__”.
- Если модуль импортируется, атрибуту __name__ присваивается имя модуля.

Источники

- https://docs.python.org/3/ Документация Python 3
- https://docs.python.org/3/reference/expressions.html#operator-precedence

  Таблица приоритетов операторов

Вопросы по курсу задавайте по почте:

Шевская Наталья Владимировна
natalya.razmochaeva@moevm.info

Правила коммуникации по электронной почте:
http://se.moevm.info/doku.php/inf:communication_rules