



МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«МИРЭА – Российский технологический университет»
РТУ МИРЭА


Институт информационных технологий (ИИТ)
Кафедра корпоративных информационных систем (КИС)

ОТЧЁТ ПО УЧЕБНОЙ ПРАКТИКЕ
Ознакомительная практика

приказ Университета о направлении на практику от 09 февраля 2022 г. № 1103-С

Отчет представлен к
рассмотрению:
Студент группы ИКБО-08-21

«08» июня 2022

 Властюк А.В.
(подпись и расшифровка подписи)

Отчет утвержден.
Допущен к защите:

Руководитель практики
от кафедры

«08» июня 2022

 Габриелян Г.А.
(подпись и расшифровка подписи)

Москва 2022 г.



МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«МИРЭА – Российский технологический университет»
РТУ МИРЭА

Институт информационных технологий (ИИТ)
Кафедра корпоративных информационных систем (КИС)

ИНДИВИДУАЛЬНОЕ ЗАДАНИЕ НА УЧЕБНУЮ ПРАКТИКУ
Ознакомительная практика

Студенту 1 курса учебной группы ИКБО-08-21

Властюку Артёму Валерьевичу

Место и время практики: РТУ МИРЭА кафедра КИС, с 09 февраля 2022 г. по 31 мая 2022 г.

Должность на практике: студент

1. СОДЕРЖАНИЕ ПРАКТИКИ:

1.1. Изучить: методические материалы по курсу, программную документацию языка программирования и стандартных библиотек, API применяемых сервисов.

1.2. Практически выполнить:

- анализ выданных в курсе практических задач;
- поиск, интерпретацию, анализ и ранжирование информации из изученных источников и баз данных, необходимых для решения практических задач;
- решение задач по темам: работа с коллекциями, работа со строками, работа с файлами, основы ООП, стандартные библиотеки, графический интерфейс, использование сторонних API;
- представление результатов выполнения практических задач в требуемом формате (условие, алгоритм, решение задачи, тестирование).

1.3. Ознакомиться: со средствами социального взаимодействия и командной работы в профессиональной среде, с учебно-методическим пособием по ознакомительной практике.

2. ДОПОЛНИТЕЛЬНОЕ ЗАДАНИЕ: нет

3. ОРГАНИЗАЦИОННО-МЕТОДИЧЕСКИЕ УКАЗАНИЯ: Положение о практической подготовке обучающихся, осваивающих основные профессиональные образовательные программы ВО; учебно-методическое пособие по ознакомительной практике.

Руководитель практики от
кафедры

Подпись

(Габриелян Г.А.)

«09» февраля 2022 г.

Задание получил

Подпись

(Властюк А.В.)

«09» февраля 2022 г.

СОГЛАСОВАНО:

Заведующий кафедрой:

Подпись

(Андрианова Е.Г.)

«09» февраля 2022 г.

Проведенные инструктажи:

Охрана труда:

Инструктирующий

Инструктируемый


Подпись


Подпись

«09» февраля 2022 г.

Трохаченкова Н.Н., старший
преподаватель кафедры КИС

Властюк А.В.

Техника безопасности:

Инструктирующий

Инструктируемый


Подпись


Подпись

«09» февраля 2022 г.

Трохаченкова Н.Н., старший
преподаватель кафедры КИС

Властюк А.В.

Пожарная безопасность:

Инструктирующий

Инструктируемый


Подпись


Подпись

«09» февраля 2022 г.

Трохаченкова Н.Н., старший
преподаватель кафедры КИС

Властюк А.В.

С правилами внутреннего распорядка ознакомлен:


Подпись

«09» февраля 2022 г.

Властюк А.В.



МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«МИРЭА – Российский технологический университет»
РТУ МИРЭА

**РАБОЧИЙ ГРАФИК ПРОВЕДЕНИЯ
ОЗНАКОМИТЕЛЬНОЙ ПРАКТИКИ**

студента Властюк А.В. 1 курса группы ИКБО-08-21 очной формы обучения, обучающегося по направлению подготовки 09.03.04 Программная инженерия

Неделя	Сроки выполнения	Этап	Отметка о выполнении
1	09.02 – 15.02	Подготовительный этап, включающий в себя организационное собрание (Вводная лекция о порядке организации и прохождения производственной практики, инструктаж по технике безопасности, получение задания на практику)	
2-3	17.02 – 01.03	Выполнение заданий по теме «Основные конструкции языка, коллекции» (анализ задач; поиск информации для решения; решение задач; представление результатов в требуемом формате)	
4-5	02.03 – 14.03	Выполнение заданий по теме «Строки, работа с файлами» (анализ задач; поиск информации для решения; решение задач; представление результатов в требуемом формате)	
6-7	16.03 – 28.03	Выполнение заданий по теме «Основы ООП» (анализ задачи; поиск информации для решения; решение задачи; представление результатов в требуемом формате)	
8-9	30.03 – 11.04	Выполнение заданий по теме «Стандартные библиотеки языка программирования» (анализ задач; поиск информации для решения; решение задач; представление результатов в требуемом формате)	
10-11	13.04 – 25.04	Выполнение заданий по теме «Графический интерфейс и внешние библиотеки» (анализ задачи; поиск информации для решения; решение задачи; представление результатов в требуемом формате)	
12-14	27.04 – 16.05	Использование сторонних API для создания приложений, (анализ задачи; поиск информации для решения; решение задачи; представление результатов в требуемом формате)	
15-16	18.05 – 31.05	Оформление материалов отчета в полном соответствии с требованиями на оформление письменных учебных работ студентов	

Руководитель практики от кафедры

/Габриелян Г.А. /

Обучающийся

/ Властюк А.В. /

Согласовано:

Заведующий кафедрой

/Андрианова Е.Г., к.т.н., доцент/

1) Тема «Основные конструкции языка, коллекции»

Задача 1

Условие задачи: Напишите функцию `sum_cube(n, m)`, которая будет вычислять сумму кубов чисел в заданном диапазоне, начиная с меньшего (но не включая его) до большего (включая). Первый аргумент не обязательно должен быть большим числом. Если оба числа совпадают, тогда диапазон пуст и результат должен быть 0.

Решение задачи:

```
def sum_cube(n, m):  
    s=0  
    if n>m :  
        n,m=m,n  
    if n==m:  
        return 0  
    for i in range (n+1,m+1):  
        s=s+i**3  
    return s
```

Тестирование:

№	Тест	Ожидаемое значение	Полученное значение
1	2, 3	27	27
2	3, 2	27	27
3	0, 4	100	100
4	17, 14	12384	12384
5	9, 9	0	0
6	5, 0	225	225

Задача 2

Условие задачи: Написать функцию `max_3_sum`, которая возвращает наибольшую сумму трех подряд идущих элементов списка.

Решение задачи:

```
def max_3_sum(arr):  
    k = 0  
    for i in range(0, len(arr) - 2):  
        if arr[i] + arr[i + 1] + arr[i + 2] > k:  
            k = arr[i] + arr[i + 1] + arr[i + 2]
```

return k

Тестирование:

№	Тест	Ожидаемое значение	Полученное значение
1	[1, 2, 3, 4, 5]	12	12
2	[10, 10, 30, 20, 10, 15, 30]	60	60
3	[20, 10, -80, 10, 10, 15, 30]	55	55
4	[10, -80, -10, -10, 15, -35, 20]	0	0

Задача 3

Условие задачи: GPS в машине каждые s секунд записывает в список x значение пройденного расстояния с начала пути. Написать функцию $\text{gps}(s, x)$, которая возвращает максимальную среднюю скорость среди интервалов.

Решение задачи:

```
def gps(s, x):
    max=0
    for i in range(len(x)-1):
        if ((x[i+1]-x[i])*3600/s)>max:
            max=int((x[i+1]-x[i])*3600)/s
    return int(max)
```

Тестирование:

№	Тест	Ожидаемое значение	Полученное значение
1	12, [0.0, 0.24, 0.48, 0.72, 0.96, 1.2, 1.44, 1.68, 1.92, 2.16, 2.4]	72	72
2	17, [0.0, 0.02, 0.44, 0.66, 0.88, 1.1, 1.32, 1.54, 1.76]	88	88
3	18, [0.0]	0	0

Задача 4

Условие задачи: Задан список, состоящий из не менее трех целых чисел. Список содержит или все четные числа кроме одного или все нечетные числа кроме одного. Написать функцию `find_outlier`, которая возвращает число-исключение

Решение задачи:

```
def find_outlier(integers):
    k = 0
    s = 0
    for i in range(0, len(integers)):
        if integers[i] % 2 == 0:
            k = k + 1
        else:
            s = s + 1
    if s == 1:
        for i in range(0, len(integers)):
            if integers[i] % 2 == 1:
                return integers[i]
    else:
        for i in range(0, len(integers)):
            if integers[i] % 2 == 0:
                return integers[i]
```

Тестирование:

№	Тест	Ожидаемое значение	Полученное значение
1	12, [0.0, 0.24, 0.48, 0.72, 0.96, 1.2, 1.44, 1.68, 1.92, 2.16, 2.4]	72	72
2	17, [0.0, 0.02, 0.44, 0.66, 0.88, 1.1, 1.32, 1.54, 1.76]	88	88
3	18, [0.0]	0	0

Задача 5

Условие задачи: Создать список (коллекция фильмов), состоящий из словарей (фильмы). Словари должны содержать как минимум 5 полей (например, номер, название, год выхода...). В список добавить хотя бы 10 словарей.

Конструкция вида:

```
films = [{ "id" : 123456, "title" : "Титаник", "year" : "1997",... } , { ... }, { ... },  
...]
```

Реализовать функции:

- вывода информации о всех фильмах;
- вывода информации о фильме по введенному с клавиатуры номеру;
- вывода количества фильмов, новее введенного года;
- обновлении всей информации о фильме по введенному номеру;
- удалении фильма по номеру.

