



МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования

«МИРЭА – Российский технологический университет»

РТУ МИРЭА

Институт Информационных технологий

Кафедра прикладной математики

Отчет по практической работе №1

по дисциплине

«Технологии и инструментарий анализа больших данных»

Выполнил:

Студент группы ИКБО-26-22

Скорик Никита Максимович

Проверил:

преподаватель Юрченков И.А.

МОСКВА 2025 г.

Оглавление

Введение.....	3
1 Решение задач.....	4
1.1 Задача №2.....	4
1.2 Задача №3.....	4
1.3 Задача №4.....	5
1.4 Задача №5.....	5
1.5 Задача №6.....	6
1.6 Задачи №7-12	6
1.7 Задача №*1	8
1.8 Задача №*2	9
1.9 Задача №*3	10
Заключение	11

Введение

В современном мире данных, где объем информации стремительно увеличивается и становится все более сложным, способность эффективно анализировать и извлекать полезные знания из больших данных приобретает особую значимость. В этой связи язык программирования Python стал одним из наиболее популярных и мощных инструментов в области анализа данных. Его простота в использовании, широкий набор библиотек и поддержка различных технологий делают Python незаменимым в процессе работы с большими данными.

Python — это высокоуровневый язык программирования, известный своей читаемостью и гибкостью. Он был разработан с целью упрощения программирования и повышения продуктивности разработчиков. В контексте анализа больших данных Python предлагает множество библиотек и инструментов, которые облегчают выполнение сложных задач, таких как обработка и визуализация данных, статистический анализ и машинное обучение.

Одной из ключевых причин популярности Python в области анализа данных является его богатая экосистема. Библиотеки, такие как NumPy, pandas, Matplotlib и SciPy, предоставляют мощные инструменты для обработки данных, их анализа и визуализации. В дополнение к этому, Python интегрируется с различными платформами и фреймворками для работы с большими данными, такими как Apache Spark и Dask, что расширяет его возможности и позволяет работать с распределенными вычислениями.

Цель данной работы — рассмотреть язык программирования Python как инструмент для анализа больших данных. В работе будут исследованы ключевые библиотеки и инструменты, которые Python предоставляет для работы с большими объемами информации, а также особенности и преимущества его использования в контексте технологий и инструментов анализа данных.

1 Решение задач

1.1 Задача №2

Решение и результат программы представлены на Рисунке 1.1.

```
import numpy as np
a, b, h, r = 1, 1, 1, 1
def triangle(a, h):
    print("Введите основание и высоту треугольника")
    a = int(input())
    h = int(input())
    return (a * h / 2)

def rectangle(a, b):
    print("Введите стороны прямоугольника")
    a = int(input())
    b = int(input())
    return (a * b)

def circle(r):
    print("Введите радиус круга")
    r = int(input())
    return (r ** 2 * np.pi)

dict = {"Треугольник" : triangle(a, h), "Прямоугольник" : rectangle(a, b), "Круг" : circle(r)}
print(dict)
```

Введите основание и высоту треугольника
4
6
Введите стороны прямоугольника
2
3
Введите радиус круга
5
{'Треугольник': 12.0, 'Прямоугольник': 6, 'Круг': 78.53981633974483}

Рисунок 1.1 – Результат выполнения программы

1.2 Задача №3

Решение и результат программы представлены на Рисунке 1.2.

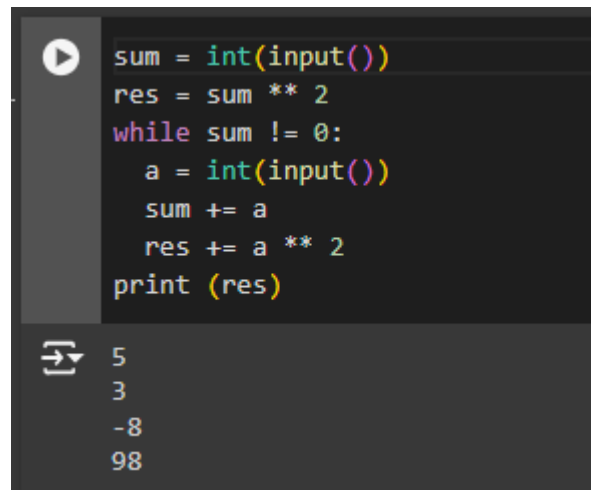
```
print(eval(input("Введите первое число : ") + input("Введите действие : ") + input("Введите второе число : ") ))
```

Введите первое число : 3
Введите действие : **
Введите второе число : 5
243

Рисунок 1.2 – Результат выполнения программы

1.3 Задача №4

Решение и результат программы представлены на Рисунке 1.3.



