МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего образования

«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Кафедра инфокоммуникаций

Лабораторная работа 3.1

Работа с IPython и Jupyter Notebook

Выполнил студент группы ИВТ-б-о-21-1

Пентухов С. А. « »_____20__г.

Подпись студента_____

Работа защищена « »_____20__г.

Проверил Воронкин Р.А. _____

(подпись)

Цель работы: исследовать базовые возможности интерактивных оболочек IPython и Jupyter Notebook для языка программирования Python.

Ход работы

1. Вывод изображений в ноутбуке

```
In [4]: from matplotlib import pylab as plt
        %matplotlib inline
        x = [i for i in range(50)]
        y = [i**2 for i in range(50)]
        plt.plot(x, y)
Out[4]: [<matplotlib.lines.Line2D at 0x1eb36f091c0>]
```

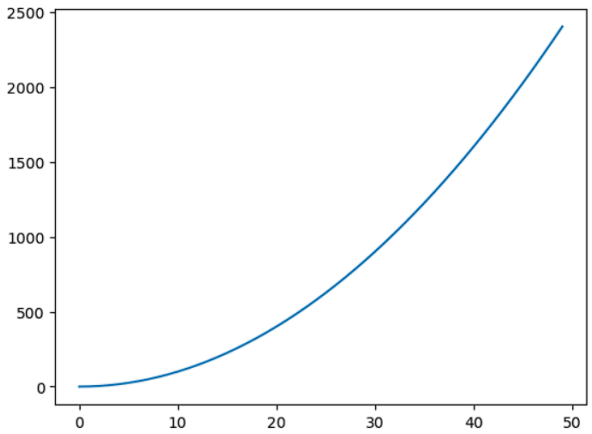


Рис. 1 Вывод графика

2. Использования команды %lsmagic

```
In [5]: %lsmagic
Out[5]: Available line magics:
%alias  %alias_magic  %autoawait  %autocall  %automagic  %autosave  %bookmark  %cd  %clear  %cls  %colors  %conda  %config  %co
nnect_info  %copy  %ddir  %debug  %dhist  %dirs  %doctest_mode  %echo  %ed  %edit  %env  %gui  %hist  %history  %killbgscripts
%ldir  %less  %load  %load_ext  %loadpy  %logoff  %logon  %logstart  %logstate  %logstop  %ls  %lsmagic  %macro  %magic  %matpl
otlib  %mkdir  %more  %notebook  %page  %pastebin  %pdb  %pdef  %pdoc  %pfile  %pinfo  %pinfo2  %pip  %popd  %pprint  %precisio
n  %prun  %psearch  %psource  %pushd  %pwd  %pycat  %pylab  %qtconsole  %quickref  %recall  %rehashx  %reload_ext  %ren  %rep
%rerun  %reset  %reset_selective  %rmdir  %run  %save  %sc  %set_env  %store  %sx  %system  %tb  %time  %timeit  %unalias  %unl
oad_ext  %who  %who_ls  %whos  %xdel  %xmode

Available cell magics:
%%!  %%HTML  %%SVG  %%bash  %%capture  %%cmd  %%debug  %%file  %%html  %%javascript  %%js  %%latex  %%markdown  %%perl  %%prun
%%pypy  %%python  %%python2  %%python3  %%ruby  %%script  %%sh  %%svg  %%sx  %%system  %%time  %%timeit  %%writefile

Automagic is ON, % prefix IS NOT needed for line magics.
```

Рис. 2 Команда %lsmagic

3. Работа с переменными окружения используется команда %env и измерения времени работы кода %%time и %timeit.

```
%env TEST = 1
env: TEST=1
```

```
%%time
import time
for i in range(10):
    time.sleep(0.2)
Wall time: 2.03 s
```

```
%timeit x = [(i**10) for i in range(100)]
38.6 µs ± 1.84 µs per loop (mean ± std. dev. of 7 runs, 10000 loops each)
```