Решение задачи:

```
films = [{"id" : 11, "Название" : "Титаник", "Год" : "1997", "Режиссер": "Джеймс  
Кэмерон", "Страна создания": "США", "Длительность": "3:14" } ,  
        {"id" : 22, "Название" : "Великая красота", "Год" : "2013", "Режиссер": "Паоло  
Соррентино", "Страна создания": "Италия, Франция", "Длительность": "2:21" },  
        {"id" : 32, "Название" : "Искушение", "Год" : "2007", "Режиссер": "Джо Райт",  
"Страна создания": "Великобритания, Франция", "Длительность": "2:00" },  
        {"id" : 421, "Название" : "Леди Берд", "Год" : "2017", "Режиссер": "Грета  
Гервиг", "Страна создания": "США", "Длительность": "1:34" },  
        {"id" : 51, "Название" : "Рука Бога", "Год" : "2021", "Режиссер": "Паоло  
Соррентино", "Страна создания": "Италия", "Длительность": "2:10" },  
        {"id" : 61, "Название" : "Красивый мальчик", "Год" :  
"2018", "Режиссер": "Феликс ван Грунинген", "Страна  
создания": "США", "Длительность": "2:00" },  
        {"id" : 72, "Название" : "1+1", "Год" : "2011", "Режиссер": "Оливье Накаш",  
"Страна создания": "Франция", "Длительность": "1:52" },  
        {"id" : 82, "Название" : "Схватка", "Год" : "1995", "Режиссер": "Майкл Манн",  
"Страна создания": "США", "Длительность": "2:51" },  
        {"id" : 91, "Название" : "Джонни Д.", "Год" : "2009", "Режиссер": "Майкл  
Манн", "Страна создания": "США", "Длительность": "2:20" },  
        {"id" : 101, "Название" : "Кокаин", "Год" : "2001", "Режиссер": "Тед Демме",  
"Страна создания": "США", "Длительность": "2:03" },  
        {"id" : 111, "Название" : "Начало", "Год" : "2010", "Режиссер": "Кристофер  
Нолан", "Страна создания": "США, Великобритания", "Длительность": "2:28" }]
```

```
def info(films):  
    for perDict in films:  
        for key in perDict:  
            print(key, " - ", perDict[key], ";", end = " ")  
        print()  
  
def infoOutput(films):  
    for i in range(len(films)):  
        print(i+1, ". ", films[i]["id"], "\t", films[i]["Название"])  
        Num = int(input("Введите номер id, информацию о котором нужно вывести:"))  
        s = ''  
        for i in range(len(films)):  
            if Num==films[i]["id"]:  
                for key in films[i]: s+=(key + " - " + str(films[i][key]))+"\n")  
        print(s)  
  
def year(films):
```



```

k=0
minYear = int(input("Введите год "))
for perDict in films:
    if int(perDict["Год"]) > minYear:
        k+=1
print("Количество фильмов новее этого года: ",k)
print("\n\n\n")

def change(films):
    k = 1
    for i in range(len(films)):
        print(i + 1, ". ", films[i]["id"], "\t", films[i]["Название"])
    Num = int(input("Введите номер id фильма, информацию о котором нужно поменять."))
    for i in range(len(films)):
        if Num==films[i]["id"]:
            for key in films[i]:
                print("Старый ",key," - ",films[i][key])
                films[i][key] = input("Введите новое значение для " + str(key) +
"\t")
            print(films[i])
            return 0
    print("Нет фильма с таким id")

def dell(films):
    for i in range(len(films)):
        print(i + 1, "\t", films[i]["id"], "\t", films[i]["Название"])
    n = int(input("Введите id фильма, которое нужно удалить "))
    for i in range(len(films)):
        if films[i]["id"] == n:
            print("Удаление фильма", films[i]["Название"])
            films.pop(i)
            for i in range(len(films)):
                print(i + 1, "\t", (films[i]["Название"]))
            return 0
    print("Нет фильма с таким id")

def numfunct():
    print(" 1. Вывод информации о всех фильмах\n 2. Вывод информации о фильме по
введенному с клавиатуры номеру \n 3. Вывод количества фильмов новее введенного года\n
4. Обновление всей информации о фильме по введенному номеру \n 5. Удаление фильма по
номеру\n 0. Закончить программу")
    n = int(input("Введите номер : "))
    if n==1:
        info(films)
    elif n == 2:
        infoOutput(films)
    elif n == 3:
        year(films)
    elif n == 4:
        change(films)
    elif n == 5:
        dell(films)
    elif n == 0:
        return 0
    else:
        print("Неверное значение номера. Попробуйте еще раз \n\n")
    numfunct()

numfunct()

```

Тестирование:

Тестирование функции info() – Вывод информации о фильмах:

```
1. Вывод информации о всех фильмах
2. Вывод информации о фильме по введённому с клавиатуры номеру
3. Вывод количества фильмов новее введённого года
4. Обновление всей информации о фильме по введённому номеру
5. Удаление фильма по номеру
0. Закончить программу
Введите номер : 1
id - 11 ; Название - Титаник ; Год - 1997 ; Режиссер - Джеймс Кэмерон ; Страна создания - США ; Длительность - 3:14 ;
id - 22 ; Название - Великая красота ; Год - 2013 ; Режиссер - Паоло Соррентино ; Страна создания - Италия, Франция ; Длительность - 2:21 ;
id - 32 ; Название - Искупление ; Год - 2007 ; Режиссер - Джо Райт ; Страна создания - Великобритания, Франция ; Длительность - 2:00 ;
id - 421 ; Название - Леди Берд ; Год - 2017 ; Режиссер - Грета Гервиг ; Страна создания - США ; Длительность - 1:34 ;
id - 51 ; Название - Рука Бога ; Год - 2021 ; Режиссер - Паоло Соррентино ; Страна создания - Италия ; Длительность - 2:10 ;
id - 61 ; Название - Красивый мальчик ; Год - 2018 ; Режиссер - Феликс ван Грунинген ; Страна создания - США ; Длительность - 2:00 ;
id - 72 ; Название - 1+1 ; Год - 2011 ; Режиссер - Оливье Накаш ; Страна создания - Франция ; Длительность - 1:52 ;
id - 82 ; Название - Схватка ; Год - 1995 ; Режиссер - Майкл Манн ; Страна создания - США ; Длительность - 2:51 ;
id - 91 ; Название - Джонни Д. ; Год - 2009 ; Режиссер - Майкл Манн ; Страна создания - США ; Длительность - 2:20 ;
id - 101 ; Название - Кокаин ; Год - 2001 ; Режиссер - Тед Демме ; Страна создания - США ; Длительность - 2:03 ;
id - 111 ; Название - Начало ; Год - 2010 ; Режиссер - Кристофер Нолан ; Страна создания - США, Великобритания ; Длительность - 2:28 ;
```

Рисунок 1 - Тестирование функции вывода информации о фильмах

Тестирование функции infoOutput() - вывода информации о фильме по номеру:

```
1. Вывод информации о всех фильмах
2. Вывод информации о фильме по введённому с клавиатуры номеру
3. Вывод количества фильмов новее введённого года
4. Обновление всей информации о фильме по введённому номеру
5. Удаление фильма по номеру
0. Закончить программу
Введите номер : 2
1 . 11      Титаник
2 . 22      Великая красота
3 . 32      Искупление
4 . 421     Леди Берд
5 . 51      Рука Бога
6 . 61      Красивый мальчик
7 . 72      1+1
8 . 82      Схватка
9 . 91      Джонни Д.
10 . 101     Кокаин
11 . 111     Начало
Введите номер id, информацию о котором нужно вывести: 11
id - 11
Название - Титаник
Год - 1997
Режиссер - Джеймс Кэмерон
Страна создания - США
Длительность - 3:14
```

Рисунок 2 - Тестирование функции вывода информации о фильме по номеру

Тестирование функции year() - вывода количества фильмов, новее введённого года:

```
1. Вывод информации о всех фильмах
2. Вывод информации о фильме по введенному с клавиатуры номеру
3. Вывод количества фильмов новее введённого года
4. Обновление всей информации о фильме по введенному номеру
5. Удаление фильма по номеру
0. Закончить программу
Введите номер : 3
Введите год 1999
Количество фильмов новее этого года: 9
```

Рисунок 3 - Тестирование функции вывода количества фильмов, новее введённого года

Тестирование функции change() - обновлении всей информации о фильме по введенному номеру:

```
1 . 11      Титаник
2 . 22      Великая красота
3 . 32      Искушение
4 . 421     Леди Берд
5 . 51      Рука Бога
6 . 61      Красивый мальчик
7 . 72      1+1
8 . 82      Схватка
9 . 91      Джонни Д.
10 . 101     Кокаин
11 . 111     Начало
Введите номер id фильма, информацию о котором нужно поменять.11
Старый id - 11
Введите новое значение для id 12
Старый Название - Титаник
Введите новое значение для Название Семь жизней
Старый Год - 1997
Введите новое значение для Год 2008
Старый Режиссер - Джеймс Кэмерон
Введите новое значение для Режиссер Габриэле Муччино
Старый Страна создания - США
Введите новое значение для Страна создания США
Старый Длительность - 3:14
Введите новое значение для Длительность 1:58
```

Рисунок 4 - Тестирование функции обновления всей информации о фильме по введенному номеру

```

id - 12 ; Название - Семь жизней ; Год - 2008 ; Режиссер - Габриэле Муччино ; Страна создания - США ; Длительность - 1:58 ;
id - 22 ; Название - Великая красота ; Год - 2013 ; Режиссер - Паоло Соррентино ; Страна создания - Италия, Франция ; Длительность - 2:21 ;
id - 32 ; Название - Искупление ; Год - 2007 ; Режиссер - Джо Райт ; Страна создания - Великобритания, Франция ; Длительность - 2:00 ;
id - 421 ; Название - Леди Берд ; Год - 2017 ; Режиссер - Грета Гервиг ; Страна создания - США ; Длительность - 1:34 ;
id - 51 ; Название - Рука Бога ; Год - 2021 ; Режиссер - Паоло Соррентино ; Страна создания - Италия ; Длительность - 2:10 ;
id - 61 ; Название - Красивый мальчик ; Год - 2018 ; Режиссер - Феликс ван Грунинген ; Страна создания - США ; Длительность - 2:00 ;
id - 72 ; Название - 1+1 ; Год - 2011 ; Режиссер - Оливье Накаш ; Страна создания - Франция ; Длительность - 1:52 ;
id - 82 ; Название - Схватка ; Год - 1995 ; Режиссер - Майкл Манн ; Страна создания - США ; Длительность - 2:51 ;
id - 91 ; Название - Джонни Д. ; Год - 2009 ; Режиссер - Майкл Манн ; Страна создания - США ; Длительность - 2:20 ;
id - 101 ; Название - Кокаин ; Год - 2001 ; Режиссер - Тед Демме ; Страна создания - США ; Длительность - 2:03 ;
id - 111 ; Название - Начало ; Год - 2010 ; Режиссер - Кристофер Нолан ; Страна создания - США, Великобритания ; Длительность - 2:28 ;

```

Рисунок 5 - Список фильмов после изменения информации о фильме

Тестирование функции dell() - удаления фильма по номеру:

```

1      12      Семь жизней
2      22      Великая красота
3      32      Искупление
4      421     Леди Берд
5      51      Рука Бога
6      61      Красивый мальчик
7      72      1+1
8      82      Схватка
9      91      Джонни Д.
10     101     Кокаин
11     111     Начало
Введите id фильма, которое нужно удалить 12
Удаление фильма Семь жизней
1      Великая красота
2      Искупление
3      Леди Берд
4      Рука Бога
5      Красивый мальчик
6      1+1
7      Схватка
8      Джонни Д.
9      Кокаин
10     Начало

```

Рисунок 6 - Тестирование функции удаления фильма по номеру

2) Тема «Строки, работа с файлами»

Задача 1

Условие задачи: Написать функцию fragmentation, которая получает на вход строку s и целое положительное число count, разбивает строку на

подстроки по count символа и возвращает список этих строк. Если в последней подстроке недостаточно символов, дополнить ее символами нижнего подчеркивания.