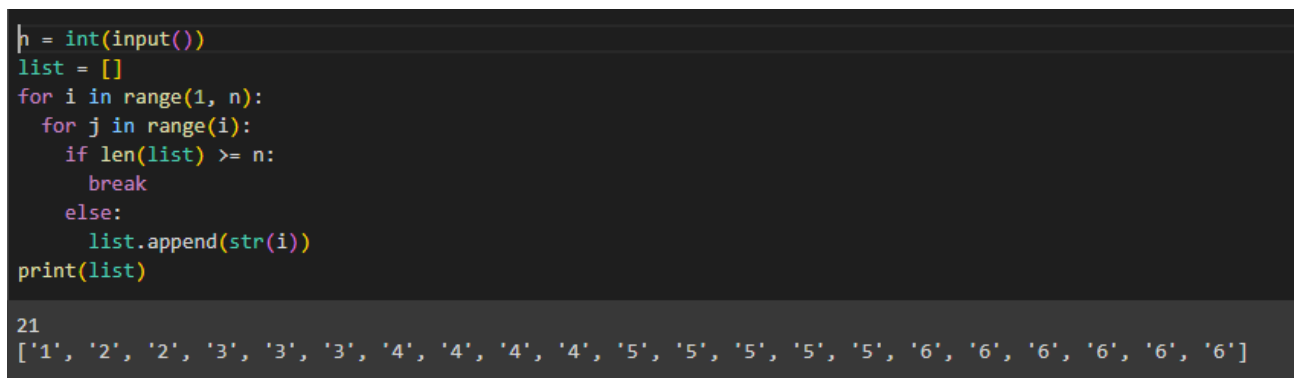
```
sum = int(input())
res = sum ** 2
while sum != 0:
    a = int(input())
    sum += a
    res += a ** 2
print (res)
```

5
3
-8
98

Рисунок 1.3 – Результат выполнения программы

1.4 Задача №5

Решение и результат программы представлены на Рисунке 1.4.



```
h = int(input())
list = []
for i in range(1, h):
    for j in range(i):
        if len(list) >= h:
            break
        else:
            list.append(str(i))
print(list)
```

21
['1', '2', '2', '3', '3', '3', '4', '4', '4', '4', '5', '5', '5', '5', '5', '6', '6', '6', '6', '6', '6']

Рисунок 1.4 – Результат выполнения программы

1.5 Задача №6

Решение и результат программы представлены на Рисунке 1.5.

```
A = [1, 2, 3, 4, 2, 1, 3, 4, 5, 6, 5, 4, 3, 2]
B = ['a', 'b', 'c', 'c', 'c', 'b', 'a', 'c', 'a', 'a', 'b', 'c', 'b', 'a']
suma, sumb, sumc = 0, 0, 0

result = {}

for i in range(len(A)):
    key = B[i]
    if key in result:
        result[key] += A[i]
    else:
        result[key] = A[i]

print(result)

{'a': 17, 'b': 11, 'c': 17}
```

Рисунок 1.5 – Результат выполнения программы

1.6 Задачи №7-12

Решение представлено на Рисунке 1.6.

```
import pandas as pd
from sklearn.datasets import fetch_california_housing

#7
"""Скачать и загрузить данные о стоимости домов в калифорнии, используя библиотеку sklearn."""
california_housing = fetch_california_housing()
df = pd.DataFrame(data=california_housing.data, columns=california_housing.feature_names)
df['MedHouseVal'] = california_housing.target

#8
"""Использовать метод info()."""
print(df.info())

#9
"""Узнать, есть ли пропущенные значения, используя isna().sum()."""
if df['MedHouseVal'].isna().sum() == True:
    print('Some values are missing')
else:
    print('All values are present')

#10
"""Вывести записи, где средний возраст домов в районе более 50 лет и население более 2500 человек, используя метод loc()."""
print(df.loc[(df['HouseAge'] > 50) & (df['Population'] > 2500)])

#11
"""Узнать максимальное и минимальное значения медианной стоимости дома."""
print(df['MedHouseVal'].max())
print(df['MedHouseVal'].min())

#12
"""Используя метод apply(), вывести на экран название признака и его среднее значение."""
mean_values = df.apply(lambda x: x.mean())
for feature, mean in mean_values.items():
    print(f"Признак: {feature}, Среднее значение: {mean}")
```

Рисунок 1.6 – Решение блока задач

```

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20640 entries, 0 to 20639
Data columns (total 9 columns):
#   Column          Non-Null Count  Dtype
---  -
0   MedInc          20640 non-null  float64
1   HouseAge        20640 non-null  float64
2   AveRooms        20640 non-null  float64
3   AveBedrms       20640 non-null  float64
4   Population      20640 non-null  float64
5   AveOccup        20640 non-null  float64
6   Latitude        20640 non-null  float64
7   Longitude       20640 non-null  float64
8   MedHouseVal     20640 non-null  float64
dtypes: float64(9)
memory usage: 1.4 MB
None
All values are present