Рис. 3 Результат кода %env и %%time

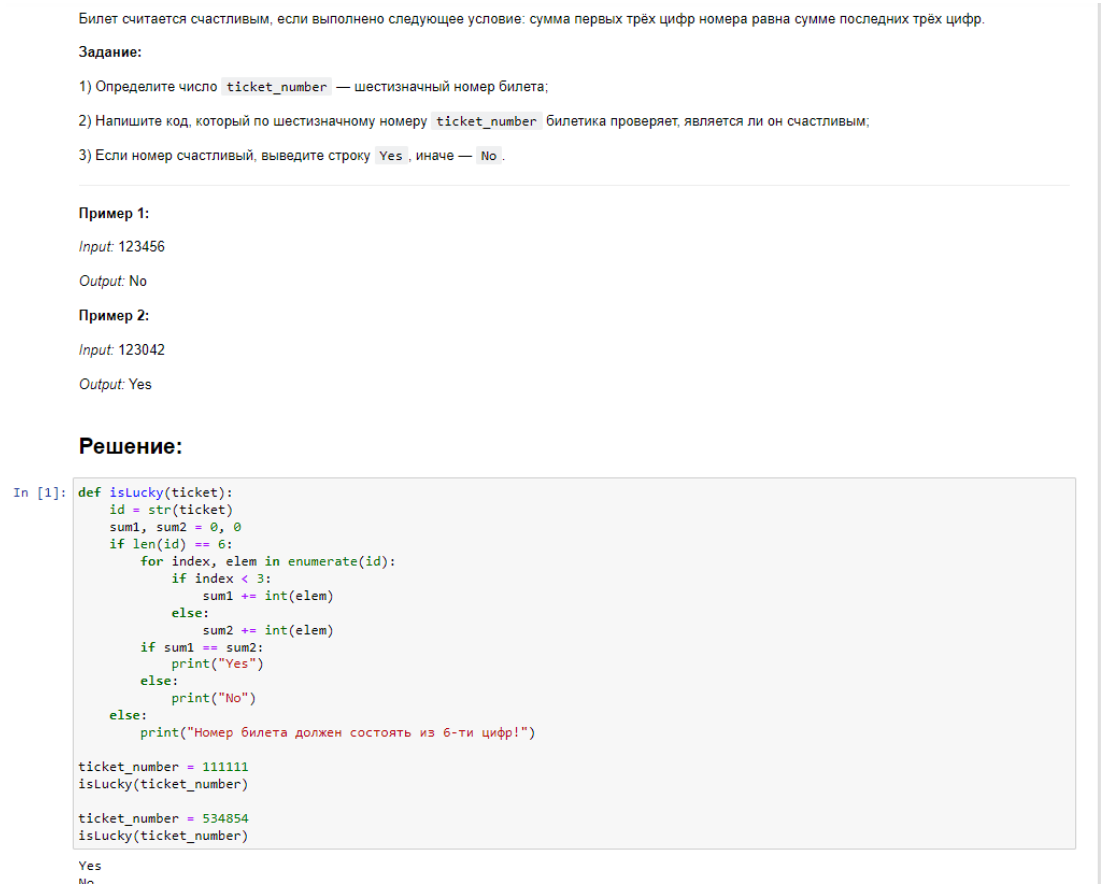
4.1 Билет считается счастливым, если выполнено следующее условие: сумма первых трёх цифр номера равна сумме последних трёх цифр.

Задание:

1) Определите число ticket_number — шестизначный номер билета;

2) Напишите код, который по шестизначному номеру ticket_number билетика проверяет, является ли он счастливым;

3) Если номер счастливый, выведите строку Yes, иначе — No.



Билет считается счастливым, если выполнено следующее условие: сумма первых трёх цифр номера равна сумме последних трёх цифр.

Задание:

1) Определите число `ticket_number` — шестизначный номер билета;

2) Напишите код, который по шестизначному номеру `ticket_number` билетика проверяет, является ли он счастливым;

3) Если номер счастливый, выведите строку `Yes` , иначе — `No` .

Пример 1:

Input: 123456

Output: No

Пример 2:

Input: 123042

Output: Yes

Решение:

In [1]:
```
def isLucky(ticket):
    id = str(ticket)
    sum1, sum2 = 0, 0
    if len(id) == 6:
        for index, elem in enumerate(id):
            if index < 3:
                sum1 += int(elem)
            else:
                sum2 += int(elem)
        if sum1 == sum2:
            print("Yes")
        else:
            print("No")
    else:
        print("Номер билета должен состоять из 6-ти цифр!")

ticket_number = 111111
isLucky(ticket_number)

ticket_number = 534854
isLucky(ticket_number)
```

```
Yes
No
```

Рис. 4 Выполнения задание 1

4.2 Пусть пароль может содержать только латинские буквы, знаки препинания и цифры.