Решение задачи:

```
def fragmentation(s, count):
    ss=""
    a=[]
    for i in range(0,len(s),count):
        ss+=s[i:i+count]
        if len(ss)<count:
            for i1 in range(len(ss),count):
                ss+="_"
        a.append(ss)
        ss=""
    if len(s) == 0:
        a = "_" * count
    return a
```

Тестирование:

№	Тест	Ожидаемое значение	Полученное значение
1	"this is my string",4	["this", " is ", "my s", "trin", "g___"]	["this", " is ", "my s", "trin", "g___"]
2	"Tomorrow is going to be raining.", 8	["Tomorrow", " is goin", "g to be ", "raining."]	["Tomorrow", " is goin", "g to be ", "raining."]
3	"", 5	"_____"	"_____"
4	"abcdef", 1	["a", "b", "c", "d", "e", "f"]	["a", "b", "c", "d", "e", "f"]

Задача 2

Условие задачи: Написать функцию sum_of_fractions, которая получает вещественное число и возвращает строку - сумму слагаемых числа в виде дробей. Между слагаемыми поставить символ +, все отделить пробелами

Решение задачи:

```
def sum_of_fractions(num):

    a=""
    ss=""
```

```

num=str(num)
for i in range(0,len(num)):
    if num[i]==".":
        c=i
        if num[:c]!="0":
            a=a+num[:c]+" + "

        for i1 in range(c+1,len(num)):
            if num[i1]=="0":
                continue
            b=10**(i1-c)
            if (len(num)-1)!=i1:
                ss=num[i1]+"/"+str(b)+" + "
            else:
                ss = num[i1] + "/" + str(b)
            a=a+ss
        break
return a

```

Тестирование:

№	Тест	Ожидаемое значение	Полученное значение
1	1.24	'1 + 2/10 + 4/100'	'1 + 2/10 + 4/100'
2	7.304	'7 + 3/10 + 4/1000'	'7 + 3/10 + 4/1000'
3	0.04	'4/100'	'4/100'

Задача 3

Условие задачи: Написать функцию parse_molecule, которая в строке, представляющей из себя молекулярную формулу, подсчитывает количество всех атомов и возвращает результат в виде словаря.

Решение задачи:

```

def parse_molecule(s):
    ss=''
    a={}
    indkv = ""
    indkr = ""
    nkvs=0
    kkvs=0
    nkrs=0
    kkrs=0
    for i in range(0, len(s)):
        if (s[i]=="("):
            nkvs=i+1
            for i1 in range(i,len(s)):
                if s[i1]==")":
                    kkvs=i1

                    for i2 in range(i1+1,len(s)):
                        if s[i2].isdigit():
                            indkv+=s[i2]

```



```

        else:
            break
    if (s[i] == "("):
        nkrs = i + 1
        for i1 in range(i, len(s)):
            if s[i1] == ")":
                kkrs = i1

                for i2 in range(i1 + 1, len(s)):
                    if s[i2].isdigit():
                        indkr += s[i2]
                    else:
                        break
    ind=""
    if len(indkr)!=0:
        for i in range(nkrs, kkrs):
            if (s[i].istitle()):
                ss=s[i]
                ind=""
                if s[i+1].islower():
                    ss=ss+s[i+1]
                    if (i != (len(s) - 2)):
                        if s[i+2].isdigit():
                            k=i+2
                            while s[k].isdigit():
                                ind+=s[k]
                                k+=1
                            else:
                                ind="1"
                        elif s[i+1].isdigit():
                            k = i + 1
                            while s[k].isdigit():
                                ind += s[k]
                                k += 1
                        else:
                            ind="1"
                    a[ss]=int(ind)
                    ss=""
                    ind=''
        for key in a:
            a[key]=int(a[key])*int(indkr)
    if len(indkv)!=0:
        for i in range(nkvs, kkvs):
            if (i>=nkrs-1)and(i<=kkrs):
                continue
            if (s[i].istitle()):
                ss=s[i]
                ind=""
                if s[i+1].islower():
                    ss=ss+s[i+1]
                    if (i != (len(s) - 2)):
                        if s[i+2].isdigit():
                            k=i+2
                            while s[k].isdigit():
                                ind+=s[k]
                                k+=1
                            else:
                                ind="1"
                        elif s[i+1].isdigit():
                            k = i + 1
                            while s[k].isdigit():
                                ind += s[k]

```

```

        k += 1
    else:
        ind="1"
        a.setdefault(ss,0)
        a[ss]=a[ss]+int(ind)
    for key in a:
        a[key] = int(a[key]) * int(indkv)
    for i in range(0,len(s)):
        if (nkvs>0 and i>=nkvs) or (nkrs>0 and i>=nkrs):
            continue

    if (s[i].istitle()):
        ss = s[i]
        ind = ""
        if (i != (len(s) - 1)):
            if s[i+1].islower():
                ss=ss+s[i+1]
                if (i != (len(s) - 2)):
                    if s[i+2].isdigit():
                        k=i+2
                        while s[k].isdigit() and k!=len(s)-1:
                            ind+=s[k]
                            k+=1
            elif s[i+1].isdigit():
                k = i + 1
                while s[k].isdigit() and k!=len(s)-1:
                    ind += s[k]
                    k += 1
        if len(ind)==0:
            ind="1"
        a.setdefault(ss, 0)
        a[ss] = a[ss] + int(ind)
    print(a)
    return a

```

Тестирование:

№	Тест	Ожидаемое значение	Полученное значение
1	"H2O"	{ 'H': 2, 'O': 1 }	{ 'H': 2, 'O': 1 }
2	"Mg(OH)2"	{ 'Mg': 1, 'O': 2, 'H': 2 }	{ 'O': 2, 'H': 2, 'Mg': 1 }
3	"K4[ON(SO3)2]2"	{ 'K': 4, 'O': 14, 'N': 2, 'S': 4 }	{ 'S': 4, 'O': 14, 'N': 2, 'K': 4 }

Задача 4

Условие задачи: Создать txt-файл, вставить туда любую англоязычную статью из Википедии.

Реализовать одну функцию, которая выполняет следующие операции:

- прочитать файл построчно;

- непустые строки добавить в список;
 - удалить из каждой строки все цифры, знаки препинания, скобки, кавычки и т.д. (остаются латинские буквы и пробелы);
 - объединить все строки из списка в одну, используя метод join и пробел, как разделитель;
 - создать словарь вида {“слово”: количество, “слово”: количество, ... }
- для подсчета количества разных слов,
- где ключом будет уникальное слово, а значением - количество;
- вывести в порядке убывания 10 наиболее популярных слов, используя форматирование (вывод примерно следующего вида: “ 1 place --- sun --- 15 times \n....”);
 - заменить все эти слова в строке на слово “PYTHON”;
 - создать новый txt-файл;
 - записать строку в файл, разбивая на строки, при этом на каждой строке записывать не более 100 символов при этом не делить слова.

Решение задачи:

```
def wiki_function():
    # -----
    -----
    # Вывод текста построчно
    file=open('a.txt')
    k=0
    s=[]
    # print("\n\n\nВывод текста построчно\n\n\n")
    for row in file:
        # print(row)
    # -----
    #Список без пустых строк
    if row!="\n":
        s.append(row)

    # print("\n\n\n\nСписок без пустых строк:\n\n\n",s,"\n\n\n")
    # print("-----")
    # -----
    -----
    # Список без лишних символов
    k=0
    for row in s:
        for i in row:
            if i.isalpha()==False and i!=" ":
                row=row.replace(i,"")
            s[k]=row
            k+=1
    # print("\n\n\n\nСписок без лишних символов:\n\n\n",s)
    #-----
```

```

# Объединенный список в строку
s_str = " ".join(s)

# print("\n\n\nОбъединенный список в строку:\n\n\n", s_str)
# print("\n\n\n-----")

#-----

# Словарь с количеством слов
s.clear()
s=s_str.lower().split()
a={}
for i in range(0,len(s)):
    a.setdefault(s[i],0)
    if a[s[i]]==0:
        for i1 in range(0,len(s)):
            if s[i]==s[i1]:
                a[s[i]]+=1
# print("\n\n\nСловарь с количеством слов:\n\n\n", a)
# print("\n\n\n-----")
#-----

# 10 популярных слов
sor=list(a.values())
sor.sort()
k=0
top=""
for i in range(len(sor)-1,-1,-1):

    if sor[i]==sor[i-1]:
        continue
    for key in a:
        if a[key]==sor[i]:
            k+=1
            #Замена слов
            keyb=key[0].upper()+key[1:]
            for word in range(0,len(s)):
                if s[word]==key or s[word]==keyb:
                    s[word]="PYTHON"
            #-----
            top+='{0} place --- {1} --- {2} times\n'.format(k,key,sor[i])
            if (k == 10):
                break
        if (k==10):
            break
    print("\n\n\n10 популярных слов:\n\n\n", top)
    print("\n\n\n-----")
    s_str=" ".join(s)
# print("\n\n\nТекст с замененными словами :\n\n\n", s_str)
# print("\n\n\n-----")
my_file=open("b.txt", "w")
ss=""
k=0
#-----
# Создание txt файла и запись в него
for i in range(0,len(s_str)):
    if i-k==99:
        if s_str[i]==" " or s_str[i+1]==" " or len(s_str)==i-1:
            my_file.write(s_str[k:i+1]+"\\n")
            if s_str[i]==" ":
                k=i+1
            elif s_str[i+1]==" ":
                k=i+2

```

```
        else:
            for i1 in range(i,k,-1):
                if s_str[i1]==" ":
                    my_file.write(s_str[k:i1+1] + "\n")
                    k = i1+1
                    break
            elif len(s_str)-1==i:
                my_file.write(s_str[k:i + 1])
    return 1
# Вызов функции
wiki_function()
```

Тестирование:

Статья в википедии:

The arts are a very wide range of human practices of creative expression, storytelling and cultural participation. They encompass multiple diverse and plural modes of thinking, doing and being, in an extremely broad range of media. Both highly dynamic and a characteristically constant feature of human life, they have developed into innovative, stylized and sometimes intricate forms. This is often achieved through sustained and deliberate study, training and/or theorizing within a particular tradition, across generations and even between civilizations. The arts are a vehicle through which human beings cultivate distinct social, cultural and individual identities, while transmitting values, impressions, judgments, ideas, visions, spiritual meanings, patterns of life and experiences across time and space.

Prominent examples of the arts include architecture, visual arts (including ceramics, drawing, filmmaking, painting, photography, and sculpting), literary arts (including fiction, drama, poetry, and prose), performing arts (including dance, music, and theatre), textiles and fashion, folk art and handicraft, oral storytelling, conceptual and installation art, criticism, and culinary arts (including cooking, chocolate making and winemaking). They can employ skill and imagination to produce objects, performances, convey insights and experiences, and construct new environments and spaces.

The arts can refer to common, popular or everyday practices as well as more sophisticated and systematic, or institutionalized ones. They can be discrete and self-contained, or combine and interweave with other art forms, such as the combination of artwork with the written word in comics. They can also develop or contribute to some particular aspect of a more complex art form, as in cinematography.

By definition, the arts themselves are open to being continually re-defined. The practice of modern art, for example, is a testament to the shifting boundaries, improvisation and experimentation, reflexive nature, and self-criticism or questioning that art and its conditions of production, reception, and possibility can undergo.

As both a means of developing capacities of attention and sensitivity, and as ends in themselves, the arts can simultaneously be a form of response to the world, and a way that our responses, and what we deem worthwhile goals or pursuits, are transformed. From prehistoric cave paintings, to ancient and contemporary forms of ritual, to modern-day films, art has served to register, embody and preserve our ever shifting relationships to each other and to the world.