```

	MedInc	HouseAge	AveRooms	AveBedrms	Population	AveOccup
460	1.4012	52.0	3.105714	1.060000	3337.0	9.534286
4131	3.5349	52.0	4.646119	1.047945	2589.0	5.910959
4440	2.6806	52.0	4.806283	1.057592	3062.0	4.007853
5986	1.8750	52.0	4.500000	1.206349	2688.0	21.333333
7369	3.1901	52.0	4.730942	1.017937	3731.0	4.182735
8227	2.3305	52.0	3.488860	1.170380	3018.0	3.955439
13034	6.1359	52.0	8.275862	1.517241	6675.0	230.172414
15634	1.8295	52.0	2.628169	1.053521	2957.0	4.164789
15652	0.9000	52.0	2.237474	1.053535	3260.0	2.237474
15657	2.5166	52.0	2.839075	1.184049	3436.0	1.621520
15659	1.7240	52.0	2.278566	1.082348	4518.0	1.780142
15795	2.5755	52.0	3.402576	1.058776	2619.0	2.108696
15868	2.8135	52.0	4.584329	1.041169	2987.0	3.966799

	Latitude	Longitude	MedHouseVal
460	37.87	-122.26	1.75000
4131	34.13	-118.20	1.93600
4440	34.08	-118.21	1.53000
5986	34.10	-117.71	2.12500
7369	33.97	-118.21	1.67600
8227	33.78	-118.20	1.62500
13034	38.69	-121.15	2.25000
15634	37.80	-122.41	2.43800
15652	37.80	-122.41	5.00001
15657	37.79	-122.41	2.75000
15659	37.79	-122.41	2.25000
15795	37.77	-122.42	3.25000
15868	37.76	-122.41	2.60300

```

5.00001
0.14999
Признак: MedInc, Среднее значение: 3.8706710029069766
Признак: HouseAge, Среднее значение: 28.639486434108527
Признак: AveRooms, Среднее значение: 5.428999742190376
Признак: AveBedrms, Среднее значение: 1.096675149606208
Признак: Population, Среднее значение: 1425.4767441860465
Признак: AveOccup, Среднее значение: 3.0706551594363742
Признак: Latitude, Среднее значение: 35.63186143410853
Признак: Longitude, Среднее значение: -119.56970445736432
Признак: MedHouseVal, Среднее значение: 2.068558169089147

```

Рисунок 1.7 – Результат выполнения блока программ

1.7 Задача №*1

Решение и результат программы представлены на Рисунке 1.7.

```
morze = {'a': '.-', 'b': '-...', 'c': '-.-.', 'd': '-..',  
        'e': '.', 'f': '..-.', 'g': '--.', 'h': '....',  
        'i': '..', 'j': '.---', 'k': '-.-', 'l': '.-..',  
        'm': '--', 'n': '-.', 'o': '---', 'p': '.---.',  
        'q': '-.-.-', 'r': '.-.-', 's': '...', 't': '-',  
        'u': '..-', 'v': '...-', 'w': '.-.-', 'x': '-.-.-',  
        'y': '-.-.-', 'z': '--..'}  
  
string = str(input())  
string = string.lower()  
words = string.split()  
for j in words:  
    res = ''  
    for i in j:  
        res += morze[i] + ' '  
    print(res)  
  
Hello World  
... . .-.. -.- -..  
.-- --- -. -.- -..
```

Рисунок 1.7 – Результат выполнения программы

1.8 Задача №2

Решение и результат программы представлены на Рисунке 1.8.

```
def register_users(n, requests):
    registered = set()
    results = []

    for name in requests:
        if name not in registered:
            registered.add(name)
            results.append("OK")
        else:
            i = 1
            new_name = f"{name}{i}"
            while new_name in registered:
                i += 1
                new_name = f"{name}{i}"
            registered.add(new_name)
            results.append(new_name)

    return results

n = int(input())
requests = [input().strip() for _ in range(n)]
responses = register_users(n, requests)

for response in responses:
    print(response)
```

5
asd
asd
zxc
zxc
asd
OK
asd1
OK
zxc1
asd2

Рисунок 1.8 – Результат выполнения программы

1.9 Задача №3

Решение и результат программы представлены на Рисунке 1.9.

```
def process_file_requests(n, file_info, m, requests):
    file_permissions = {}

    for info in file_info:
        parts = info.split()
        filename = parts[0]
        actions = set(parts[1:])
        file_permissions[filename] = actions

    results = []
    for request in requests:
        operation, filename = request.split()
        operation_map = {
            "write": "w",
            "read": "r",
            "execute": "x"
        }

        if filename in file_permissions and operation_map[operation] in file_permissions[filename]:
            results.append("OK")
        else:
            results.append("Access denied")

    return results

n = int(input("Введите количество файлов: "))
file_info = [input().strip() for _ in range(n)]
m = int(input("Введите количество запросов: "))
requests = [input().strip() for _ in range(m)]

responses = process_file_requests(n, file_info, m, requests)

for response in responses:
    print(response)
```

Введите количество файлов: 2
list.txt r w
virus.exe x
Введите количество запросов: 6
read list.txt
write list.txt
execute list.txt
read virus.exe
write virus.exe
execute virus.exe
OK
OK
Access denied
Access denied
Access denied
OK

Рисунок 1.9 – Результат выполнения программы

Заключение

В ходе выполнения практической работы были изучены основы языка программирования Python, его философия (The Zen of Python), синтаксис, основные типы данных, структуры данных (списки, кортежи, множества, словари), условные операторы, циклы, функции, а также работа с популярными библиотеками для анализа данных: numpy, pandas и scikit-learn.