

Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт цифрового развития
Кафедра инфокоммуникаций

ОТЧЕТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ №7
дисциплины «Программирование на Python»

Выполнил:
Горбунов Данила Евгеньевич
2 курс, группа ИВТ-б-о-22-1,
09.03.01 «Информатика и
вычислительная техника»,
направленность (профиль)
«Программное обеспечение средств
вычислительной техники и
автоматизированных систем», очная
форма обучения

(подпись)

Руководитель практики:
Воронкин Р. А., канд. технических
наук, доцент кафедры
инфокоммуникаций

(подпись)

Отчет защищен с оценкой _____ Дата защиты _____

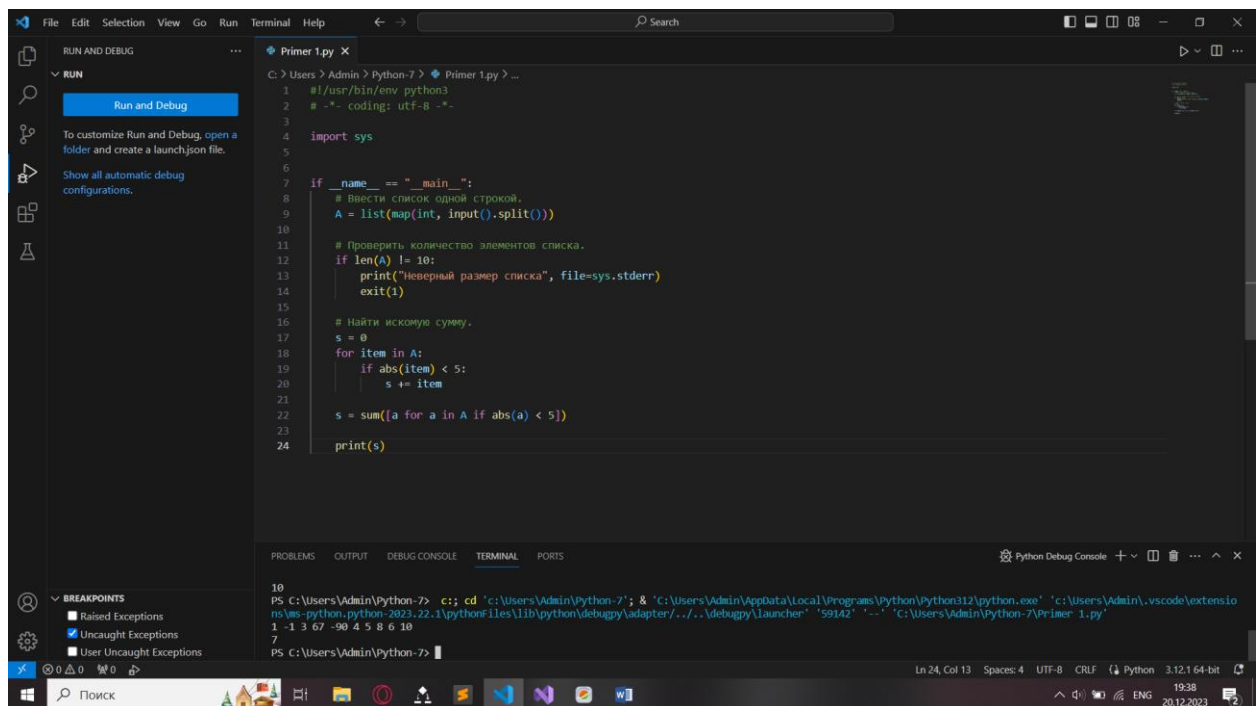
Ставрополь, 2023 г.

Тема: Работа со списками в языке Python

Цель: приобретение навыков по работе со списками при написании программ с помощью языка программирования Python версии 3.x.

Ход работы

1. Создал общедоступный репозиторий на GitHub, в котором использована лицензия MIT и язык программирования Python. Выполнил клонирование созданного репозитория.
2. Дополнил файл .gitignore необходимыми правилами.
3. Организовал созданный репозиторий в соответствии с моделью ветвления git-flow.
4. Проработал примеры лабораторной работы. Создал для каждого примера отдельный модуль языка Python. Привел в отчете скриншоты результатов выполнения каждой из программ примеров при различных исходных данных вводимых с клавиатуры.



