

Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт цифрового развития
Кафедра инфокоммуникаций

«Работа с IPython и Jupyter Notebook»

ОТЧЕТ
по лабораторной работе №1
дисциплины
«Технологии распознавания образов»

Выполнила:

Кувшин Ирина Анатольевна
2 курс, группа ПИЖ-б-о-21-1,
011.03.04 «Программная инженерия»,
направленность (профиль) «Разработка
и сопровождение программного
обеспечения», очная форма обучения

(подпись)

Проверил:

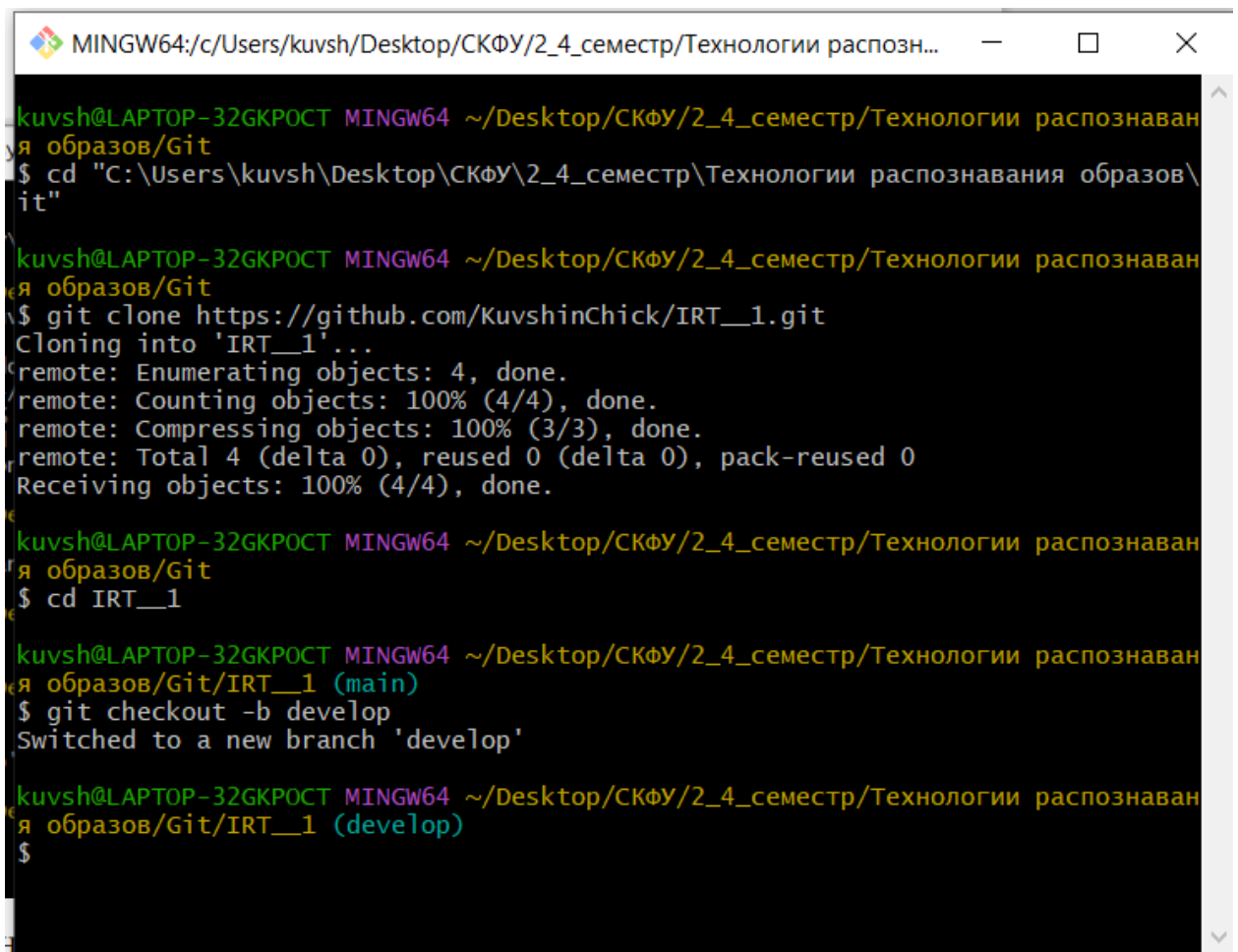
(подпись)

Отчет защищен с оценкой _____ Дата защиты _____

Ставрополь, 2022 г.