Вывод программы:

```
10 популярных слов:

1 place --- and --- 36 times
2 place --- of --- 15 times
3 place --- the --- 12 times
4 place --- to --- 11 times
5 place --- arts --- 10 times
6 place --- a --- 9 times
7 place --- art --- 7 times
8 place --- can --- 6 times
9 place --- or --- 6 times
10 place --- as --- 6 times
```

Рисунок 7 - Вывод программы

PYTHON PYTHON are PYTHON very wide range PYTHON human practices PYTHON creative expression storytelling PYTHON cultural participation they encompass multiple diverse PYTHON plural modes PYTHON thinking doing PYTHON being in an extremely broad range PYTHON media both highly dynamic PYTHON PYTHON characteristically constant feature PYTHON human life they have developed into innovative stylized PYTHON sometimes intricate forms this is often achieved through sustained PYTHON deliberate study training and/or theorizing within PYTHON particular tradition across generations PYTHON even between civilizations PYTHON PYTHON are PYTHON vehicle through which human beings cultivate distinct social cultural PYTHON individual identities while transmitting values impressions judgments ideas visions spiritual meanings patterns PYTHON life PYTHON experiences across time PYTHON space prominent examples PYTHON PYTHON PYTHON include architecture visual PYTHON including ceramics drawing filmmaking painting photography PYTHON sculpting literary PYTHON including fiction drama poetry PYTHON prose performing PYTHON including dance music PYTHON theatre textiles PYTHON fashion folk PYTHON PYTHON handicraft oral storytelling conceptual PYTHON installation PYTHON criticism PYTHON culinary PYTHON including cooking chocolate making PYTHON winemaking they PYTHON employ skill PYTHON imagination PYTHON produce objects performances convey insights PYTHON experiences PYTHON construct new environments PYTHON spaces PYTHON PYTHON PYTHON refer PYTHON common popular PYTHON everyday practices PYTHON well PYTHON more sophisticated PYTHON systematic PYTHON institutionalized ones they PYTHON be discrete PYTHON self-contained PYTHON combine PYTHON interweave with other PYTHON forms such PYTHON PYTHON combination PYTHON artwork with PYTHON written word in comics they PYTHON also develop PYTHON contribute PYTHON some particular aspect PYTHON PYTHON more complex PYTHON form PYTHON in cinematography by definition PYTHON PYTHON themselves are open PYTHON being continually redefined PYTHON practice PYTHON modern PYTHON for example is PYTHON testament PYTHON PYTHON shifting boundaries improvisation PYTHON experimentation reflexive nature PYTHON self-criticism PYTHON questioning that PYTHON PYTHON its conditions PYTHON production reception PYTHON possibility PYTHON undergo PYTHON both PYTHON means PYTHON developing capacities PYTHON attention PYTHON sensitivity PYTHON PYTHON ends in themselves PYTHON PYTHON PYTHON simultaneously be PYTHON form PYTHON response PYTHON PYTHON world PYTHON PYTHON way that our responses PYTHON what we deem worthwhile goals PYTHON pursuits are transformed from prehistoric cave paintings PYTHON ancient PYTHON contemporary forms PYTHON ritual PYTHON modern-day films PYTHON has served PYTHON register embody PYTHON preserve our ever shifting relationships PYTHON each other PYTHON PYTHON PYTHON world

Рисунок 8 - Содержимое файла после работы программы

3) Тема «Основы объектно-ориентированного программирования»

Класс Item. Класс Item с полями название, цена. `__init__` - конструктор класса.

```
class Item():
    name=""
    price=0
    def __init__(self,name,price):
        self.name=name
        self.price=price
```

Класс Food. Класс Food наследуется от класса Item с полями название, цена. `__init__` - конструктор класса. Новые поля: `weight` (масса блюда), `cookingtime` (время приготовления), `compositionFood` (состав блюда (список ингредиентов)).

`__init__` - конструктор, с вызовом родительского конструктора.

`changeCookingTime(self,k)` - функция изменения времени приготовления.

`k` – новое время приготовления.

dellCompositionsFood(self,k) - метод удаления ингредиента. k – ингредиент, который надо удалить.

ChangeCompositionFood(self,k) – метод изменения списка ингредиентов. k – новый список ингредиентов.

addCompositionsFood(self,s) - метод добавления ингредиента. k – новый ингредиент.

printcompositionFood(self) – метод, возвращает строку со списком ингредиентов.

__str__(self) - Переопределение метода преобразования в строку для печати основной информации

PrintmenuFood(food) – Функция для вывода списка блюд.

```
import ClassItem
class Food(ClassItem.Item):
    weight=0
    cookingtime=0
    compositionFood=[]
    def __init__(self,name,price,weight,compositionFood,cookingtime):
        super().__init__(name, price)
        self.weight=weight
        self.cookingtime=cookingtime
        self.compositionFood=compositionFood
        # Вызов конструктора основного класса

    def changeCokingTime(self,k):
        self.cookingtime=k

    def dellCompositionsFood(self,k):
        for i in range(len(self.compositionFood)):
            if self.compositionFood[i].lower()==k.lower():

                self.compositionFood.pop(i)
                break
        print(self.printcompositionFood())
    def ChangeCompositionFood(self,k):
        self.compositionFood=k
        print(self.printcompositionFood())

    def addCompositionsFood(self,s):
        print(self.printcompositionFood())

        self.compositionFood.append(s)
        print(self.printcompositionFood())
    def printcompositionFood(self):
        # print("Состав " + self.name + ": " + ",
".join(self.compositionFood).lower()+"\n")
        return ("Состав " + self.name + ": " + ",
".join(self.compositionFood).lower()+"\n")
    def __str__(self):
```

```

        print("%22s%15d%25d%25d"%(self.name,self.price,self.weight,self.cookingtime))
        return("%-22s%-15d%-25d%-25d"%(self.name,int(self.price),int(self.weight),int(self.cookingtime)))
def PrintmenuFood(food):
    print("%-20s%-15s%-15s%-25s" % ("Название", "Цена(руб.)", "Масса(г.)", "Время
приготовления(мин.)"))
    if (type(food)==list):
        for i in food:
            print(str(i))
    else:
        print(str(food))

food1=[Food("Биг мак",135,205,["булочка для гамбургеров","две рубленые котлеты из
говядины","ломтик сыра","салат","соус биг мак","соленные огурцы"],5),
        Food("чизбургер",50,117,["булочка для гамбургеров","Рубленые котлета
из говядины","ломтик сыра","горчиный соус","лук ребчатый"],3),
        Food("чickenбургер",50,100,["Булочка для габургеров","Куриная
котлета","салат","соус на основе растительных масел"],3),
        Food("Картофель фри",75,100,["Картофель фри","Масло растительное
фритюрное","Соль"],4),
        Food("Чикен макнагетс",99,138,["Котлеты куриные макнагетс","Масло
растительное"],3),
        Food("Салат цезарь",199,172,["Салат","томаты чери","котлета
куриная","сыр тертый"],5)]

```

Класс Drink. Класс Drink - производный от Item. Новые поля: volume (объем напитка) , menusection (раздел меню), compositionDrink (состав напитка (словарь вида название ингредиента: количество)).

__init__ - конструктор, с вызовом родительского конструктора.

addcompositionDrink(self,name,k) - метод добавления ингредиента в состав напитка . k – количество ингредиента, name – название ингредиента.

dell(self,s) - метод удаления ингредиента. s– ингредиент, который надо удалить.

printcomposition(self) - метод, вывод строку со списком ингредиентов.

__str__(self) - Переопределение преобразования в строку для печати основной информации

PrintmenuDrink(drink) - Функция для вывода списка напитков.

```

import ClassItem
class Drink(ClassItem.Item):
    volume=0
    menusection=""
    compositionDrink={}
    def __init__(self,name,price,volume,menusection,compositionDrink):
        super().__init__(name,price)
        self.volume=volume

```

```

        self.menusection=menusection
        self.compositionDrink=compositionDrink

    def addcompositionDrink(self,name,k):
        # self.printcomposition()
        self.compositionDrink[name] = k
        print(self.printcomposition())
    def dell(self,s):

        for i in self.compositionDrink:
            if s in i:

                self.compositionDrink.pop(i)
                break
        print(self.printcomposition())

    def printcomposition(self):
        print("Состав " + self.name + ": ",end="")
        lenkey=list(dict.keys(self.compositionDrink))[len(self.compositionDrink)-1]

        for key in self.compositionDrink:
            for i in range(0,len(key)):
                if key[i]=="(":
                    if lenkey!=key:

                        print((key[:i + 1] + " " + str(self.compositionDrink[key]) +
" " + str(key[i + 1:]) + ", "),sep="", end="")
                    else:
                        print((key[:i + 1] + " " + str(self.compositionDrink[key]) +
" " + str(key[i + 1:]) + "."),sep="")

                break

    def __str__(self):
        print("%37s%32s%15s%15s"%(self.name,self.menusection,self.volume,self.price))
        return ("%37s%-32s%-15s%-15s%15s"%(self.name,self.menusection,self.volume,self.price))
    def PrintmenuDrink(drink):

        print("%-40s%-27s%-15s%-25s" % ("      Название", "Раздел меню", "Объем(г.)",
"Цена (руб.)"))
        if (type(drink)==list):
            for i in range(len(drink)):
                print(str(drink[i]))
        else:
            print(str(drink))

drink1=[Drink("Шоколадный молочный коктейль",93,300,"Молочные коктейли",{ "Сироп
шоколадный (мл.)":20,"Смесь молочная (мл.)":280}),
        Drink("Чай черный",55,200,"Горячие напитки",{ "Пакетик чая(шт.)":1,"Сахар
(г.)":2,"Вода(мл.)":200}),
        Drink("Латте",100,300,"Горячие
напитки",{ "Молоко(мл.)":50,"Сахар(г.)":2,"Сироп      Ванильный(мл.)":20,"Кофейные
зерна(г.)":1.5,"Вода(мл.)":220}),
        Drink("Кока-Кола",75,400,"Прохладительные напитки",{ "Лед(кубики)":5,"Кока-
Кола(мл.)":280}),
        Drink("Лимонад Сантори",100,200,"Прохладительные напитки",{ "Сироп с ароматом
красных ягод и лимона(мл.)":30,"Лед(кубики)":5,"Вода(мл.)":180})]

```

Класс Menu. Поля: namerest (название ресторана), adress (адрес), drink (список напитков (список экземпляров класса Drink)), food (список горячих блюд (список экземпляров класса Food)).

def __init__ - конструктор.

def __add__(self, other) - Переопределение операции сложения для добавления элемента меню. Other – элемент меню.

def __sub__(self, s) – Переопределение операции вычитания для удаления элемента меню. Other – элемент меню.

def __str__(self) - Переопределение метода преобразования в строку для печати меню.

def __len__(self) - Переопределение метода len() для получения количества пунктов меню.

def PrintMenu(self) – метод, который выводит основную информацию.

def writingToFile(self) – метод для создания txt-файла и записи всей информации в него (в том числе списков ингредиентов напитков и блюд).

def __getitem__(self, k) - Переопределение метода получения по индексу для получения напитка/блюда по индексу. k – индекс.

def __setitem__(self, k, value): Переопределение метода изменения по индексу для изменения напитка/блюда по индексу. k – индекс, value – то на что изменяем.

