**ИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ**

Федеральное государственное бюджетное образовательное учреждение   
высшего образования

**«Сибирский государственный университет науки и технологий   
имени академика М.Ф. Решетнева»**

Институт информатики и телекоммуникаций

Кафедра информатики и вычислительной техники

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ**

Языки программирования

|  |
| --- |
| Классы в Python |

Руководитель А.В. Проскурин

подпись, дата инициалы, фамилия

Обучающийся БПИ22-02, 221219040 К.В. Трифонов

номер группы, зачетной книжки подпись, дата инициалы, фамилия

Красноярск 2023 г.

# ЦЕЛЬ РАБОТЫ

Получение практических навыков разработки и отладки программ, использующих экземпляры собственного класса.

# порядок выполнения работы

1. Ознакомиться с общей постановкой задачи.

2. Ознакомится с вариантом задания – соответствует вашему номеру в списке группы (при нехватке заданий вариант задания вычисляется как номер\_в\_списке\_группы - количество\_заданий).

3. Разработать класс согласно варианту задания.

4. Написать и отладить программу на подготовленных наборах тестовых данных.

5. Подготовить отчет по лабораторной работе. Отчет должен включать в себя:

• титульный лист;

• цель лабораторной работы;

• постановку задачи;

• текст программы с комментариями;

• демонстрацию работы программы (Снимки экрана при выполнении действий программы с описанием).

• краткие ответы на контрольные вопросы;

• выводы по лабораторной работе.

6. Защитить лабораторную работу перед преподавателем.

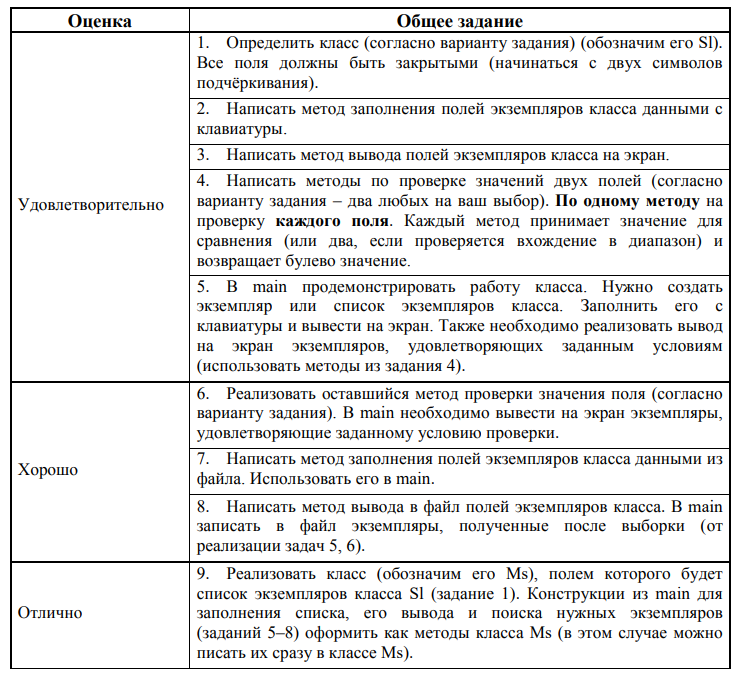
# Задания

Необходимо разработать программу на языке Python, реализующую задачу ранее решенную на языке С++ в лабораторной работе №1.

Программ должна формировать и обрабатывать список экземпляров класса (согласно варианту задания).

В реализуемой программе необходимо предусмотреть возможность демонстрации методов созданного класса.

В зависимости от оценки, на которую вы претендуете, необходимо выполнить следующие задания (Для каждой следующей оценки нужно выполнить ВСЕ предыдущие задания, если обратное не указано явно):



**Вариант 21**. Класс Aeroflot (Рейс): Поля: Пункт назначения, Номер рейса, Тип самолета, Время вылета, Дни недели. Реализовать выборки: а) списка рейсов для заданного пункта назначения; б) списка рейсов самолетов указанного типа; в) списка рейсов для заданного дня недели, время вылета которых находится в заданном интервале. Реализовать сортировку массива объектов класса на основе: - времени вылета.

# ХОД РАБОТЫ

**Текст программы:**

**Содержимое файла aero.py:**

*#Рейс*

*class* Aeroflot:

*#Конструктор с параметрами*

*def* \_\_init\_\_(*self*, *destination* ="", *number* ="", *Type* ="", *time* ="", *day* =""):

*self*.\_\_destination = destination

*self*.\_\_number = number

*self*.\_\_Type = Type

*self*.\_\_time = time

*self*.\_\_day = day

*#Сетер из строки*

*def* set\_st(*self*, *st*):

*self*.\_\_destination,*self*.\_\_number,*self*.\_\_Type,*self*.\_\_time,*self*.\_\_day = st.strip().split('\t')

*#Гетеры*

*def* get(*self*):

        return *f*"Рейс №{*self*.\_\_number} в {*self*.\_\_destination} самолётом {*self*.\_\_Type} прилетает в {*self*.\_\_time} по {*self*.\_\_day}"

*def* getU(*self*):

        return *f*"{*self*.\_\_destination}\t{*self*.\_\_number}\t{*self*.\_\_Type}\t{*self*.\_\_time}\t{*self*.\_\_day}"

*def* get\_des(*self*):

        return *self*.\_\_destination

*def* get\_type(*self*):

        return *self*.\_\_Type

*def* get\_time(*self*):

        return (*int*(*self*.\_\_time),*self*.\_\_day)

Исходный класс Aeroflot (Рейс): Поля: Пункт назначения, Номер рейса, Тип самолета, Время вылета, Дни недели. Конструктор с параметрами, методы задания полей через строку, возврата полей в виде форматированной и неформатированной строки, возврата полей пункта назначения, типа и времени (для выборок).