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

1. Изучить теоретический материал работы.
2. Создать общедоступный репозиторий на GitHub, в котором будет использована лицензия MIT и выбранный Вами язык программирования (выбор языка программирования будет доступен после установки флажка Add .gitignore).
3. Выполните клонирование созданного репозитория на рабочий компьютер.
4. Организуйте свой репозиторий в соответствии с моделью ветвления git-flow.



```
MINGW64:/c:/Users/kuvsh/Desktop/СКФУ/2_4_семестр/Технологии распознаван...
kuvsh@LAPTOP-32GKPOCT MINGW64 ~/Desktop/СКФУ/2_4_семестр/Технологии распознаван
я образов/Git
$ cd "C:\Users\kuvsh\Desktop\СКФУ\2_4_семестр\Технологии распознавания образов\
it"

kuvsh@LAPTOP-32GKPOCT MINGW64 ~/Desktop/СКФУ/2_4_семестр/Технологии распознаван
я образов/Git
$ git clone https://github.com/KuvshinChick/IRT__1.git
Cloning into 'IRT__1'...
remote: Enumerating objects: 4, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 4 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (4/4), done.

kuvsh@LAPTOP-32GKPOCT MINGW64 ~/Desktop/СКФУ/2_4_семестр/Технологии распознаван
я образов/Git
$ cd IRT__1

kuvsh@LAPTOP-32GKPOCT MINGW64 ~/Desktop/СКФУ/2_4_семестр/Технологии распознаван
я образов/Git/IRT__1 (main)
$ git checkout -b develop
Switched to a new branch 'develop'

kuvsh@LAPTOP-32GKPOCT MINGW64 ~/Desktop/СКФУ/2_4_семестр/Технологии распознаван
я образов/Git/IRT__1 (develop)
$
```

Рисунок 1.1 – Клонирование репозитория и создание ветки develop

```
MINGW64:/c:/Users/kuvsh/Desktop/СКФУ/2_4_семестр/Технологии распозн...
kuvsh@LAPTOP-32GKPOCT MINGW64 ~/Desktop/СКФУ/2_4_семестр/Технологии распозн...
ия образов/Git/IRT__1 (develop)
$ git commit -"modified .gitignore & readme"
[develop e029be3] odified .gitignore & readme
2 files changed, 132 insertions(+), 1 deletion(-)
create mode 100644 .gitignore

kuvsh@LAPTOP-32GKPOCT MINGW64 ~/Desktop/СКФУ/2_4_семестр/Технологии распозн...
ия образов/Git/IRT__1 (develop)
$ git push origin develop
Enumerating objects: 6, done.
Counting objects: 100% (6/6), done.
Delta compression using up to 8 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 1.40 KiB | 1.40 MiB/s, done.
Total 4 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/KuvshinChick/IRT__1.git
141c798..e029be3 develop -> develop

kuvsh@LAPTOP-32GKPOCT MINGW64 ~/Desktop/СКФУ/2_4_семестр/Технологии распозн...
ия образов/Git/IRT__1 (develop)
$ |
```

Рисунок 1.2 – Обновление .gitignore и readme

6. Проработать примеры лабораторной работы.

