Курс по Python

Урок №1

“Присваивание”

# рассмотрим присваивание значений в Python  
# присваивание происходит последовательно справа налево  
x = y = 0  
print(x)  
print(y)  
  
y1 = 5  
x1 = y1 = 7  
  
print(x1)  
print(y1)

Урок №2

“Логический тип”

# рассмотрим логический тип данных  
# возвращает true или false  
print(13 == 13)  
  
# в Python присутствует сножественное сравнение  
x = 2  
print(1 < x < 3)  
  
# логические операторы  
# логические операторы ленивые если возможно  
# вернуть раньше значение из функции, чем закончится выполнение выражения  
# то возвращаем результат  
x = False  
y = True  
  
print(x and y or x and (not y))  
print(False and True)  
  
# задача  
year = 2017  
is\_leap = year % 4 == 0 and (year % 100 != 0 or year % 400 == 0)  
print(is\_leap)

Урок №3

Строки

# рассмотрим работу со строками как их задавать  
# и так же как их экранировать  
  
# Сырые строки обозначаются r  
# и убирают экранирование символов, оставляя их как  
# просто символы в строке  
example\_str = r"C:\\work\\"  
print(example\_str)  
  
# строки можно переносить  
# для этого используем обратный слеш  
str2 = "this text is this text " \  
 "this text is this text" \  
 "this text is this text"  
print(str2)  
  
# для написания большого блока кода  
  
str3 = """  
 this text is this text   
 this text is this text  
 this text is this text  
"""  
print(str3)  
  
# строки можно объединять  
# строки неизменяемые при сложении двух строк  
# создаётся новый объект в памяти  
# это можно узнать по адресу в памяти  
str4 = "str4"  
print(id(str4))  
str4 += 'a'  
print(id(str4))  
  
# срезы строк  
# [start:stop:step]  
# срез может идти справа налево или  
# слева направо главное помнить что он начнётся с элемента  
# start, по умолчанию шаг 1, если указать только шаг -1  
# то массив перевернётся  
  
example\_str = "Курс по Python на Coursera"  
  
print(example\_str[9:])  
print(example\_str[9:15])  
print(example\_str[-8:-2])  
print(example\_str[::-1])

# методы строк  
quote = "покажите мне код"  
  
# метод нахождения колличества букв в строке  
print(quote.count('о'))  
  
# сделать первую букву большой  
print(quote.capitalize())  
  
# позволяет проверить является ли строка числом  
tempStr = '2017'  
print(tempStr.isdigit())  
  
# оператор in позволяет проверить содержится ли  
# подстрока в строке  
tempStr = 'Hello'  
print('lo' in tempStr)  
  
# преобразования строк  
num\_string = str(999.01)  
print(num\_string, type(num\_string))  
  
# иттерация по строке  
for elem in num\_string:  
 print(elem, end="")

# форматирование строк  
  
# передаём placeholder  
str = "{} не лгут, а лгут {}".format('Цифры', "люди")  
print(str)  
  
# передам именованный placeholder  
# неименованные аргументы указать уже нельзя  
problem = "задач"  
str = "{num} Кб должно хватить для {problem}".format(  
 num = 640, problem=problem)  
print(str)  
  
# f-строки  
subj = "optimize"  
auth = 'Knuth'  
  
print(f"Преждевременная {subj} - плохо. Автор {auth}")  
  
  
# вывод чисел с модификаторами  
num = 2 / 3  
print(f"{num:.3f}")  
print("{num:.3f}".format(num=num))

# ввод и вывод строк  
#nameUser = input()  
#print(nameUser)  
  
# байтовые строки  
# Русские символы не поддерживает  
example\_str = b"hello"  
print(example\_str, type(example\_str))  
  
for elem in example\_str:  
 print(elem, end=" ")  
print()  
  
# метод encode для строки  
# поддерживает Русские символы  
# каждый символ кодируется двумя буквами  
example\_str = "привет"  
  
example\_str = example\_str.encode()  
print(example\_str)  
  