Пароль считается надёжным, если удовлетворяет следующим условиям:

- содержит буквы в разных регистрах;
- содержит цифры;
- содержит не менее 4 уникальных символов;
- не содержит ваше имя латинницей, записанное буквами любых

регистров (anna, iVan, ...).

Иначе пароль считается слабым.

Задание:

1) Определите строку password — придуманный вами пароль;

2) Напишите код, который по паролю password проверяет, является ли он надёжным;

3) Если пароль надёжный, выведите строку strong, иначе — weak.

```
        unique_symb = False
        not_a_name = False
        upp, dwn = 0, 0
        for i in pwd:
            if i.isupper() == True:
                upp += 1
            else:
                dwn += 1
        if upp > 0 and dwn > 0:
            diff_reg = True
        numbers = ['1', '2', '3', '4', '5', '6', '7', '8', '9']
        for i in pwd:
            if i in numbers:
                contain_numbers = True
        pwd_lower = pwd.lower()
        if (len(set(pwd_lower))) >= 4:
            unique_symb = True
        if not(nme.lower() in pwd.lower()):
            not_a_name = True
        if diff_reg == True and contain_numbers == True and unique_symb == True and not_a_name == True:
            print("strong")
        else:
            print("weak")

        print("Символы разных регистров: ", diff_reg)
        print("Содержит цифры: ", contain_numbers)
        print("Не менее 4-х уникальных символов: ", unique_symb)
        print("Не содержит имени: ", not_a_name)
    else:
        print("Пароль не может быть пустым!")

password = "AbCdEfI232"
name = "Andrei"
isStrong(password, name)

password = "andreiiibdEf232"
name = "Andrei"
isStrong(password, name)
```

```
strong
Символы разных регистров:  True
Содержит цифры:  True
Не менее 4-х уникальных символов:  True
Не содержит имени:  True
weak
Символы разных регистров:  True
Содержит цифры:  True
Не менее 4-х уникальных символов:  True
Не содержит имени:  False
```

Рис. 5 Выполнения задание 2

4.3 Определите число amount — количество чисел Фибоначчи, которые надо вывести;

Напишите код, который выводит первые amount чисел Фибоначчи.

Числа Фибоначчи

Как известно, числа Фибоначчи — это последовательность чисел, каждое из которых равно сумме двух предыдущих (первые два числа равны 1):

1, 1, 2, 3, 5, 8, 13, …

Задание:

1) Определите число `amount` — количество чисел Фибоначчи, которые надо вывести;

2) Напишите код, который выводит первые `amount` чисел Фибоначчи.

Пример 1:

Input: 3

Output: 1 1 2

Пример 2:

Input: 10

Output: 1 1 2 3 5 8 13 21 34 55

Решение:

In [3]:
```
def fib(n):
    if n in (1, 2):
        return 1
    return (fib(n - 1) + fib(n - 2))

amount = 10
for i in range(1, amount + 1):
    print(fib(i), end=' ')
```

```
1 1 2 3 5 8 13 21 34 55
```

Рис. 6 Выполнения задание 3

4.4 На сайте https://www.kaggle.com/ выберите любой набор данных в формате CSV и проведите для него маленькое исследование: загрузите данные из набора с использованием стандартного модуля csv, посмотрите средние значения и стандартные отклонения двух выбранных числовых атрибутов, найдите методом наименьших квадратов уравнение линейной зависимости, связывающей один числовой атрибут с другим. Для оценки заданной зависимости найдите коэффициент парной корреляции, сделайте соответствующие выводы.