def __delitem__(self, key) - Переопределение метода удаления по индексу для удаления напитка/блюда по индексу. k – индекс.

```
from ClassFood import *
from ClassDrink import *
class Menu():
    namerest=""
    address=""

    drink=[]
    food=[]

    def __init__(self, namerest, address1, drink1, food1):
        self.namerest=namerest
        self.address=address1
```

```
self.drink=drink1  
self.food=food1  
  
def __add__(self,other):  
    if type(other)==Drink:  
        self.drink.append(other)  
    else:  
        self.food.append(other)  
  
def __sub__(self,s):  
    for i in range(len(self.drink)):  
        if self.drink[i]==s:  
            self.drink.pop(i)  
  
    for i in range(len(self.food)):  
        if self.food[i]==s:  
            self.food.pop(i)  
  
    self.PrintMenu()  
  
def __str__(self):  
    PrintmenuFood(self.food)  
    PrintmenuDrink(self.drink)  
  
def __len__(self):  
    return (len(self.drink)+len(self.food))  
  
def PrintMenu(self):  
    print(" МЕНЮ: ")  
    print("Название ресторана: "+self.namerest)  
    print("Адрес ресторана"+self.address)  
    for i in range(len(self.drink)):  
        print(i+1,self.drink[i].name)  
    for i in range(0,len(self.food)):  
        print(int(i+len(self.drink))+1,self.food[i].name)  
    print("\n")  
  
def writingToFile(self):  
    my_file = open("C.txt", "w")  
    my_file.write("МЕНЮ: \n")  
    my_file.write("Название ресторана: "+self.namerest+"\n")  
    my_file.write("Адрес ресторана: "+self.address+"\n")  
    my_file.write("Название"+" "*30+"Раздел меню"+" "*15+"Объем(г.)"+" "*6+"Цена (руб.)"+" "*15+"\n")  
    for i in range(len(self.drink)):  
        my_file.write(str(self.drink[i]))  
        my_file.write("Состав " + self.drink[i].name + ": ")  
  
        for key in self.drink[i].compositionDrink:  
            for ii in range(0, len(key)):   
                if key[ii] == "(":  
                    my_file.write((key[:ii] + " ] " + " " + "  
str(self.drink[i].compositionDrink[key]) + " " + str(key[ii + 1:]) + ", "))  
  
                break  
            my_file.write("\n")  
my_file.write("Название"+" "*12+"Цена(руб.)"+" "*7 +"Масса(г.)"+" "*12+"Время приготовления(мин.)\n")  
for i in range(len(self.food)):  
    my_file.write(str(self.food[i]))  
    my file.write(self.food[i].printcompositionFood())
```



```

def __getitem__(self, k):
    try:
        self.PrintMenu()

        if k <= len(self.drink):
            PrintmenuDrink(self.drink[k-1])
            self.drink[k-1].printcomposition()
        elif k > len(self.drink) and k <= len(self):
            PrintmenuFood(self.food[k-len(self.drink)-1])
            self.food[k-len(self.drink)-1].printcompositionFood()
        else:
            raise ValueError("Введите число в нужном диапазоне")
    except BaseException as err:
        print("Произошла ошибка!")
        print("Тип ", type(err))
        print("Описание ", err)

    finally:
        self.PrintMenu()

def __setitem__(self, k, value):
    try:
        if k <= len(self.drink):
            self.drink[k-1].name=value[0]
            self.drink[k-1].price=value[1]
            self.drink[k - 1].volume =value[2]
            self.drink[k-1].menusection=value[3]
            self.drink[k - 1].compositionDrink=value[4]

        elif k > len(self.drink) and k <= len(self):
            self.food[k - len(self.drink) - 1].name = value[0]
            self.food[k - len(self.drink) - 1].price = value[1]
            self.food[k - len(self.drink) - 1].CokingTime=value[2]
            self.food[k - len(self.drink) - 1].weight =value[3]
            self.food[k - len(self.drink) - 1].compositionFood=value[4]

        else:
            raise ValueError("Введите число в нужном диапазоне")
    except BaseException as err:
        print("Произошла ошибка!")
        print("Тип ", type(err))
        print("Описание ", err)

    finally:
        self.PrintMenu()
def __delitem__(self, key):
    self.PrintMenu()
    try:
        if k <= len(self.drink):

            self.drink.pop(k-1)
        elif k > len(self.drink) and k <= len(self):

            self.food.pop(k - len(self.drink) - 1)
        else:
            raise ValueError("Введите число в нужном диапазоне")
    except BaseException as err:
        print("Произошла ошибка!")

```

```
        print("Тип ", type(err))
        print("Описание ", err)
        del self[k]

    finally:
        self.PrintMenu()

    self.PrintMenu()
    print("\n")

rest=Menu("Макдональдс","Москва, ул. Митинская 40",drink1,food1)
```

4) Тема «Стандартные библиотеки языка программирования»

Модуль `math` - содержит наиболее применяемые математические функции и константы. Используемые методы:

- `math.log(X, [base])` - логарифм `X` по основанию `base`.
- `math.pow(X, Y)` – возведение числа `X` в степень `Y`.
- `math.sqrt(X)` - квадратный корень из `X`
- `math.fabs(X)` - модуль `X`
- `math.pi` - $\pi = 3,1415926\dots$
- `math.e` - $e = 2,718281\dots$
- `math.radians(X)` - конвертирует градусы в радианы.
- `math.cos(X)` - косинус `X` (`X` указывается в радианах).
- `math.sin(X)` - синус `X` (`X` указывается в радианах).

Модуль `Re` для регулярных выражений в Python. Используемые методы:

- `re.sub(pattern, repl, string, max=0)` - Этот метод заменяет все вхождения `pattern` в `string` на `repl`, если не указано на `max`. Он возвращает измененную строку.
- `re.search(pattern, string, flags=0)`. Функция `re.search` возвращает объект `match` если совпадение найдено, и `None`, когда нет совпадений.

Модуль `os` - функции для работы с операционной системой, не зависящие от используемой операционной системы. Используемые функции:

- `os.mkdir()` – функция для создания папки. В скобках указывается название папки
- `os.replace()` - используется для перемещения файлов или каталогов. Первым аргументом указывается текущий путь к файлу/папке, вторым куда вы хотите переместить файл/папку.
- `Os.listdir()` - возвращает список, содержащий имена файлов и директорий в каталоге. В скобках указывается путь к папке.

Модуль `time` – модуль для работы со временем. Используемые функции:

- `time.time()` - время, выраженное в секундах с начала эпохи.

Задача 2

Условие задачи: Реализовать две функции, вычисляющие математические формулы. Параметры формул являются аргументами функций.

$$a = \ln kx + \frac{1 - 0,5k}{4\sqrt{|dz^3 - 2|}} - 0,025 + d \log_3 x$$

$$y = \frac{\sin^{1/3}(x + \varphi) + \cos x}{\pi x + 4.15 \times y^4 e^{|x-y|}}, \text{ где } \varphi = 28^\circ$$

Рисунок 9 - Формулы

Решение задачи:

```
import math
def formula1(x,k,d,z):
    a=math.log(k*x,math.e)+(1-0.5*k)/(4*math.sqrt(math.fabs(d*math.pow(z,3)-2)))-
    0.025+d*math.log(x,3)
    return a
def formula2(x,y):
    f=math.radians(28)

a=(math.pow(math.sin(x+f),1/3)+math.cos(x))/(x*math.pi+4.15*math.pow(y,4)*math.pow(ma
th.e,math.fabs(x-y)))

return (a)
```

Тестирование для первой функции:

№	Тест	Ожидаемое значение	Полученное значение
1	2, 2, 2, 2	2.6231538682628055	2.6231538682628055
2	2, 4, 2, 3	3.281632286558674	3.281632286558674

Тестирование для второй функции:

№	Тест	Ожидаемое значение	Полученное значение
1	math.pi/4, 2	0.007482669567632109	0.007482669567632109
2	Math.pi/8, 0.5	1.2092900160992606	1.2092900160992606

Задача 3

Условие задачи: Показать выполненное тестирование.

– Задача. Шифровка

Владимиру потребовалось срочно запутать финансовую документацию. Но так, чтобы это было обратимо. Он не придумал ничего лучше, чем заменить каждое целое число (последовательность цифр) на его куб. Помогите ему.

Решение задачи:

```
import re
import math
def encryption(s):
    s=int(s.group())
    s=int(math.pow(s,3))
    return str(s)
s=""
s=input()
print(re.sub('[0-9]+',encryption,s))
```

Тестирование:

№	Тест	Ожидаемое значение	Полученное значение
1	Было закуплено 12 единиц техники по 410.37 рублей.	Было закуплено 1728 единиц техники по 689210 0.50653 рублей.	Было закуплено 1728 единиц техники по 689210 0.50653 рублей.

Задача 4

Условие задачи:

- Собрать в папке файлы «task_****.py» – все ранее решенные задачи из тем А, В.
- Написать функцию, которая создаст папку «Ознакомительная папка» с двумя подпапками («тема А», «тема В»), переместит все файлы в правильные подпапки.
- Написать функцию, которая получает адрес ранее созданной папки «Ознакомительная папка» и выполнит обход всех подпапок и:
 - чтение всех «task_****.py» файлов, нахождение в тексте названия функции и параметров
 - программный запуск и выполнение данных файлов, подсчет времени выполнения

Решение задачи:

```
import os,time
def create():
    op='C:/Users/ВАЛЕРИЙ/Desktop/op/Ознакомительная практика'
    os.mkdir(op)
    os.mkdir(op+'/Тема А')
    os.mkdir(op+'/Тема В')

    task="C:/Users/ВАЛЕРИЙ/Desktop/op/task"
    for i in os.listdir(task):
        if i[5]=="A":
            os.replace(task+"/"+i, 'C:/Users/ВАЛЕРИЙ/Desktop/op/Ознакомительная
практика/Тема А/'+i)
        elif i[5]=="B":
            os.replace(task + "/" + i, 'C:/Users/ВАЛЕРИЙ/Desktop/op/Ознакомительная
практика/Тема В/'+i)
    return op

def search(f):
    a=[]
    while (True):
        k=f.find("def ")
        if k>0:
            for i in range(k,len(f)):
                if f[i]=="\n":
                    a.append(f[k+4:i-1])

                    f=f[k+1:]
                    break
            else:
                return a
```

```

def read(s):

    for folder in os.listdir(s):
        print("folder "+"'"+folder+"'")
        for file in os.listdir(s+"/"+folder):
            print(">>>script "+"'"+file+"'")

            with open(s+"/"+folder+"/"+file,'r',encoding='utf-8') as f:
                f=f.read()

                funct=search(f)
                print('>>> >>>function "'+ "', '".join(funct)+'"')

            print(">>> >>>",end="")
            start=time.time()
            exec (f)

            print(">>> >>>time",time.time()-start)

op=create()
read(op)

```

5) Тема «Графический интерфейс и внешние библиотеки»

Tkinter - это графическая библиотека, позволяющая создавать программы с оконным интерфейсом. Используемые функции:

- Tk является базовым классом любого Tkinter приложения. При создании объекта этого класса запускается интерпретатор tcl/tk и создаётся базовое окно приложения.
- Title() – в скобках указывается название приложения.
- ttk.Notebook — еще один новый виджет из модуля ttk. Он позволяет добавлять разные виды отображения приложения в одном окне, предлагая после этого выбрать желаемый с помощью клика по соответствующей вкладке.
- Tkinter.Frame() используется для группировки виджетов в окне
- Виджет Combobox предназначен для отображения списка значений, их выбора или изменения пользователем.
- Label - это виджет, предназначенный для отображения какой-либо надписи без возможности редактирования пользователем.
- grid() – используется для указания расположения

- Entry() - это виджет, позволяющий пользователю ввести одну строку текста.
- Text() - это виджет, который позволяет пользователю ввести любое количество текста.
- Виджет Button - самая обыкновенная кнопка, которая используется в тысячах программ
- Combobox.get() – получение значения combobox.
- text.get() – получение значение введенное в поле Text()

Модуль urllib.request определяет функции и классы, которые помогают открывать URL-адреса (в основном HTTP).

Модуль xml - содержит встроенные XML инструменты для парсинга, к которым вы можете получить доступ.

Модуль Matplotlib - это основная библиотека для построения графиков.

Модуль datetime - модуль для работы с датой и временем в python.