The screenshot shows the Visual Studio Code interface. The main editor window displays a Python file named 'Primer 1.py'. The code is as follows:

```
1 #!/usr/bin/env python3
2 #-*- coding: utf-8 -*-
3
4 import sys
5
6
7 if __name__ == "__main__":
8     # Ввести список одной строкой.
9     A = list(map(int, input().split()))
10
11     # Проверить количество элементов списка.
12     if len(A) != 10:
13         print("Неверный размер списка", file=sys.stderr)
14         exit(1)
15
16     # Найти искомую сумму.
17     s = 0
18     for item in A:
19         if abs(item) < 5:
20             s += item
21
22     s = sum([a for a in A if abs(a) < 5])
23
24     print(s)
```

The bottom panel shows the 'TERMINAL' tab with the following output:

```
PS C:\Users\Admin\Python-7> c:\cd 'c:\Users\Admin\Python-7'; & 'c:\Users\Admin\AppData\Local\Programs\Python\Python312\python.exe' 'c:\Users\Admin\.vscode\extensions\ms-python.python-2023.22.1\pythonFiles\lib\python\debugpy\adapter\..\..\debugpy\launcher' '59142' '-.' 'c:\Users\Admin\Python-7\Primer_1.py'
1 -1 3 67 -90 4 5 8 6 10
7
PS C:\Users\Admin\Python-7>
```

The status bar at the bottom indicates the file is at line 24, column 13, using UTF-8 encoding, CRLF line endings, and is a Python 3.12.1 64-bit file.