Импорт данных:

In [4]:
```
import csv
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from math import sqrt

data = []
x = []
y = []

df = pd.read_csv('Twitch_game_data.csv', encoding="cp1252")

with open("Twitch_game_data.csv", encoding='cp1252') as f:
    reader = csv.DictReader(f, dialect='excel')
    for row in reader:
        data.append(row)

for i in data:
        x.append(int(i.get('Peak_viewers')))
        y.append(int(i.get('Peak_channels')))
```

Средние значения показателей:

In [5]:
```
def average(lst):
    return sum(lst) / len(lst)

print("Средние значения: ")
print("X: ", average(x))
print("Y: ", average(y))

print("Проверка: ")
print("X: ", np.mean(x))
print("Y: ", np.mean(y))
```

```
Средние значения:
X:  55095.117847222224
Y:  586.7592361111111
Проверка:
X:  55095.117847222224
Y:  586.7592361111111
```

Рис. 7 Выполнения задание 4

Визуализация:

In [9]:
```
plt.plot(x, y_pred)
plt.scatter(x, y_pred)
plt.scatter(x, y)
plt.ylim(0, 40000)
plt.xlabel('Peak_viewers')
plt.ylabel('Peak_channels')
plt.show()
```

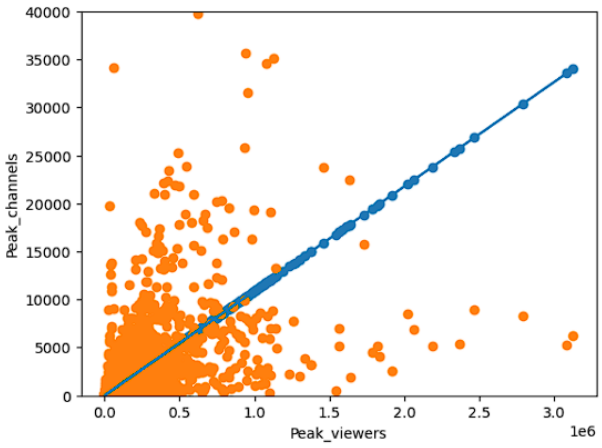


Рис. 8 Выполнения задание 4

5. Выполнить Пример задачи из области физики

In [2]:
```
deltaU = 5/m*(p3*v2 - p1*v1)
print("Внутренняя энергия газа:", deltaU)
```

```
Внутренняя энергия газа: 3250000.0
```

Работа газа - это результат взаимодействия системы с внешними объектами, в результате чего изменяются параметры системы.
Общая работа газа:

$$A = A_{12} + A_{23}$$

Рассмотрим состояния на участках:
Состояние 1-2: постоянное давление, значит работа расширения газа выражается формулой:

$$A_1 = \frac{m}{M} RT = \frac{m}{M} R(T_2 - T_1)$$

Состояние 2-3: постоянный объем, работа газа

$$A_2 = 0$$

Следовательно, работа газа равна:

$$A = A_{12} = p_1(V_2 - V_1)$$

In [3]:
```
A = p1*(v2-v1)
print("Работа газа:", A, "Дж")
```

```
Работа газа: 400000.0 Дж
```

Количество теплоты можно найти из первого закона термодинамики: теплота передаваемая газу равна сумме изменения внутренней энергии:

$$Q = \Delta U + A$$

In [4]:
```
Q = deltaU + A
print("Количество теплоты:", Q, "Дж")
```

```
Количество теплоты: 3650000.0 Дж
```

In [5]:
```
# График процесса pV
import matplotlib.pyplot as plt
p_values = [p1 / 10**5, p2 / 10**5, p3 / 10**5]
v_values = [v1, v2, v3]
plt.grid()
plt.title('График процесса изменения состояния газа')
plt.xlabel('V, м^3.')
plt.ylabel('P, Па.')
plt.plot(v_values, p_values)
plt.scatter(v_values, p_values)
plt.show()
```


Рис. 9 Выполнения задачи

Вывод: В результате выполнения работы были исследованы базовые возможности оболочек IPython и Jupyter Notebook для языка программирования Python, а также были написаны программы в них.

Контрольные вопросы:

1. Как осуществляется запуск Jupyter Notebook?

В командной строке Anaconda набрать команду: jupyter notebook.

2. Какие существуют типы ячеек в Jupyter Notebook?

Существует два вида ячеек: 1) Ячейка кода содержит код, который должен быть выполнен в ядре, и отображает его вывод ниже; 2) Ячейка Markdown содержит текст, отформатированный с использованием Markdown, и отображает его вывод на месте при запуске.