Решение задачи:

Создание оформления и запись в словарь валют, полученных с сайта ЦБ:

```
root=Tk()
root.title("Конвертер валют")
# root.geometry('600x350')
tab_control=Notebook(root)
tab1=Frame(tab_control)
tab2=Frame(tab_control)
tab_control.add(tab1,text="Калькулятор валют")
tab_control.add(tab2,text="Динамика курса")
tab_control.pack(expand = True, fill = BOTH)
combobox1=Combobox(tab1)
combobox1.grid(pady=10,padx=10,column=0,ipadx=70)
combobox2=Combobox(tab1)
combobox2.grid(padx=10,pady=5,ipadx=70)
text=Entry(tab1)
data=datetime.date.today()

r=urllib.request.urlopen("http://www.cbr.ru/scripts/XML_daily.asp?date_req="+data.strftime('%d/%m/%Y'))
dom=ET.parse(r).getroot()
valutes={"Российский рубль":1}
valutesID={}
for i in dom:
```

```

        valutes[i[3].text] = float(i[4].text.replace(',', '.')) / int(i[2].text)
        valutesID[i[3].text] = i.attrib['ID']
    combobox1['values']=list(valutes.keys())
    combobox2['values']=list(valutes.keys())
    combobox1.set(combobox1['value'][0])
    combobox2.set(combobox2['value'][1])

    label1=Label(tab1,text="")
    label1.grid(row=1,column=1)
    text.grid(row=0,column=1,padx=20)
    button=Button(tab1,text="Конвертировать",command=convert)
    label=Label(tab2,text="Валюта")
    label.grid(column=0,row=0,padx=13)
    button.grid(row=0,column=2)
    combobox3=Combobox(tab2)
    combobox3.grid(column=0,row=1,ipadx=65)

    combobox3['value']=list(valutes.keys())
    combobox3.set(combobox3['value'][1])
    label=Label(tab2,text="Период")
    label.grid(column=1,row=0,padx=40)
    label=Label(tab2,text="Выбор периода")
    label.grid(row=0,column=2,padx=20)

```

Функция, которая конвертирует валюты:

```

def convert():
    a=valutes[combobox1.get()]/valutes[combobox2.get()]
    label1['text']=round(float(text.get()) * a, 3)

```

Построение графика:

```

def graph():
    period=combobox.get()
    valute=combobox3.get()
    url = "https://www.cbr.ru/scripts/XML_dynamic.asp?date_req1=" \
        + period[:10] + "&date_req2=" + period[11:] + "&VAL_NM_RQ=" \
        + valutesID[valute]

    r1 = urllib.request.urlopen(url)
    dom1 = ET.parse(r1).getroot()
    dict_graph = {}
    for tag in dom1:
        dict_graph[tag.attrib['Date']] = float(tag[1].text.replace(',', '.')) /
int(tag[0].text)

    x = [datetime.datetime.strptime(i, '%d.%m.%Y').date() for i in dict_graph.keys()]
    y = list(dict_graph.values())

    fig = plt.figure()
    canvas = FigureCanvasTkAgg(fig, master=tab2)
    plot_widget = canvas.get_tk_widget()
    fig.clear()

    if radio_state.get()==1:
        plt.gca().xaxis.set_major_formatter(matplotlib.dates.DateFormatter('%d/%m'))
        plt.gca().xaxis.set_major_locator(matplotlib.dates.DayLocator())
    elif radio_state.get()==2:
        plt.gca().xaxis.set_major_formatter(matplotlib.dates.DateFormatter('%d/%m'))

```

```

plt.gca().xaxis.set_major_locator(matplotlib.dates.DayLocator(interval=4))
elif radio_state.get()==3:
    plt.gca().xaxis.set_major_formatter(matplotlib.dates.DateFormatter('%d/%m'))
    plt.gca().xaxis.set_major_locator(matplotlib.dates.DayLocator(interval=12))
elif radio_state.get()==4:
    plt.gca().xaxis.set_major_formatter(matplotlib.dates.DateFormatter('%m/%Y'))
    plt.gca().xaxis.set_major_locator(matplotlib.dates.DayLocator(interval=50))

fig.suptitle(valute + " к рублю")

plt.plot(x, y)

plt.grid()
plot_widget.grid(row=5, column=0)

```

Тестирование:

Рисунок 10 - Тестирование калькулятора валют

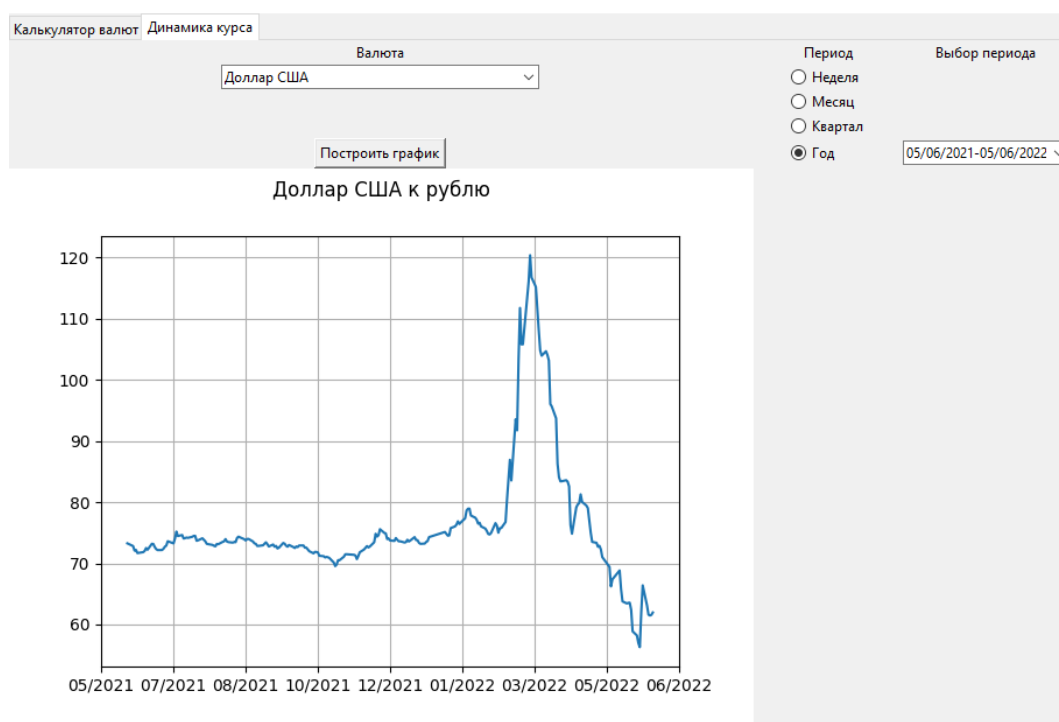


Рисунок 11 - Тестирование функции построения графиков

6) Использование сторонних API для создания приложений

Используемые модули:

- Модуль `openpyxl` - библиотека для чтения/записи форматов Office Open XML (файлов Excel 2010).
- Модуль `json` позволяет кодировать и декодировать данные в удобном формате.
- Модуль `Re` для регулярных выражений в Python.
- Модуль `requests`, используется для отправки всех видов HTTP-запросов
- Модуль `Beautiful Soup` для извлечения данных из файлов HTML и XML.
- Модуль `Matplotlib` - это основная библиотека для построения графиков.
- Модуль `datetime` - модуль для работы с датой и временем в python.
- Модуль `PIL` для работы с изображениями.
- Модуль `math` - содержит наиболее применяемые математические функции и константы.
- Модуль `vk_api` для создания скриптов для социальной сети Вконтакте

Ключевые участки кода:

Получение файлов с расписанием:

```
import requests
from bs4 import BeautifulSoup
page=requests.get("https://www.mirea.ru/schedule/")
soup=BeautifulSoup(page.text,"html.parser")

result=soup.find("div",{"class":"rasspisanie"}).find(string="Институт информационных технологий")\
    .find_parent("div").find_parent("div").findAll("a")

for x in result:

    if "ИИТ_1" in x['href']:

        f=open("file.xlsx","wb")
```

```

resp=requests.get(x['href'])
f.write(resp.content)
if "ИИТ_2" in x['href']:

    f=open("file2.xlsx","wb")
    resp=requests.get(x['href'])
    f.write(resp.content)
if "ИИТ_3" in x['href']:

    f=open("file3.xlsx","wb")
    resp=requests.get(x['href'])
    f.write(resp.content)

```

Запись расписания в словарь:

```

import openpyxl

import re,json
book=openpyxl.load_workbook("file.xlsx")

sheet=book.active
num_cols=sheet.max_column
num_rows=sheet.max_row
print(num_rows,num_cols)
reg=r"([А-Я])([А-Я])Б0-([0-9])([0-9])-(19|20|21)"

raspisanie1={}
raspisanie2={}
raspisanie_by_day={}
subject_and_info=[]
info=[]
s=""
regf=r"(([А-Яа-яёЁ]{3,20})(-?)([А-Яа-я]+)?( +)?([А-ЯЁ][., ] ?[А-Я][., ]?)?)?"

for i in range(1,num_cols):
    if re.search(reg,str(sheet.cell(row=2, column=i).value))!=None:
        group=re.search(reg,str(sheet.cell(row=2, column=i).value)).group()
        for i1 in range(4,76,2):
            if sheet.cell(row=i1, column=i).value!=None:
                s+=sheet.cell(row=i1, column=i).value
                if sheet.cell(row=i1, column=i+1).value != None:
                    s+=", " + str(sheet.cell(row=i1, column=i + 1).value)
                    if sheet.cell(row=i1, column=i + 2).value != None:
                        s += ", " + str(sheet.cell(row=i1, column=i + 2).value)
                        if sheet.cell(row=i1, column=i + 3).value != None:
                            s += ", " + str(sheet.cell(row=i1, column=i + 3).value)
                subject_and_info.append(s)
                s=""
            else:
                subject_and_info.append("--")

            if (i1-2)%12==0:
                raspisanie_by_day[str(sheet.cell(row=i1-10,
column=1).value)]=subject_and_info

                subject_and_info=[]

        raspisanie1[str(sheet.cell(row=2, column=i).value)]=raspisanie_by_day
        raspisanie_by_day={}

```

```

for i1 in range(5, 76, 2):
    if sheet.cell(row=i1, column=i).value != None:
        s += sheet.cell(row=i1, column=i).value
        if sheet.cell(row=i1, column=i + 1).value != None:
            s += ", " + str(sheet.cell(row=i1, column=i + 1).value)
            if sheet.cell(row=i1, column=i + 2).value != None:
                s += ", " + str(sheet.cell(row=i1, column=i + 2).value)
                if sheet.cell(row=i1, column=i + 3).value != None:
                    s += ", " + str(sheet.cell(row=i1, column=i + 3).value)
            subject_and_info.append(s)
            s = ""
        else:
            subject_and_info.append("--")
    if (i1 - 3) % 12 == 0:
        raspisanie_by_day[str(sheet.cell(row=i1 - 11, column=1).value)] =
subject_and_info

        subject_and_info = []
        raspisanie2[str(sheet.cell(row=2, column=i).value)] = raspisanie_by_day
        raspisanie_by_day = {}

```

Запись расписания преподавателей в словарь:

```

raspisanie_for_teacher1={}
raspisanie_for_teacher2={}

a=[]
for i in range(4,76,12):
    for i1 in range(6):
        a.append(" ")

def week():
    week_info = {}
    a = []
    for i in range(4, 76, 12):
        for i1 in range(6):
            a.append(" ")

            week_info[sheet.cell(row=i, column=1).value] = a
            a=[]
    return week_info

for i in range(1, num_cols):
    if sheet.cell(row=3, column=i).value != None and ("ФИО" in str(sheet.cell(row=3,
column=i).value)):
        for i1 in range(4, 76):

            tag = re.finditer(regf, str(sheet.cell(row=i1, column=i).value))
            for name in tag:

                if sheet.cell(row=i1, column=i) != None:

                    surname = name.group().replace(", ", ".")

