**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

|  |
| --- |
| Обработка списков |

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

Освоить списки языка Python, а также основные алгоритмы и языковые средства их обработки.

# порядок выполнения работы

1. Изучите теоретические сведения к данной лабораторной работе.

2. Ознакомиться с постановкой задачи (вариант задания соответствует вашему номеру в списке группы).

3. Составить программу и наборы тестовых исходных данных. Отладить программу

на тестовых данных.

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

- титульный лист;

- цель лабораторной работы;

- постановку задачи;

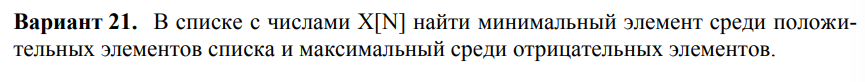
- текст программы с комментариями;

- результаты работы программы на тестовых исходных данных;

- краткие ответы на контрольные вопросы;

- выводы по лабораторной работе.

5. Защитить лабораторную работу перед преподавателем



# ХОД РАБОТЫ

**Текст программы:**

Содержимое файла minmax.py:

*def* minmax(*X*):

    minP = 10000000000

    maxO = -10000000000

    for i in X:

        if i > 0:

            minP = min(minP,i)

        elif i < 0:

            maxO = max(maxO,i)

    return (minP,maxO)

Если элемент массива X больше нуля, то он сравнивается с переменной minP, если меньше, то с переменной maxO. Возвращаются минимальный отрицательный и максимальный положительный элементы кортежем.

Содержимое файла main.py:

from minmax import \*

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

    X = *list*(map(*float*,input().split()))

    minP,maxO = minmax(X)

    print("Минимальный элемент среди положительных:", minP)

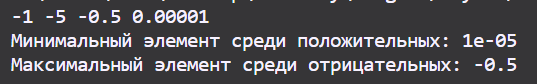
    print("Максимальный элемент среди отрицательных:", maxO)

В функцию minmax() отправляется введённый пользователем массив X,

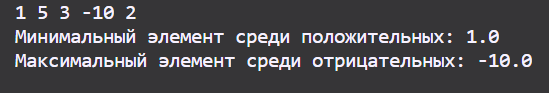
кортеж присваивается переменным minP и maxO, которые затем выводятся на экран.

# Тестирование

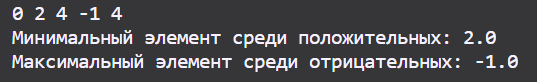
Пример работы программы с исходным массивом -1 -5 -0.5 0.00001:



Пример работы программы с исходным массивом 1 5 3 -10 2:



Пример работы программы с исходным массивом 0 2 4 -1 4:



На этом примере вводных данных хорошо заметно, что программа игнорирует 0 значения, т.к. и они не положительные и не отрицательные.

# ОТВЕТЫ НА КОНТРОЛЬНЫЕ ВОПРОСЫ

1. **Дайте определение кортежа.**

Кортеж - неизменяемая упорядоченная последовательность элементов, заключенных в круглые скобки.

1. **Дайте определение списка.**

Список - изменяемая упорядоченная последовательность элементов, заключенных в квадратные скобки.

1. **Чем отличается список от кортежа?**

Список может быть изменен (добавлены, удалены элементы), в то время как кортеж остается неизменным.

1. **Какими способами можно создать список?**

С использованием квадратных скобок: **my\_list = [1, 2, 3]** или функции **list()**: **my\_list = list(range(5))**.

1. **Что такое индекс элемента списка?**

Индекс — это порядковый номер элемента в списке, начиная с 0.

1. **Как получить доступ к значению элемента списка, зная его индекс?**

Используя квадратные скобки и индекс: **value = my\_list[2]** вернет значение элемента с индексом 2.

1. **Что такое срез? Какие параметры можно задать? Какой синтаксис** **использования среза?**

Срез — это подпоследовательность элементов списка. Синтаксис: **my\_list[start:stop:step]**. Параметры могут быть опущены.

1. **Как найти длину списка?**

С использованием функции **len()**: **length = len(my\_list)**.

1. **Как объединить списки?**

С использованием оператора **+**: **combined\_list = list1 + list2**.

1. **Как добавить в список новый элемент?**

С использованием метода **append()**: **my\_list.append(4)**.

1. **Как удалить существующий элемент?**

С использованием метода **remove()** или оператора **del**: **my\_list.remove(3)** или **del my\_list[2]**.

1. **Какие есть способы перебора элементов списка?**

С использованием цикла **for**: **for item in my\_list:**.

1. **Как отсортировать список?**

С использованием метода **sort()**: **my\_list.sort()**.

1. **Что такое метод? Как его использовать (синтаксис)?**

Метод — это функция, привязанная к объекту. Используется через точку: **object.method()**.

1. **Как узнать, входит ли элемент в список?**

С использованием оператора **in**:

if 3 in my\_list:

*#Действие*

1. **Какие основные методы списков?**

append(), extend(), remove(), pop(), index(), count(), sort(), reverse().

1. **Что такое списковое включение? Как его использовать?**

Списковое включение - это краткий способ создания списка. Пример:

squared = [x\*\*2 for x in range(5)].

1. **Что такое многомерный список? В чем особенности работы с ним?**

Многомерный список - это список, элементами которого могут быть другие списки. Работа с ним включает использование вложенных индексов.

1. **Способы копирования списков и различия в них.**

Поверхностное копирование с использованием **copy()** или среза **my\_copy = my\_list[:]**. Глубокое копирование с модулем **copy** или **copy.deepcopy()**.

1. **Напишите фрагменты программ для решения следующих задач:**

вычисление суммы положительных элементов в заданном одномерном списке:

positive\_sum = sum(x for x in my\_list if x > 0)

поиск первого максимального элемента и его индекса в заданном одномерном списке:

max\_value = max(my\_list)

max\_index = my\_list.index(max\_value)

поиск последнего минимального элемента и его индекса в заданном одномерном списке:

min\_value = min(my\_list)

min\_index = len(my\_list) - my\_list[::-1].index(min\_value) - 1

# ВЫВОД

Был разработан алгоритм, который рекурсивно вычисляет значения последовательности и осуществляет сравнение между последовательными элементами для обнаружения первого элемента, удовлетворяющего заданному условию. Эта работа позволила изучить и применить концепцию рекурсии и алгоритмов поиска элементов в последовательностях, что необходимо для решения задач анализа данных и оптимизации.