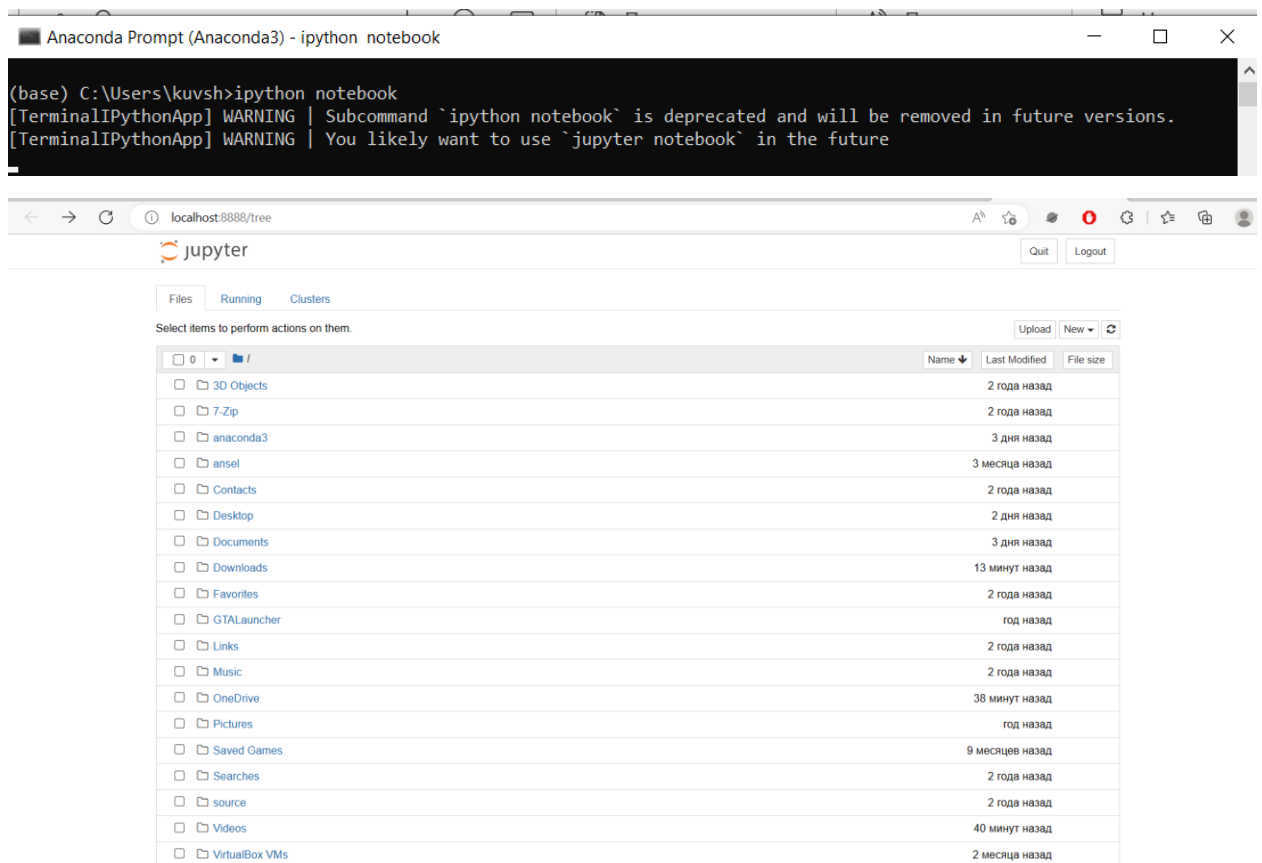


Рисунок 1.3 – Результат запуска оболочки в браузере

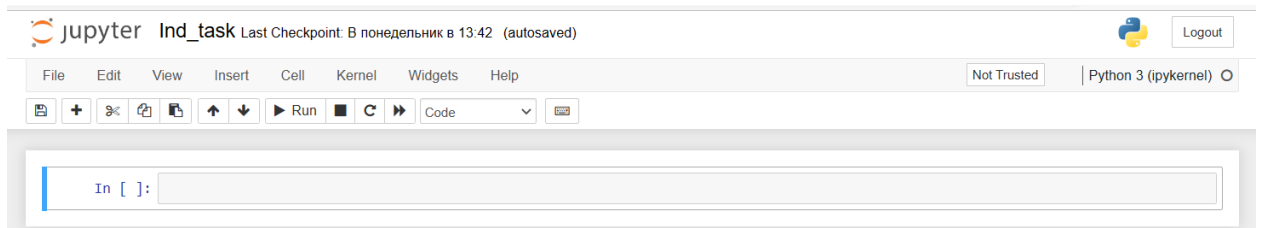


Рисунок 1.4 – Результат запуска ноутбука

```
In [1]: 3+2
Out[1]: 5
```

Рисунок 1.5 – Результат проработки примеров

```
In [3]: a = 5
        b = 7
        print(a + b)
12

In [4]: n = 7
        for i in range(n):
            print(i*10)
0
10
20
30
40
50
60

In [6]: i = 0
        while True:
            i += 1
            if i > 5:
                break
            print("Test while")
Test while
Test while
Test while
Test while
Test while
```

Рисунок 1.6 – Результат проработки примеров

```
In [1]: from matplotlib import pylab as plt
        %matplotlib inline
```

```
In [2]: x = [i for i in range(50)]
        y = [i**2 for i in range(50)]
        plt.plot(x,y)
```

```
Out[2]: <matplotlib.lines.Line2D at 0x25347191e20>
```

