Практическая работа № 2 от 21.02.2020

Основы Python

Цель работы

Изучение базового синтаксиса Python с использованием Jupyter notebook.

Задачи работы

1. Изучить основные типы данных и операции Python.
2. Научиться работать с Jupyter notebook.

Перечень обеспечивающих средств

1. ПК.
2. Учебно-методическая литература.
3. Задания для самостоятельного выполнения.

Общие теоретические сведения

Определения

Python – интерпретируемый высокоуровненвый язык программировани общего назначения. Обладает простым для изучения и использования синтаксисом. Активно развивается, имеется большое количество библиотек для работы с различными предметными областями.

Jupyter notebook – инструмент для итерационного выполнения кода на Python. Запускается в веб-браузере.

Binder (mybinder.org) – интернет-сервис для работы с копиями Jupyter notebook в режиме онлайн. Использует в качестве источника данных проект GitHub.

Переменные и вывод результатов Python

Для определения переменной достаточно выполнить код с указанием ее имени, например:

```
x
my_first_variable
```

Значение переменной присваивается с помощью оператора «=», например:

```
x = 0
my_first_variable = 'Привет!'
```

Можно сразу же определить перменную и присвоить ей значение, например:

```
x = 0
```

Для вывода значения переменной используется функция «print()», например:

```
print(x)
```

Типы данных и основные операции с ними в Python

В Python тип переменной определяется по ее значению. Ниже рассмотрены базовые типы данных.

Целые числа (int) задаются цифрами и, при необходимости, знаком минуса. Примеры целочисленных переменных:

```
x = 5
y = 100
z = -12
```

Дробные числа (float) задаются цифрами, десятичной точкой и, при необходимости, знаком минуса. Примеры дробных переменных:

```
x = 1.5
y = -0.125
z = 3.415926
```

Арифметические операции, т. е. операции с числами, с примерами:

Операция	**Оператор**	**Пример записи**	**Результат**
сложение	+	1 + 1	2
вычитание	-	8 - 0.5	7.5
умножение	*	2 * 5	10
деление	/	5.5 / 0.5	11
возведение в степень	**	2 ** 5	32
взятие по модулю (остаток от деления)	%	7 % 3	1
целочисленное деление	//	7 // 3	2

Порядок выполнения перечисленных арифметических операций:

1. Возведение в степень.
2. Умножение, деление, взятие по модулю и целочисленное деление.
3. Сложение и вычитание.

Для управления порядком выполнения операции используютсякруглые скобки.

Строки (str) задаются последовательностью символов заключенных в одинарные или двойные кавычки. Примеры строковых переменных:

```
x = "Привет!"
y = 'Кто здесь?'
z = "I like Python."
```

Операции со строками, с примерами:

Операция	**Оператор**	**Пример записи для** **x = "Привет!"** **y = 'Кто здесь?'**	**Результат**
конкатенация (сложение, слияние)	+	x + y	"Привет!Кто здесь?"
дублирование	*	x * 2	"Привет!Привет!"
получение символа по его индексу: 1) если индекс ≥ 0, то отсчёт идёт с начала строки, начиная с 0; 2) если индекс < 0, то отсчёт идёт с конца строки, начиная с -1.	[]	x[1] x[-3]	"p" "e"

Списки (list) – заключенная в квадратные скобки последовательность элементов любых типов (в том числе других списков), разделенных запятыми. Примеры списков:

```
[1, 'текст', 2, -0.19]
[[0, 1], 2, 'три']
```

Условия Python

Структура условного оператора (оператора ветвления):

```
if <Условие 1> :
   <Инструкция 1>

elsif <Условие 2> :
   <Инструкция 2>

elsif <Условие 3> :
   <Инструкция 3>

else:
   <Инструкция 4>
```

Elsif можно повторять столько раз, сколько требуется, либо не использовать вообще. При этом указывается очередное условие для проверки.
Else либо употребляется один раз, либо не используется. Инструкции в блоке else выполняются, если ни одно из вышеперечисленных условий не выполнено.

Возможные значения условий: истина — true и ложь — false.
Операции сравнения:

Операция	**Оператор**	**Пример записи для x = 1 и y = 2**	**Результат**
равно	==	x == y	false
не равно	!=	x != y	true
больше	>	x > y	false
меньше	<	x < y	true
больше или равно	>=	x >= y	false
меньше или равно	<=	x <= y	true

Логические операции:

Операция	**Оператор**	**Пример записи для x = true и y = false**	**Результат**
и	and	x and y	false
или	or	x or y	true
отрицание	not	not x	false

Пример:

```
x = 5
if x < 5 :
   print('Меньше пяти')
elssif x > 5 :
   print('Больше пяти')
else :
   print('Равно пяти')
```

Цикл for в Python
Цикл **for** может выполняться по списку или по диапазону чисел.
Структура цикла **for** по списку:

```
for <Элемент> in <Список> :
   <Инструкция>
```

Пример цикла for по списку:

```
x = [1, 2, 3, 4]
for e in x :
   print(e)
```

Структура цикла **for** по диапазону чисел:

```
for <Индекс> in range(<Начало>, <Остановка>, <Шаг>) :
   <Инструкция>
```

При этом функция range(x, y, z), где x, y и z — целые числа, создает диапазон целых чисел, такой что:

- первое число в диапазоне равно x,
- разность между последовательными числами равна z,
- последнее число
  - меньше y, если z > 0,
  - больше y, если z < 0.

При этом, если не указано значение z, то оно считается равным 1; если не указано значение x, то оно считаетсяч равным 0.

Например, range(0, 5, 1) сформирует диапазон 0, 1, 2, 3, 4.
range(2, 10, 2) — 2, 4, 6, 8.
range(5, 0, -1) — 5, 4, 3, 2, 1.
range(2, 6) – 2, 3, 4, 5.
range(4) – 0, 1, 2, 3.