# метод decode  
example\_str = example\_str.decode()  
print(example\_str)

Урок №4

None

# рассмотрим особый тип None  
# его булевый тип всегда False как и у пустой строки  
income = None  
  
if income is None:  
 print("Ещё даже не продавали")  
elif not income:  
 print("Ничего не заработали")  
  
print(None is income)  
print((bool(income)))

Урок №5

Управление потоком

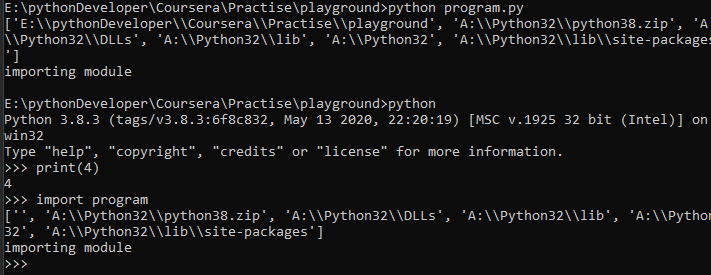
# управление потоком  
  
# после первого выполненного блока, выходит их цикла  
company = 'google.com'  
  
if 'com' in company:  
 print("good")  
elif 'goog' in company:  
 print('good1')  
else:  
 print('bad')  
  
# аналог тернарного оператора  
winner = 'Argentina' if 5 > 4 else 'Jamaica'  
print(winner)  
  
# оператор while  
indexLoop = 0  
while(indexLoop < 10):  
 indexLoop += 1  
  
# встроенный range  
# иттерируемся по целым числам  
# последнее число не входит  
# range(start, stop, step)  
for i in range(3):  
 print(i, end=" ")  
  
print()  
for i in range(1, 10, 2):  
 print(i, end=" ")  
  
print()  
for i in range(-5, -1, 1):  
 print(i, end=" ")

Урок №6

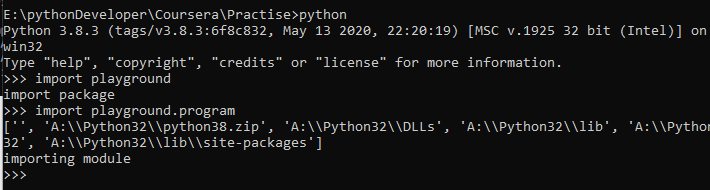
Модули и пакеты

import sys  
  
# чтобы узнать где python будет искать библиотеки   
# при импорте  
print(sys.path)

1. В python файлы можно вызывать из консоли, как в примере выше, но также можно вызывать файлы, как модули (пакеты), как показано во втором примере. При этом весь код, который может выполниться внутри файла будет выполнен сразу.



2. В python можно создать пакет, где в пакете находится множество модулей, так же в данном пакете обязательно должен быть файл \_\_init\_\_.py , который выполнится при импорте пакета, модули из пакета импортируются через точку.



3. В python код не выполняется напрямую из файла, сначала он преобразуется в оптимизированный байт код.

4. if \_\_name\_\_ == “\_\_main\_\_”:

<код>

Данный код позволяет выполнять код только, если он вызван, как главный фай, а не из модуля

5. Чтобы вызвать функцию из другого модуля используем псевдоним

import playground.utils as funck  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 print('main')  
 print(funck.multiply(2, 3))  
  
print("import is pack")

6. Чтобы импортировать функцию так же можно использовать синтаксис

from playground.utils import multiply  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 print('main')  
 print(multiply(2, 3))  
  
print("import is pack")

7.

import inspect  
import os  
  
# чтобы посмотреть где находятся модули в операционной системе  
# аргумент это имя модуля, который уже импортирован  
print(inspect.getfile(inspect))  
  
# чтобы посмотреть все файлы в директории  
# используем путь и следующую команду  
# где в качестве аргумента указываем путь  
print(os.listdir("A:/Python32/lib/"))

Урок №7

Виртуальное окружение

1. Для удобной работы с библиотеками python, перед началом работы (создания проекта) создаём виртуальное окружение.



2. Можно установить jupyter notebook, в него так же входит редактор кода iPython

pip install kupyter

2.1 Для запуска активируем окружение куда установили jupyter, а и прописываем команду

jupyter-notebook