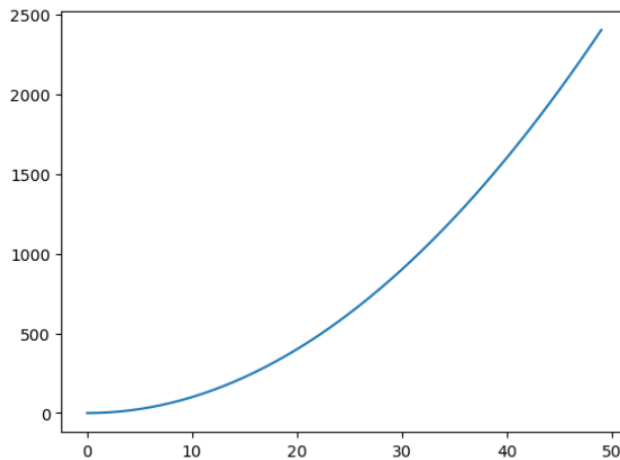


Рисунок 1.7 – Результат проработки примеров

```
In [1]: %lsmagic
```

```
Out[1]: 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.
```

Рисунок 1.8 – Список доступных магических команд

```
In [3]: %env TEST=5
```

```
env: TEST=5
```

Рисунок 1.9 – Работа с переменными окружения

```
In [7]: %%time
import time
for i in range(50):
    time.sleep(0.1)
```

```
CPU times: total: 0 ns
Wall time: 5.39 s
```

Рисунок 1.10 – Результат проработки примеров

```
In [8]: %timeit x = [(i**10) for i in range(10)]
```

3.17 μ s \pm 65.7 ns per loop (mean \pm std. dev. of 7 runs, 100,000 loops each)

Рисунок 1.11 – Результат проработки примеров

7. Решить задания в ноутбуках, выполненных преподавателем.

Счастливый билетик



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` билета 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

```
In [13]: ticket_number = 277011
```

```
In [14]: num1 = ticket_number//1000
num2 = ticket_number%1000
if sum(map(int,str(num1))) == sum(map(int,str(num2))):
    print("Yes")
else:
    print("No")
```

No

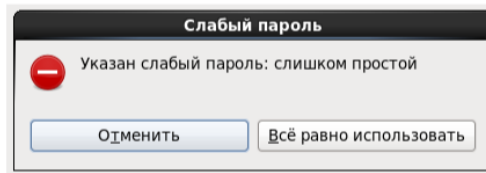
```
In [2]: ticket_number = 120300
```

```
In [6]: num1 = ticket_number//1000
num2 = ticket_number%1000
if sum(map(int,str(num1))) == sum(map(int,str(num2))):
    print("Yes")
else:
    print("No")
```

Yes

Рисунок 1.12 – Решение и результат задачи

Пароль



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 латиницей, 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`.

Пусть имя пользователя — Андрей.

Пример 1:

Input: Aandrei123

Output: weak

Пример 2:

Input: an12dRei

Output: strong

```
In [17]: name = input("Enter your name:")
password = input("Enter your password:")
```

Enter your name:Irina
Enter your password:Ir75Feb72Ra

```
In [18]: if not(password == password.lower() or password.upper() == password or password.
          or len(set(password)) < 4 or name.lower() in password.lower()):
          print("strong")
        else:
          print("weak")
```

strong

```
In [19]: name = input("Enter your name:")
password = input("Enter your password:")
```

Enter your name:Mick
Enter your password:_MiCk86G

```
In [20]: if not(password == password.lower() or password.upper() == password or password.
          or len(set(password)) < 4 or name.lower() in password.lower()):
          print("strong")
        else:
          print("weak")
```

weak

Рисунок 1.13 – Решение и результат задачи

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

```
In [52]: amount = int(input("Enter a number: "))
```

Enter a number: 11

```
In [53]: num = 0
temp_sum = 1
for i in range(amount):
    sum = num + temp_sum
    num = temp_sum
    print (temp_sum, end=" ")
    temp_sum = sum
```

1 1 2 3 5 8 13 21 34 55 89

Рисунок 1.14 – Решение и результат задачи

Время исследований

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.

Результаты надо обязательно прокомментировать и пояснить!

Пример 1:

Пусть таблица `bikes.csv` содержит данные по арендам велосипедов за 2 года:

- `datetime` : дата и время аренды
- `season` : время года
- `temp` : температура воздуха по Цельсию
- `windspeed` : скорость ветра
- `registered` : число аренд

Одно из направлений исследования могло бы заключаться в проверке зависимости суммарного числа аренд от температуры воздуха.