3. Как осуществляется работа с ячейками в Jupyter Notebook?

После выбора ячейки «Code», можно записать код на языке Python, а затем нажать Ctrl+Enter или Shift+Enter. В первом случае введенный код будет выполнен интерпретатором Python, а во втором – будет создана новая ячейка, которая расположится уровнем ниже.

4. Что такое "магические" команды Jupyter Notebook? Какие "магические" команды Вы знаете?

Важной частью функционала Jupyter Notebook является поддержка магии. Под магией понимаются дополнительные команды, выполняемые в рамках оболочки, которые облегчают процесс разработки и расширяют возможности.

Узнать команды: %lsmagic

Для работы с переменными окружениями используется команда %env.

Запуск кода с расширением .ipynb осуществляется с помощью команды %run.

Для измерения времени работы необходимо использовать команды %%time и %timeit.

%matplotlib используется для отображения объектов графиков на экране, ключ после него указывает каким способ отображать график.

5. Самостоятельно изучите работу с Jupyter Notebook и IDE PyCharm и Visual Studio Code. Приведите основные этапы работы с Jupyter Notebook в IDE PyCharm и Visual Studio Code.

PyCharm:

1. Создать новый проект

2. В этом проекте создать новый файл ipynb.

3. Если не установлен пакет Jupyter Notebook, появится сообщение об ошибке: «Пакет Jupyter не установлен», и будет опция «Установить пакет jupyter».

4. «Установить пакет jupyter». Это запустит процесс установки, который вы можете просмотреть, щелкнув запущенные процессы в правом нижнем углу окна PyCharm.

5. Можно создать ячейки кода и выполнить их.

6. Чтобы запустить сервер Jupyter, нужно выполнить ячейку кода. По умолчанию сервер Jupyter использует порт 8888 на локальном хосте. Эти конфигурации доступны в окне инструментов сервера. После запуска вы можете просмотреть сервер над окном исходного кода, а рядом с ним вы можете просмотреть ядро, созданное как «Python 2» или «Python 3».

7. Теперь можно получить доступ к вкладке переменных в PyCharm, чтобы увидеть, как значения переменных меняются при выполнении ячеек кода. Можно также установить точки останова в строках кода, а затем щелкнуть значок «Выполнить» и выбрать «Debug Cell» (или использовать сочетание клавиш Alt+Shift+Enter), чтобы начать отладку.

Visual Studio Code:

1. Чтобы создать новый Jupyter Notebook можно запустить Command Palette (Ctrl+Shift+P) и ввести new notebook. Первым результатом должен быть Jupyter: Create New Blank Jupyter Notebook. Также, его можно создать, нажав на новый файл .ipynb.

2. Блокноты, созданные VS Code, по умолчанию являются доверенными (trusted). Ставить пометку trust нужно вручную по запросу редактора перед выполнением.

3. После создания блокнота нажать save на верхней части панели инструментов, чтобы сохранить его в рабочем пространстве. Теперь можно экспортировать созданный блокнот как скрипт Python или файл HTML/PDF, используя соответствующую иконку.

4. По умолчанию в новом блокноте появится пустая ячейка. Добавьте в нее код и выполните его с помощью Ctrl+Enter. Эта команда запустит выделенную ячейку. Shift+Enter выполняет то же действие, но при этом создает и выделяет новую ячейку ниже, а Alt+Enter выполняет выделенную, создает еще одну ниже, но при этом сохраняет метку на предыдущей.

Иконка + добавляет новую ячейку для кода, а bin удаляет ее. Чтобы перемещать фрагменты вверх и вниз, пользуемся соответствующими стрелками.

Изменить тип ячейки на markdown довольно просто: просто нажмите на иконку M, расположенную над кодом. Чтобы снова установить значение code, выберите значок {}. Выполнить эти действия также можно с помощью клавиш M и Y.

Также, расширение Jupyter для VS Code поддерживает построчное выполнение кода в ячейке, нажав на кнопку, расположенную рядом с иконкой Play. Вторая возможность для отладки – можно просто экспортировать блокнот как скрипт Python и работать с ним прямо в отладчике VS Code, не переходя в другую среду.