Пример цикла for по диапазону чисел:

```
for i in range(4) :
    print(i)
```

Функции Python

Структура задания функции:

```
def <Имя функции>(<Список параметров>) :
        <Инструкции>
```

Если функция должна возвращать какое-либо значение, используется выражение:

```
return <Возвращаемое значение>
```

Например, задание функции, вычисляющей квадрат числа:

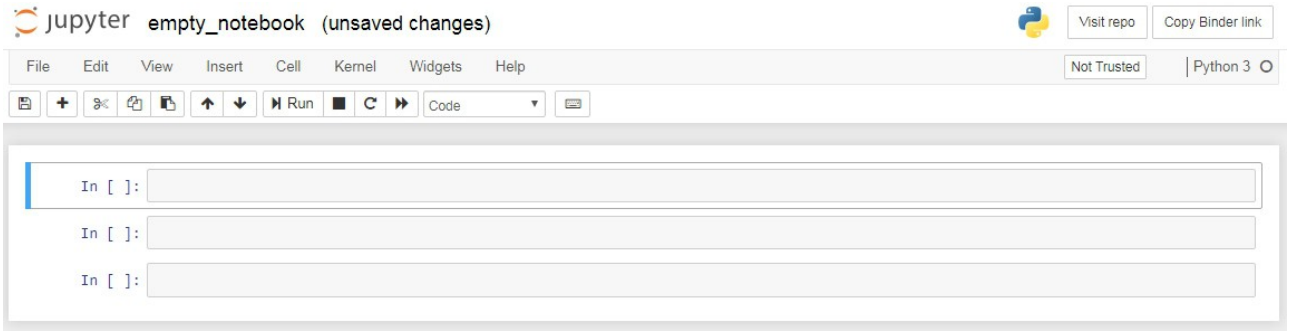
```
def squared(x) :
   return x ** 2
```

И вызов этой функции:

```
x = 2
print(squared(x))
```

Основные операции Jupyter notebook

Код может быть размешен и выполнен в нескольких ячейках:



Основные горячие клавиши:

Сочетание клавиш	**Действие**
Enter	Открыть выделенную ячейку на редактирование (синяя полоска слева становится зеленой).
Esc	Выйти из режима редактирования выделенной ячейки (зеленая полоска слева становится синей).
Курсор вверх	Выделить ячейку сверху.
Курсор вниз	Выделить ячейку снизу.
a	Создать ячейку над выделенной.
b	Создать ячейку под выделенной.
Двойное нажатие d	Удалить выделенную ячейку.
z	Отменить последнее удаление ячейки.
x	Вырезать выделенную ячейку.
c	Скопировать выделенную ячейку.
v	Вставить вырезанную или скопированную ячейку.
Shift + Enter	Выполнить код в выделенной ячейке и выделить следующую ячейку.
Ctrl + Enter	Выполнить код в выделенной ячейке.

Задание

Часть 1

- Сделайте форк репозитория https://github.com/mosalov/EmptyJupyterNoteebookForBinder
- Запустите Binder, нажав кнопку «launch binder». Кнопка доступна при просмотре файла «README.md».
- В открывшемся окне Binder дождитесь открытия репозитория.
- Откройте (кликните) файл «empty_notebook.ipynb».
- В ячейке для ввода введите «import this» и нажмите Ctrl+Enter или кнопку «Run».
- Загрузите файл на локальный компьютер: меню «File → Download as → Notebook (.ipynb)». Назовите файл «Задание 2_1.ipynb».
- Загрузите файл в репозиторий, созданный на первом шаге.

Часть 2

- Вернитесь к файлу «empty_notebook.ipynb», открытому в Binder.
- Создайте две целочисленных переменных *x* и *y*, присвойте им значения 10 и 25.
- Выведите сумму, разность, произведение и частное этих переменных.
- Рассчитайте значение выражения

$$\frac{(x^2+y)(y-1)}{x-y+y^2}$$

- Сохраните файл на локальном компьютере и в репозитори с названием «Задание 2_2.ipynb».

Часть 3

- Вернитесь к файлу «empty_notebook.ipynb», открытому в Binder.
- Задайте две перменные *x* и *y* со значениями 1 и 2.
- Реализуйте следующую логику: если X > Y, вывести текст «X больше Y»; если X < Y, вывести текст «X меньше Y»; если X = Y, вывести текст «X равно Y».
- Выполните написанный код.
- Сохраните файл на локальном компьютере и в репозитори с названием «Задание 2_3.ipynb».

Часть 4

- Вернитесь к файлу «empty_notebook.ipynb», открытому в Binder.
- Создайте список с элементами 2, 4, 6, 8, 10.
- Напишите цикл, который проходит по списку и сравнивает элемент с

числом 5. Если элемент больше 5, то он выводится. Если элемент меньше 5, то выводится противоположное ему число (например, для элемента «3» выводится «-3».

- Сохраните файл на локальном компьютере и в репозитори с названием «Задание 2_4.ipynb».

Контрольные вопросы

1. Реализуйте код, который по значению *n* вычисляет факториал *n*, т.е. *n! = 1 * 2 * … * (n-1) * n*.
2. Реализуйте код, который по значению *p* проверяет, является ли *p* простым числом.

Требования к отчету

Требуется представить отчет в виде письма на адрес mosalov.op@ut-mo.ru с указанием ссылки на репозиторий с сохраненными файлами Jupyter notebook.

Литература

1. https://pythonworld.ru/samouchitel-python
2. https://www.coursera.org/learn/diving-in-python
3. https://stepik.org/course/67/promo
4. http://pythontutor.ru/