**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.

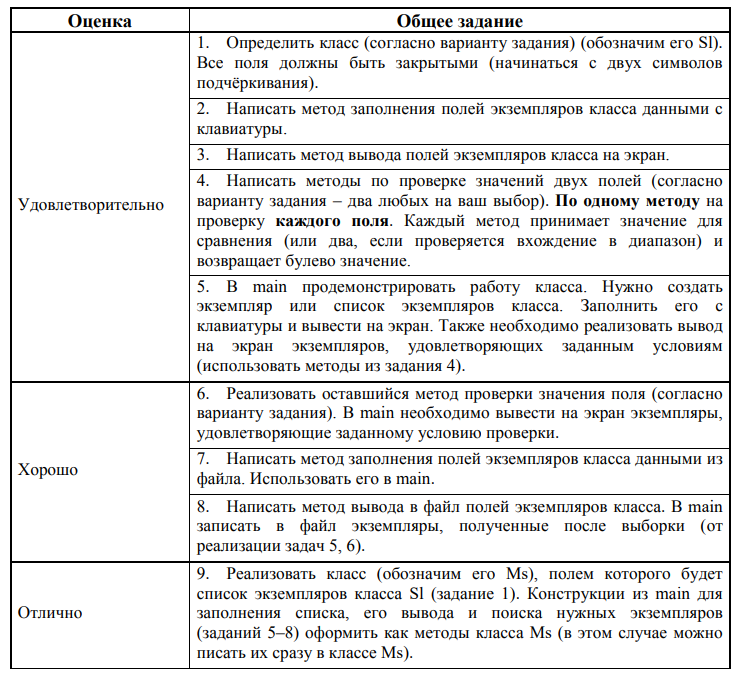
# Задания

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 module.py:

*#Чтение*

*def* read\_file(*file\_path*):

    with open(file\_path, "r") as file:

        elements = file.read().split()

        return [*int*(element) if element.isdigit() else *float*(element) for element in elements]

Чтение происходит при помощи метода open(), затем они записываются в массив elements

*#Обработка*

*def* process\_list(*original\_list*):

    nZ = 1

    aboveAvg = 0

    total\_sum = 0

    for element in original\_list:

        total\_sum += element

        if element != 0:

            nZ \*= element

        if element > sum(original\_list) / len(original\_list):

            aboveAvg += 1

    return (nZ, aboveAvg)

Обработка списка original\_list с помощью цикла, в котором, проверяется каждый элемент на то, является ли он больше среднего и умножается к общему произведению не нулевых чисел.

*#Запись*

*def* save\_result(*result*, *output\_file\_path*):

    with open(output\_file\_path, "w") as file:

        file.write(*f*"Произведение ненулевых элементов: {result[0]}\n")

        file.write(*f*"Количество элементов, превышающих среднее: {result[1]}")

Запись в файл с помощью метода write().

# Тестирование

**Пример работы программы с исходным файлом, содержащим числа 1 4 9:**

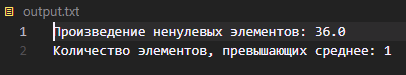
Содержимое файла input.txt:



Результат работы программы:



Содержимое файла output.txt:



**Пример работы программы с исходным файлом, содержащим числа 4 -1 6 -9 -2 -50 23 -3 8:**

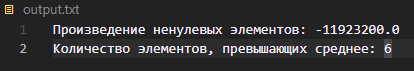
Содержимое файла input.txt:



Результат работы программы:



Содержимое файла output.txt:



Произведение отрицательное т.к. в последовательности нечетное количество отрицательных элементов.

# ОТВЕТЫ НА КОНТРОЛЬНЫЕ ВОПРОСЫ

1. **Какие типы файлов существуют? В чем их особенность?**

Существует два основных типа файлов: текстовые и бинарные. Текстовые файлы содержат данные, представленные в виде текста, читаемого человеком. Бинарные файлы содержат данные в необработанном бинарном формате и предназначены для хранения и передачи неструктурированных или бинарных данных, таких как изображения, звуковые файлы и т. д.

1. **Какая последовательность действий при работе с файлом?**
   1. Открытие файла.
   2. Выполнение операций (чтение, запись, обработка данных).
   3. Закрытие файла.
2. **Какие параметры необходимо указать при открытии файла?**

При открытии файла необходимо указать его имя (или путь к файлу) и режим доступа. Режим доступа может включать в себя параметры для чтения, записи, добавления и других операций.

1. **В чем разница между относительными и абсолютными путями к файлу?**

Абсолютный путь указывает полное расположение файла в файловой системе, начиная с корневого каталога. Относительный путь указывает путь относительно текущего рабочего каталога.

1. **Какие существуют режимы доступа для открытия файла?**

Режимы доступа включают:

* 1. **'r'**: чтение (по умолчанию),
  2. **'w'**: запись (создает новый файл или обрезает существующий),
  3. **'a'**: добавление (добавляет данные в конец файла),
  4. **'b'**: бинарный режим (для работы с бинарными файлами),
  5. **'x'**: создание (выбрасывает ошибку, если файл уже существует).

1. **Какие методы существуют для чтения данных и в чем разница между ними?**

Основные методы чтения данных из файла:

* 1. **read(size)**: читает указанное количество байтов или все до конца файла.
  2. **readline()**: читает одну строку из файла.
  3. **readlines()**: читает все строки из файла в список.

1. **В чем преимущество использования менеджера контекста?**

Менеджер контекста (конструкция **with**) автоматически управляет открытием и закрытием файла, что гарантирует правильное освобождение ресурсов даже в случае возникновения исключений.

1. **В чем отличия работы с текстовыми и бинарными файлами?**

Текстовые файлы предназначены для хранения текстовой информации и могут автоматически выполнять преобразования символов при чтении и записи. Бинарные файлы предназначены для хранения данных в необработанном виде и читаются/записываются в байтах без изменений.

1. **Как добавить и считать данные из бинарного файла?**

Для добавления данных в бинарный файл используется режим **'ab'** при открытии файла. Для чтения данных из бинарного файла используется метод **read()** для чтения байтов или **struct** модуль для разбора структурированных данных.

# ВЫВОД

В ходе выполнения лабораторной работы были успешно изучены основы работы с файлами в языке программирования Python. Были освоены методы открытия файлов, чтения и записи данных, а также использование менеджера контекста (**with**), что обеспечивает корректное управление ресурсами и закрытие файлов. Основные понятия, такие как относительные и абсолютные пути, режимы доступа, типы файлов (текстовые и бинарные), были изучены и успешно применены в процессе работы с файлами.