```

```

        surname = surname.replace(".", ". ")
        surname = surname.replace(' ', ' ')

        if surname[len(surname) - 1] != '.' and surname[len(surname) - 1]
== '.':
            surname += '.'
            if surname[len(surname) - 1] == '-' or (len(surname) == 2 and
surname[len(surname) - 1] == ' '):
                continue
            raspisanie_for_teacher1.setdefault(surname, week())
            # print("Словарь до добавления", raspisanie_for_teacher1)
            raspisanie_for_teacher2.setdefault(surname, week())

            if i1 % 2 == 0:

                if str(sheet.cell(row=i1, column=i).value).find(surname[:3])
== 0:

                    raspisanie_for_teacher1.get(surname)[
sheet.cell(row=int(4 + 12 * int(int(i1 - 4) /
int(12))), column=1).value][
int(((i1 - 4) % 12) / 2)] \
+= str(sheet.cell(row=i1, column=i - 2).value)[
:str(sheet.cell(row=i1, column=i -
2).value).find('\n')] \
+ ", " + str(
sheet.cell(row=i1, column=i - 1).value) \
+ ", " + str(sheet.cell(row=i1, column=i +
1).value) + ", " + str(
sheet.cell(row=2, column=i - 2).value)+'\n'
                else:
                    raspisanie_for_teacher1.get(surname)[
sheet.cell(row=int(4 + 12 * int(int(i1 - 4) /
int(12))), column=1).value][
int(((i1 - 4) % 12) / 2)] \
+= str(sheet.cell(row=i1, column=i - 2).value)[
:str(sheet.cell(row=i1, column=i -
2).value).find('\n') + 1:] + ", " + str(
sheet.cell(row=i1, column=i - 1).value) \
+ ", " + str(sheet.cell(row=i1, column=i +
1).value) + ", " + str(
sheet.cell(row=2, column=i - 2).value)+'\n'
                else:
                    if str(sheet.cell(row=i1, column=i).value).find(surname[:3])
== 0:

                        raspisanie_for_teacher2.get(surname)[
sheet.cell(row=int(4 + 12 * int(int(i1 - 4) /
int(12))), column=1).value][
int(((i1 - 5) % 12) / 2)] \
+= str(sheet.cell(row=i1, column=i - 2).value)[
:str(sheet.cell(row=i1, column=i -
2).value).find('\n')] \

```

```

        + ", " + str(
        sheet.cell(row=i1, column=i - 1).value) \
        + ", " + str(sheet.cell(row=i1, column=i +
1).value) + ", " + str(
        sheet.cell(row=2, column=i - 2).value)+'\n'
    else:
        raspisanie_for_teacher2.get(surname)[
        sheet.cell(row=int(4 + 12 * int(int(i1 - 4) /
int(12))), column=1).value][
        int(((i1 - 5) % 12) / 2)] \
        += str(sheet.cell(row=i1, column=i - 2).value)[
        str(sheet.cell(row=i1, column=i -
2).value).find('\n') + 1:] + ", " + str(
        sheet.cell(row=i1, column=i - 1).value) \
        + ", " + str(sheet.cell(row=i1, column=i +
1).value) + ", " + str(
        sheet.cell(row=2, column=i - 2).value)+"\n"

```

Запись словарей с расписанием в json файл:

```

my_file=open("raspisanie1.json","w")

my_file.write(json.dumps(raspisanie1))
my_file2=open("raspisanie2.json","w")
my_file2.write(json.dumps(raspisanie2))
my_file=open("raspisanie_for_teacher1.json","w")

my_file.write(json.dumps(raspisanie_for_teacher1))
my_file=open("raspisanie_for_teacher2.json","w")

my_file.write(json.dumps(raspisanie_for_teacher2))

```

Получение погоды на сегодня:

```

respons=
requests.get("https://api.openweathermap.org/data/2.5/forecast?q=moscow&appid=93e8395
6c91b2e00ebbabe0672230693&units=metric")
    respons = respons.json()
    responses = requests.get(

"http://api.openweathermap.org/data/2.5/weather?q=moscow&appid=93e83956c91b2e00ebbabe
0672230693&units=metric")
    responses = responses.json()
    print(event.text.lower())
    vk.messages.send(

        user_id=event.user_id,
        random_id=get_random_id(),

        message="Ожидайте...")
    if event.text.lower()=="сейчас":

        upload=VkUpload(vk_session)
        attachemens=[]

image=requests.get("http://openweathermap.org/img/wn/{ }@2x.png".format(responses[ 'wea
ther' ][0][ 'icon' ]), stream=True)

```



```

print("http://openweathermap.org/img/wn/{0}@2x.png".format(responses['weather'][0]['icon']))
        photo=upload.photo_messages(photos=image.raw)[0]

attachemens.append("photo{0}_{1}".format(photo["owner_id"],photo['id']))
        derictionC = deriction_def(responses['wind']['deg'])
        print(derictionC)
        traslator = Translator(from_lang='en', to_lang="ru")
        print(attachemens)
        sspeed = sspeed_def(responses['wind']['speed'])

        ss = translate(responses['weather'][0]['description'])
        s = "{0} , температура: {1}-{2} °C\nОщущается: {8}\nДавление: {3} мм
пт. ст., влажность: {4}%\nВетер: {5}, {6} м/с, {7}". \
        format(ss,
                round(responses["main"]["temp_min"]),
                round(responses["main"] \
["temp_max"]),
                round(responses['main']['pressure'] / 1.33),
                responses['main'] \
                ['humidity'], sspeed, responses['wind']['speed'],
                derictionC,
                responses['main']['feels_like'])

        vk.messages.send(

                user_id=event.user_id,
                random_id=get_random_id(),
                attachment=attachemens[0],
                message="Погода сейчас:\n"+s)

```

Получение статистики коронавируса в определенном городе:

```

response = requests.get("https://coronavirusstat.ru/country/russia/")
soup = BeautifulSoup(response.text, "html.parser")
result = soup.findAll('div', {'class': "row border border-bottom-0
c_search_row"})
city=event.text[7:].capitalize()
print(result)
a=''
for i in range(len(result)):
    if city in result[i].find('a').text:
        s = result[i].findAll("span", {"class": "dline"})

        a+="Активных: "+s[0].text+"\n"
        a+="Вылеченно: "+s[1].text+"\n"
        a+="Умерло: "+s[2].text+"\n"
        s = result[i].findAll("div", {"class": "h6 m-0"})
        print(s)
        a+="Заразилось: "+s[0].text
        break
if a=="":
    vk.messages.send(

        user_id=event.user_id,
        random_id=get_random_id(),

        message="Город не найден")

```

```

else:

    vk.messages.send(

        user_id=event.user_id,
        random_id=get_random_id(),

        message=a[:len(a)-3])           else:

    vk.messages.send(

        user_id=event.user_id,
        random_id=get_random_id(),

        message=a[:len(a)-3])

```

Получение статистики коронавируса в Москве и построение графика:

```

response = requests.get("https://coronavirusstat.ru/country/russia/")
soup = BeautifulSoup(response.text, "html.parser")
result = soup.findAll('table')[0].find("tbody").findAll("td")

s = soup.findAll('body')[0].find("h6").find('strong').text + '\n'
k = 0
for i in result[0]:
    if k == 0:
        s += "Активных: " + i.text

    elif k == 1:
        s += "({} за сегодня)".format(i.text) + '\n'
    else:
        break
    k += 1
k = 0
for i in result[1]:
    if k == 0:
        s += "Вылечено: " + i.text

    elif k == 1:
        s += "({} за сегодня)\n".format(i.text)
    else:
        break
    k += 1
k = 0
for i in result[2]:
    if k == 0:
        s += "Умерло: " + i.text
    elif k == 1:
        s += "({} за сегодня)\n".format(i.text)
    else:
        break
    k += 1
k = 0
for i in result[3]:
    if k == 0:
        s += "Случаев: " + i.text

    elif k == 1:
        s += "({} за сегодня)".format(i.text)
    else:
        break

```

```

        k += 1
        result = soup.findAll('table')[0].find("tbody").findAll("td", {"class":
"d-none d-sm-block"})
        print(result)
        #
        # result.findAll("span", {"class": "badge badge-danger"})
        infected = []
        k = 0
        for i in result:
            if k < 10:
                if i.find("span", {"class": "badge badge-danger"}):
                    infected.append(int(i.find("span", {"class": "badge badge-
danger"}).text))
                k += 1
            else:
                break

        result = soup.findAll('table')[0].find("tbody").findAll("span", {"class":
"badge badge-success"})
        print(result)

        cured = []
        k = 0
        for i in result:
            if k < 20 and k % 2 == 1:
                print(i.text)
                cured.append(int(i.text))
            elif k > 20:
                break
            k += 1
        result = soup.findAll('table')[0].find("tbody").findAll("th")
        data = []
        k = 0
        for i in result:
            if k < 10:
                print(i.text)
                data.append(i.text[:5])
            else:
                break
            k += 1
        print(data)
        print(s)
        print(infected)
        print(cured)
        a = np.array([cured, infected])
        # df=DataFrame(a, columns=['Заболевшие', "Выздоровевшие"], index=data)
        barWidth = 0.25
        fig = plt.subplots()

        # Set position of bar on X axis
        br1 = np.arange(len(cured))
        br2 = [x + barWidth for x in br1]

        # Make the plot
        plt.bar(br1, cured, color='r', width=barWidth,
                edgecolor='grey', label='Выздоровевшие')
        plt.bar(br2, infected, color='g', width=barWidth,
                edgecolor='grey', label='Заболевшие')

        # Adding Xticks
        plt.xlabel('', fontweight='bold', fontsize=15)
        plt.ylabel('Кол-во', fontweight='bold', fontsize=15)

```

```

plt.xticks([r + barWidth for r in range(len(cured))],
           data)

plt.legend()
plt.savefig("grafic.png")

upload = VkUpload(vk_session)
photo = upload.photo_messages(photos="grafic.png")[0]
attachemens = []
attachemens.append("photo{}_{}".format(photo["owner_id"], photo['id']))
vk.messages.send(

    user_id=event.user_id,
    random_id=get_random_id(),
    attachment=attachemens[0],
    message=s)

```

Тестирование:

Получение статистики коронавируса в Москве:

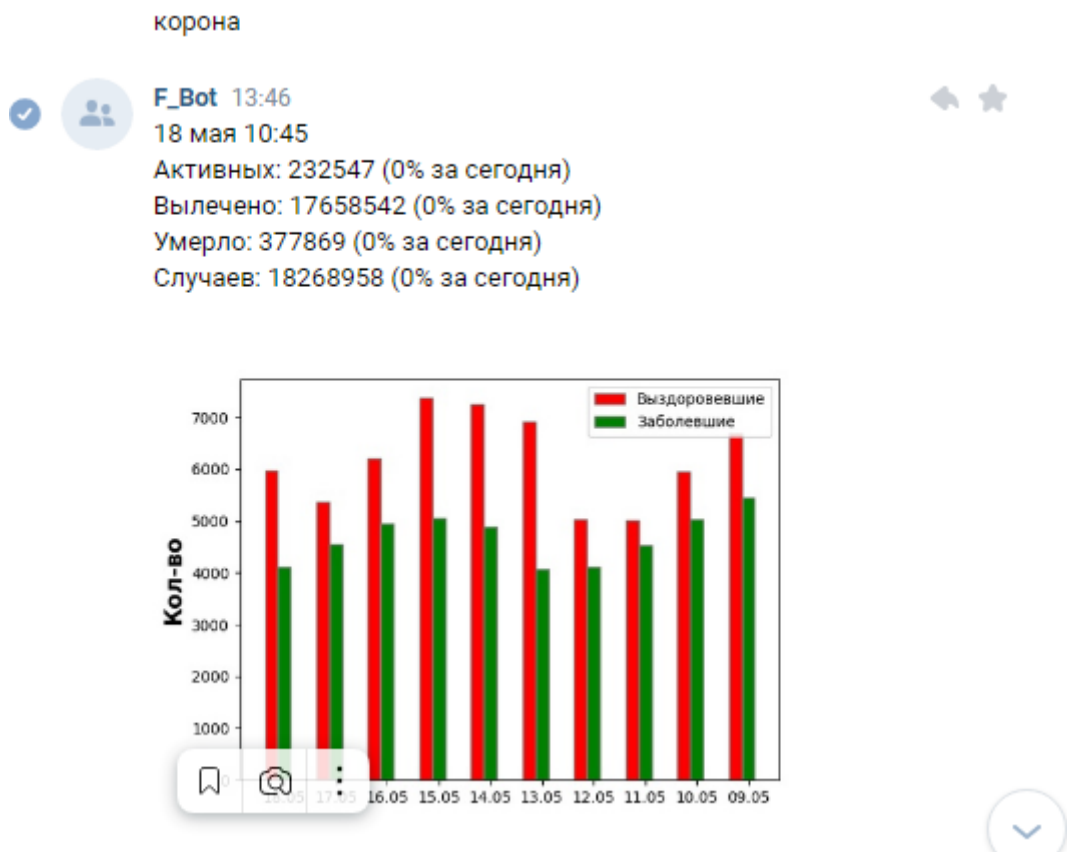


Рисунок 12 - статистика коронавируса в Москве

Получение статистики коронавируса в определенном регионе:

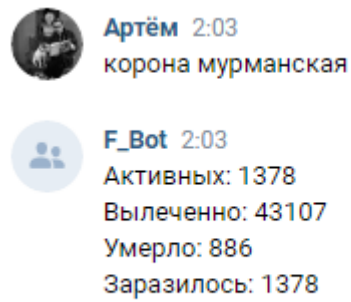


Рисунок 13 - статистика коронавируса в мурманской области

Получение погоды в Москве на данный момент:

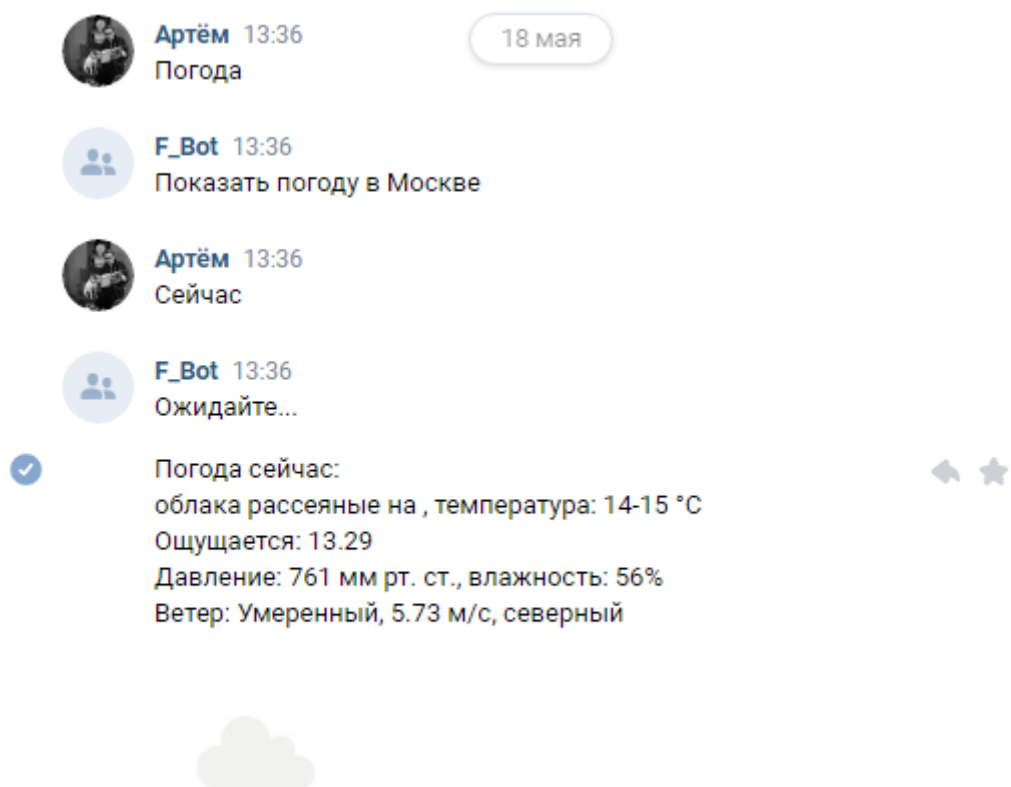


Рисунок 14 - погода в Москве

Получение расписания преподавателя:

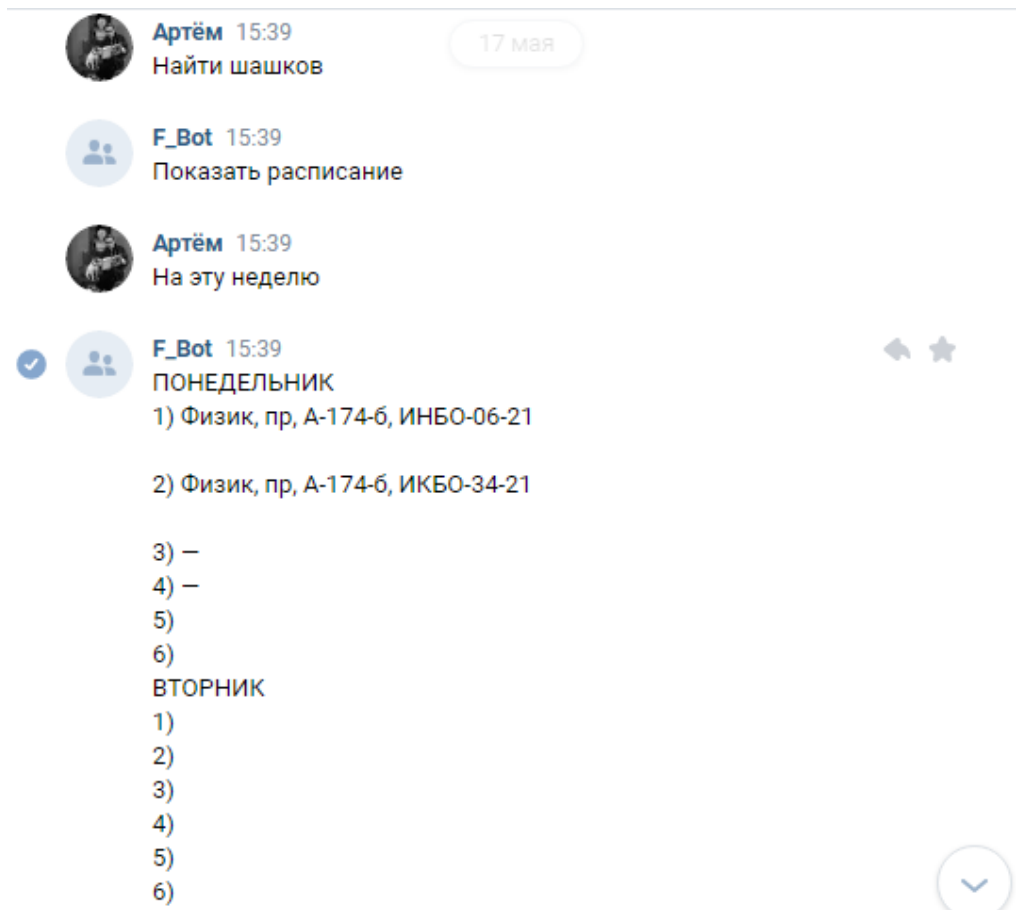


Рисунок 15 - расписание преподавателя

Получение расписания:

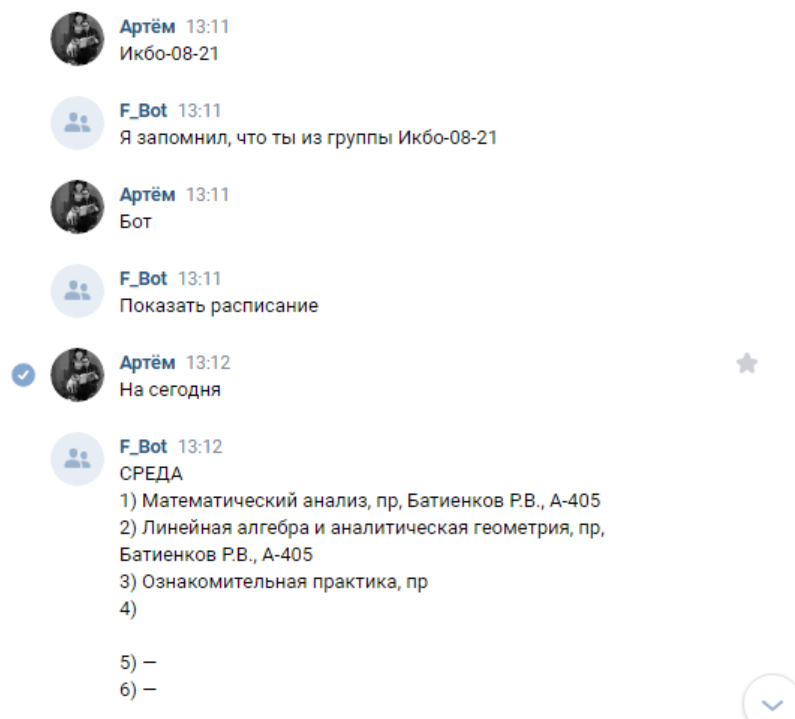


Рисунок 16 - Расписание

7) Заключение

В результате прохождения ознакомительной практики мы работали с таким языком программирования как python. Было написано множество интересных программ, которые могут пригодиться для использования в повседневной жизни. Мы изучили шесть различных тем и ознакомились с множеством библиотек в питоне, а самые интересные на мой взгляд это:

- tkinter, с помощью которой можно создавать приложения с оконным интерфейсом, мы же писали конвертер валют.
- re, очень полезный модуль для работы с текстами, можно достаточно просто извлекать или заменять нужные куски текста с помощью регулярных выражений.
- BeautifulSoup, с помощью которой можно извлекать информацию из html файлов. Мы часто использовали этот модуль для извлечения нужной информации с различных сайтов.
- vk_api, с данной библиотекой мы работали в последней теме и написали бота в Вконтакте.

За курс изучения python мы от написания простейших программ постепенно дошли до написания бота и приложения для перевода валют.