*#Класс массива рейсов*

*class* Flights:

*#Конструктор по умолчанию*

*def* \_\_init\_\_(*self*):

*self*.\_\_aeroflot = []

*#Ввод из файла*

*def* read(*self*,*filename*):

*self*.\_\_aeroflot = []

        with open(filename, 'r') as file:

            lines = file.readlines()

        for line in lines:

            flight = Aeroflot()

            flight.set\_st(line)

*self*.add(flight)

*#Вывод в файл*

*def* save(*self*, *filename*):

        with open(filename, 'w') as file:

            for flight in *self*.\_\_aeroflot:

                file.write(*f*"{flight.getU()}\n")

*def* add(*self*, *A*):

*self*.\_\_aeroflot.append(A)

*def* get(*self*):

        for i in *self*.\_\_aeroflot:

            print(i.get())

*#ВЫБОРКИ*

*#а) списка рейсов для заданного пункта назначения*

*def* get\_dest(*self*, *dest*):

        for i in *self*.\_\_aeroflot:

            if i.get\_des() == dest:

                print(i.get())

*#б) списка рейсов самолетов указанного типа*

*def* get\_type(*self*, *Type*):

        for i in *self*.\_\_aeroflot:

            if i.get\_type() == Type:

                print(i.get())

*#в) списка рейсов для заданного дня недели, время вылета которых находится в заданном интервале*

*def* get\_time(*self*, *day*, *time1*, *time2*):

        for i in *self*.\_\_aeroflot:

            if i.get\_time()[1] == day and *int*(i.get\_time()[0]) > *int*(time1) and *int*(i.get\_time()[0]) < *int*(time2):

                print(i.get())

*#Сортировка по полю времени вылета*

*def* sort\_by\_time(*self*):

*self*.\_\_aeroflot = sorted(*self*.\_\_aeroflot, *key*=*lambda* *flight*: flight.get\_time()[0])

Класс flights списка объектов типа Aeroflot с методами чтения, записи из файла, вывода в консоль, выборок по пункту назначения, типу, времени и сортировки списка по полю времени.

**Содержимое файла aero.py:**

from aero import \*

if \_\_name\_\_ == "\_\_main\_\_":

    a = Flights()

    while True:

        print("Главное меню")

        print("1) Добавить с консоли")

        print("2) Добавить из файла")

        print("3) Вывести в консоль")

        print("4) Вывести в файл")

        print("5) Выборка по назначению")

        print("6) Выборка по типу")

        print("7) Выборка по времени")

        print("8) Отсортировать по времени")

        print("0) <<< Выход")

        ans = *int*(input())

        match ans:

            case 1:

                des, num, ty, t1, day = input("Введите назначение, номер, тип, время вылета и день через пробел: ").split()

                m = Aeroflot(des,num,ty,t1,day)

                a.add(m)

            case 2:

                a.read("input.txt")

            case 3:

                a.get()

            case 4:

                a.save("output.txt")

            case 5:

                des = input("Введите назначение: ")

                a.get\_dest(des)

            case 6:

                ty = input("Введите тип: ")

                a.get\_type(ty)

            case 7:

                d,t1,t2 = input("Введите день, время от и до через пробел d\_t1\_t2: ").split()

                a.get\_time(d,t1,t2)

            case 8:

                a.sort\_by\_time()

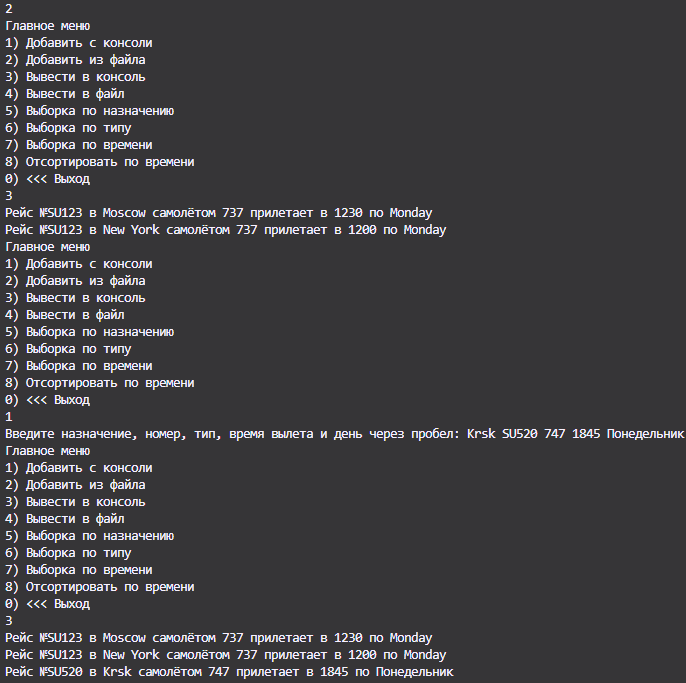
            case 0:

                break

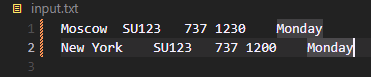
Меню функций модуля aero, для выбора используется оператор match-case.

# Тестирование

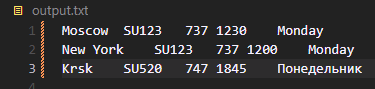
Тестирование чтения из файла input.txt, добавления одного рейса и вывода в файл output.txt**:**



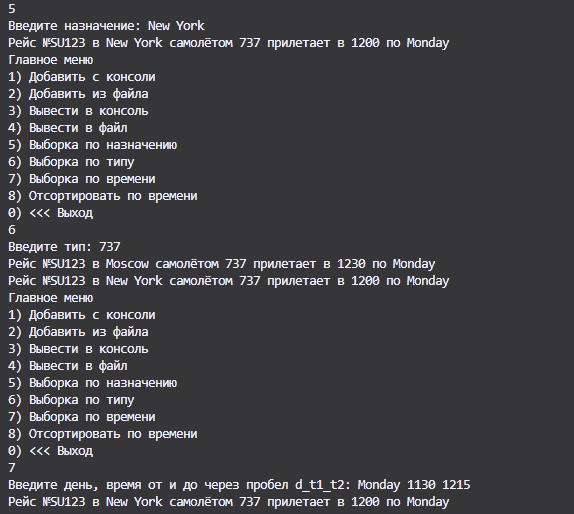
Содержимое файла input.txt:



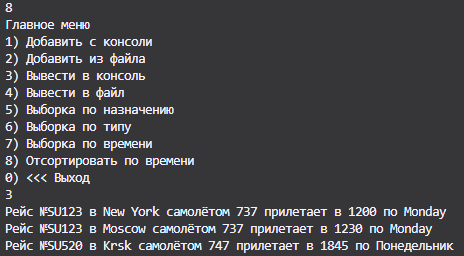
Содержимое файла ouput.txt:



Тестирование выборок:



Тестирование сортировки рейсов по полю времени:



# ОТВЕТЫ НА КОНТРОЛЬНЫЕ ВОПРОСЫ

1. **В чем состоит сущность методологии объектно-ориентированного программирования?**
   * Методология объектно-ориентированного программирования (ООП) основывается на концепции объектов, которые объединяют данные и методы для их обработки. Основные принципы включают инкапсуляцию, наследование и полиморфизм для упрощения проектирования и повышения переиспользования кода.
2. **Что такое класс? Как выполнить описание класса?**
   * Класс - это шаблон или чертёж для создания объектов. Он определяет атрибуты (поля) и методы, которые будут у объектов этого класса. Описание класса включает в себя ключевое слово **class**, за которым следует имя класса и блок, содержащий его атрибуты и методы.
3. **Что такое экземпляр класса и как его создать в программе?**
   * Экземпляр класса - это конкретный объект, созданный на основе определения класса. Для создания экземпляра используется вызов конструктора класса, например: **my\_instance = MyClass()**.
4. **Что такое поле класса?**
   * Поле класса - это переменная, которая хранит данные для каждого экземпляра класса. Они определяются внутри класса и представляют характеристики объекта.
5. **Что такое методы класса?**
   * Методы класса - это функции, связанные с классом, которые могут выполнять операции с данными класса. Они определяются внутри класса и могут вызываться для экземпляров класса.
6. **Как выполняется вызов метода класса?**
   * Вызов метода класса осуществляется через экземпляр класса, используя точечную нотацию: **instance.method()**.
7. **Какие методы называются конструкторами?**
   * Конструкторы - это методы, которые выполняют инициализацию объекта. В Python, конструктор имеет имя **\_\_init\_\_**. Он вызывается автоматически при создании экземпляра класса.
8. **Как выполняется инициализация экземпляров класса?**
   * Инициализация экземпляра класса происходит в конструкторе класса (**\_\_init\_\_**), где устанавливаются начальные значения атрибутов объекта.
9. **Как в классе реализуются инкапсуляция?**
   * Инкапсуляция в классе реализуется с помощью использования двойных подчеркиваний перед именами атрибутов (например, **self.\_\_attribute**). Это делает атрибуты приватными и ограничивает доступ к ним извне класса.

# ВЫВОД

Работа с примером кода по созданию классов Aeroflot и Flights позволяет лучше понять применение концепций ООП, включая создание объектов, работу с методами, инициализацию и использование конструкторов, а также сортировку данных в объектах класса.