Прикладное программирование

Язык программирования Python

Козлов Даниил Александрович
н.к. ауд. 607
по всем вопросам в Discord

Hello, Python

Python — высокоуровневый интерпретируемый язык программирования общего назначения, ориентированный на повышение производительности разработчика и читаемости кода.

Основные особенности:

- динамическая типизация
- автоматическое управление памятью
- полная интроспекция
- встроенный механизм обработки исключений
- поддержка многопоточных вычислений
- высокоуровневые структуры данных
- модульность

ОСОБЕННОСТИ ЯЗЫКА ПРОГРАММИРОВАНИЯ PYTHON

PYTHON ПРОСТ В ИСПОЛЬЗОВАНИИ

ЕДИНЫЙ СТИЛЬ НАПИСАНИЯ КОДА ДЛЯ ВСЕХ – СОГЛАШЕНИЕ PEP

PYTHON – ГИБКИЙ ЯЗЫК

PYTHON – ДИНАМИЧЕСКИЙ ЯЗЫК

PYTHON – ИНТЕРПРЕТИРУЕМЫЙ ЯЗЫК

PYTHON – РАСШИРЯЕМЫЙ ЯЗЫК

PYTHON – «СКЛЕИВАЮЩИЙ» ЯЗЫК.

PYTHON – КРОСС-ПЛАТФОРМЕННЫЙ ЯЗЫК.

ЧТО НЕОБХОДИМО ДЛЯ КУРСА

Установленный интерпретатор Python 3.11

https://www.python.org/downloads/

Установленная IDE VSCode

Visual Studio Code

Установленный менеджер пакетов Conda

https://docs.conda.io/projects/conda/en/latest/user-guide/install/windows.html

ТИПЫ ДАННЫХ В ЯП PYTHON

ТИПЫ ДАННЫХ В PYTHON

К основным встроенным типам относятся:

- None (неопределенное значение переменной)
- Логические переменные (Boolean Type)
- Числа (Numeric Type)
  - int – целое число
  - float – число с плавающей точкой
  - complex – комплексное число
- Списки (Sequence Type)
  - list – список
  - tuple – кортеж
  - range – диапазон
- Строки (Text Sequence Type)
  - str
- Бинарные списки (Binary Sequence Types)
  - bytes – байты
  - bytearray – массивы байт
  - memoryview – специальные объекты для доступа к внутренним данным объекта через protocol buffer

ТИПЫ ДАННЫХ В PYTHON

- Множества (Set Types)
  - set – множество
  - frozenset – неизменяемое множество
- Словари (Mapping Types)
  - dict – словарь

Эти типы данных можно, в свою очередь, классифицировать по нескольким признакам:

- изменяемые (списки, словари и множества)
- неизменяемые (числа, строки и кортежи)
- упорядоченные (списки, кортежи, строки и словари)
- неупорядоченные (множества)

Динамическая типизация

Динамическая типизация — приём, используемый в языках программирования и языках спецификации, при котором переменная связывается с типом в момент присваивания значения, а не в момент объявления переменной. Таким образом, в различных участках программы одна и та же переменная может принимать значения разных типов.

```
>>> a = 2000
>>> type(a)
<class 'int'>
>>> a = 'Dynamic Typing'
>>> type(a)
<class 'str'>
>>> a = [1, 2, 3]
>>> type(a)
<class 'list'>
```

СИНТАКСИС ЯП PYTHON

Комментарии

Комментарии в Python могут быть однострочными:

```
# Очень важный комментарий
a = 10
b = 5 # Очень нужный комментарий
```

При необходимости написать несколько строк с комментариями, можно сделать многострочный комментарий:

```
"""
Очень важный
и длинный комментарий
"""
a = 10
b = 5
```

Арифметические операции

- Сложение
- Вычитание
- Умножение
- Деление
- Получение целой части от деления
- Получение остатка от деления
- Возведение в степень

Библиотека (модуль) math

Представление чисел в других системах счисления

Битовые операции

Работа с комплексными числами

Условный оператор ветвления if

```
if выражение_1:
    инструкции_(блок_1)
```

```
if выражение_1:
    инструкции_(блок_1)
else:
    инструкции_(блок_4)
```

```
if выражение_1:
    инструкции_(блок_1)
elif выражение_2:
    инструкции_(блок_2)
elif выражение_3:
    инструкции_(блок_3)
else:
    инструкции_(блок_4)
```

Оператор цикла while

```
while выражение:
    инструкция_1
    инструкция_2
    ...

    инструкция_n
```

```
a = 0
while a >= 0:
   if a == 7:
       break
   a += 1
   print("A")
```

Оператор **break** предназначен для досрочного прерывания работы цикла while.
Оператор **continue** запускает цикл заново, при этом код, расположенный после данного оператора, не выполняется.

Оператор цикла for

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

```
lst = [1, 3, 5, 7, 9]
for i in lst:
    print(i ** 2)
```

```
word_str = "Hello, world!"
for l in word_str:
    print(l)
```

Внутри тела цикла можно использовать операторы break и continue, принцип работы их точно такой же как и в операторе while.