[Video Game Sales | Kaggle](#)

Этот набор данных(vgsales.csv) содержит список видеоигр, продажи которых превысили 100 000 копий. Направление исследования заключается в проверке зависимости года выпуска игры от продаж в Европе(в миллионах). После открытия файла формируются списки:

1. Жанры игр
2. Число продаж

```
In [33]: import csv
import statistics
import math
from matplotlib import pylab as plt
%matplotlib inline

# Открытие файла только для чтения
with open('vgsales.csv', 'r', newline='', encoding='utf-8') as csvfile:
    reader = csv.reader(csvfile, delimiter=',')
    # Списки
    Year = []
    EU_Sales = []
    # Пропуск заголовка
    next(reader)
    # Заполнение списков
    for row in reader:
        if row[3] != "N/A":
            Year.append(float(row[3]))
            EU_Sales.append(float(row[7]))

# Подсчет средних значений
mean_Year = sum(Year) / len(Year)
mean_EU_Sales = sum(EU_Sales) / len(EU_Sales)
print(f"Среднее значение года выпуска игры: {mean_Year}")
print(f"Среднее значение продаж в Европе в миллионах: {mean_EU_Sales}")

# Стандартные отклонения двух выбранных числовых атрибутов
Year_std = statistics.stdev(Year)
EU_Sales_std = statistics.stdev(EU_Sales)
print("")
print(f"Стандартные отклонения года выпуска игры: {Year_std}")
print(f"Стандартные отклонения значений продаж в Европе в миллионах: {EU_Sales_std}")

Среднее значение года выпуска игры: 2006.4064433147546
Среднее значение продаж в Европе в миллионах: 0.14755435781224593

Стандартные отклонения года выпуска игры: 5.82898114712805
Стандартные отклонения значений продаж в Европе в миллионах: 0.5087656986961251
```

<https://ai-shell.ru/articles/zadachi-regressionnogo-analiza-metod-naimenshih-kvadratov-uravnenie-lineynoy-regressii/> - описание

```
In [43]: # Сумма значений года выпуска
sum_Year = sum(Year)
# Сумма значений продаж
sum_EU_Sales = sum(EU_Sales)

# Сумма произведений элементов и квадрат значений года выпуска
sum_mult = 0
sum_square_year = 0

for i, elem in enumerate(Year):
    sum_mult += elem * EU_Sales[i]
    sum_square_year += elem**2

n = len(Year)
# Нахождение коэффициентов по формулам
k = (n * sum_mult - sum_Year * sum_EU_Sales) / (n * sum_square_year - sum_Year**2)
b = (sum_EU_Sales - k * sum_Year) / n

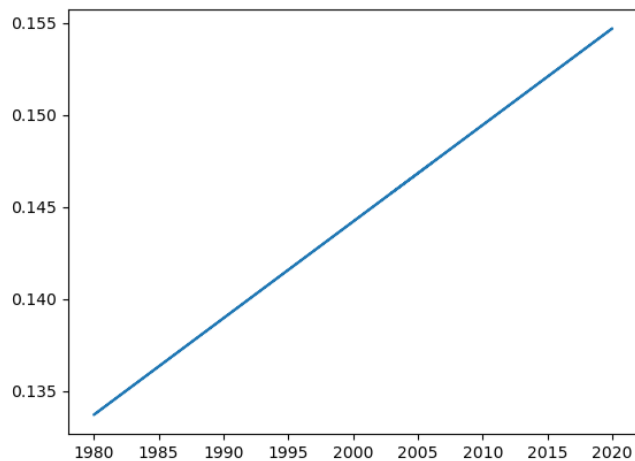
# Значения функции
func_val = []
for elem in Year:
    func_val.append(k * elem + b)
print(f"Уравнение линейной зависимости: y = {k}x + {b}")
print("График функции методом наименьших квадратов: ")

# График
plt.plot(Year, func_val)

Уравнение линейной зависимости: y = 0.0005249047035943598x + -0.905617821605699
График функции методом наименьших квадратов:
```

Уравнение линейной зависимости: $y = 0.0005249047035943598x + -0.905617821605699$
График функции методом наименьших квадратов:

Out[43]: [matplotlib.lines.Line2D at 0x2cc01692520]



http://www.rniz.ru/econometrica/raschet_koefficienta_korrelyatsii.php - описание

```
In [42]: # Сумма значений года выпуска
sum_Year = sum(Year)
# Сумма значений продаж
sum_EU_Sales = sum(EU_Sales)

# Сумма произведений элементов, квадрат значений года выпуска
# и квадрат значений продаж в Европе в миллионах
sum_mult = 0
sum_square_year = 0
sum_square_EU_Sales = 0

for i, elem in enumerate(Year):
    sum_mult += elem * EU_Sales[i]
    sum_square_year += elem**2
    sum_square_EU_Sales += EU_Sales[i]**2

n = len(Year)
# Нахождение коэффициента корреляции по формуле
r = (sum_mult - sum_Year * sum_EU_Sales/n)/(sqrt((sum_square_year-sum_Year**2/n)*(sum_square_EU_Sales-sum_EU_Sales**2/n)))
print(f"Коэффициент парной корреляции: {r}")
```

Коэффициент парной корреляции: 9.251189352935169e-05

Значение -0.0000925 можно рассматривать как почти полное отсутствие корреляции.

Рисунок 1.15 – Решение и результат задачи

8. Создать ноутбук, в котором выполнить решение вычислительной задачи (например, задачи из области физики, экономики, математики, статистики и т. д.), условие которой предварительно необходимо согласовать с преподавателем.

При проведении лабораторной работы по вычислению ускорения свободного падения математический маятник длиной $l = 1\text{ м}$ совершил за время $t = 34\text{ с}$ 17 колебаний. Какое значение ускорения свободного падения получено из данных опыта?

$\pi = 3,14$

```
In [3]: import math
# Дано
l = 1
t = 34
Pi = 3.14
N = 17
# Решение
# Период колебаний математического маятника
#  $T = 2\pi\sqrt{l/g}$ 
#  $T^2 = 4\pi^2 l/g$ 
#  $T^2 g = 4\pi^2 l$ 
#  $g = 4\pi^2 l / T^2$ 
T = t/N
g = 4*Pi**2*l/(t/N)**2
print(f"Ускорение свободного падения: {g}")
```

Ускорение свободного падения: 9.8596

Рисунок 1.16 – Решение и результат задачи

9. Зафиксируйте сделанные изменения в репозитории.
10. Выполните слияние ветки для разработки с веткой main (master).
11. Отправьте сделанные изменения на сервер GitHub.

Контрольные вопросы

1. Как осуществляется запуск Jupyter notebook?

Jupyter Notebook входит в состав Anaconda. Для запуска Jupyter Notebook перейдите в папку Scripts (она находится внутри каталога, в котором установлена Anaconda) и в командной строке наберите:

```
>ipython notebook / jupyter notebook
```

2. Какие существуют типы ячеек в Jupyter notebook?

Markdown/Code

3. Как осуществляется работа с ячейками в Jupyter notebook?

Ячейки в jupyter notebook можно создавать удалять и запускать их работу при помощи комбинаций клавиш

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

Важной частью функционала Jupyter Notebook является поддержка магии. Под магией в IPython понимаются дополнительные команды, выполняемые в рамках оболочки, которые облегчают процесс разработки и расширяют ваши возможности. Список доступных магических команд можно получить с помощью команды `%lsmagic`

```
In [1]: %lsmagic
Out[1]: Available line magics:
%alias %alias_magic %autocall %automagic %autosave %bookmark %cd %clear %cls %colors %config %connect_info %copy %ddir %debug %dhist %dirs %doctest_mode %echo %ed %edit %env %gui %hist %history %killbgscripts %ldir %les %load %load_ext %loadpy %logoff %logon %logstart %logstate %logstop %ls %lsmagic %macro %magic %matplotlib %mkdir %more %notebook %page %pastebin %pdb %pdef %pdoc %pfile %pinfo %pinfo2 %popd %pprint %precision %profile %prun %pssearch %psource %pushd %pwd %pycat %pylab %qtconsole %quickref %recall %rehashx %reload_ext %rename %rep %rerun %reset %reset_selective %rmdir %run %save %sc %set_env %store %sx %system %tb %time %timeit %unalias %unload_ext %who %who_ls %whos %xdel %xmode

Available cell magics:
%%! %%HTML %%SVG %%bash %%capture %%cmd %%debug %%file %%html %%javascript %%js %%latex %%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.
```

5. Самостоятельно изучите работу с Jupyter notebook и IDE PyCharm и Visual Studio Code.

PyCharm и работа с Jupyter Notebook (ip-calculator.ru)

Working with Jupyter Notebooks in Visual Studio Code

6. Приведите основные этапы работы с Jupyter notebook в IDE PyCharm и Visual Studio Code.

Расширение записной книжки Jupyter в коде Visual Studio -
GeeksforGeeks (turbopages.org)

How to create Jupyter Notebook in PyCharm - Softhints