Рисунок 1. Результат работы программы из примера 1

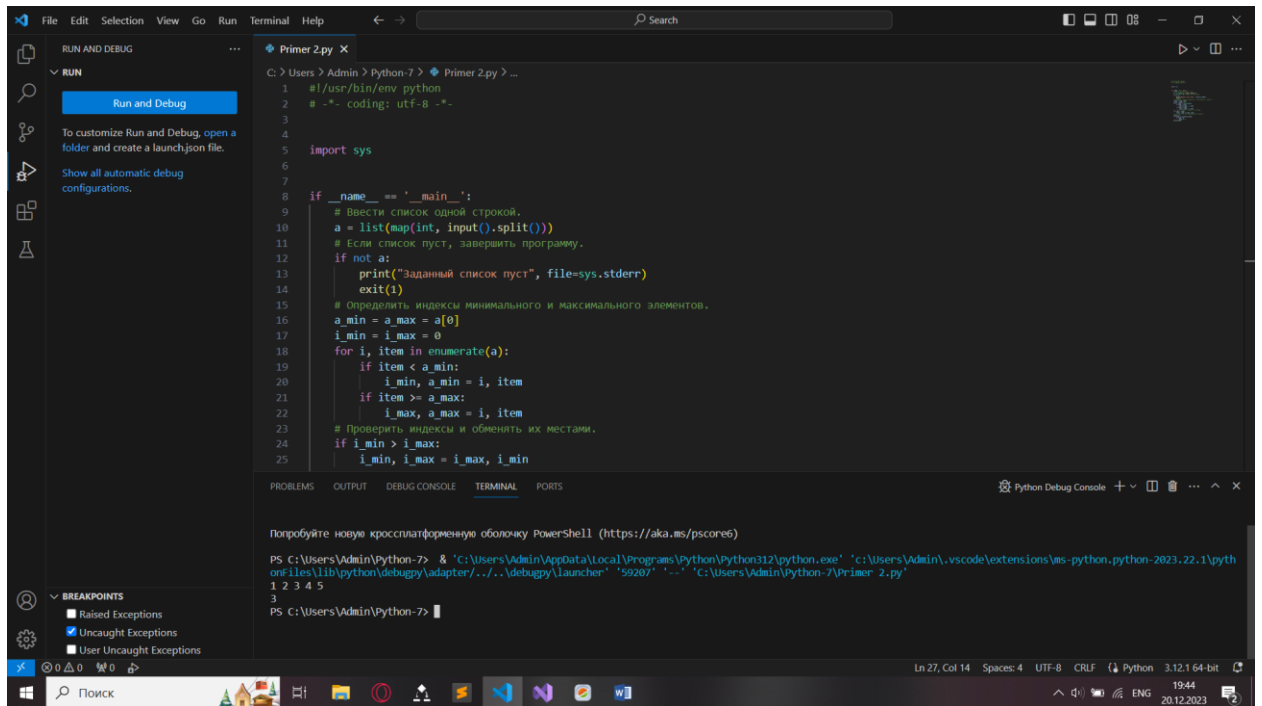


Рисунок 2. Результат работы программы из примера 2

5. Выполнил индивидуальные задания, согласно варианту 7. Привёл в отчете скриншоты работы программ.

Задание 1. Ввести список А из 10 элементов, найти произведение отрицательных элементов и вывести его на экран.

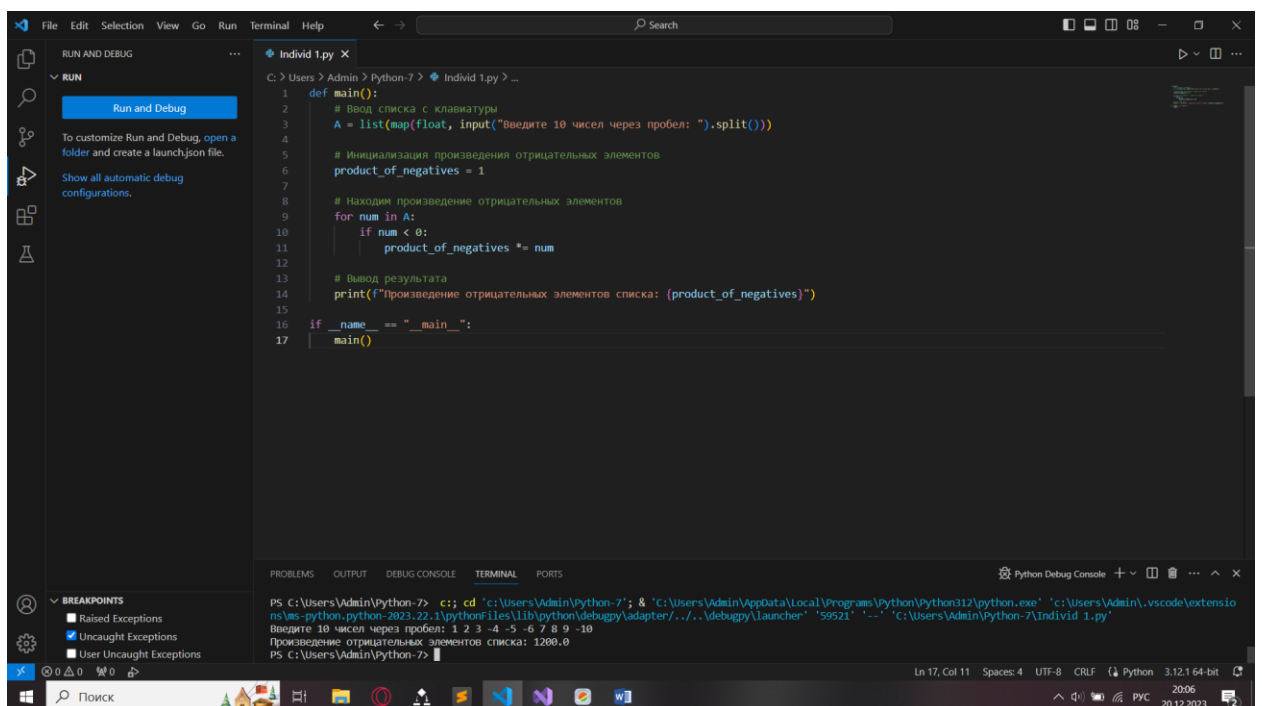


Рисунок 3. Результат работы программы из 1 индивидуального задания

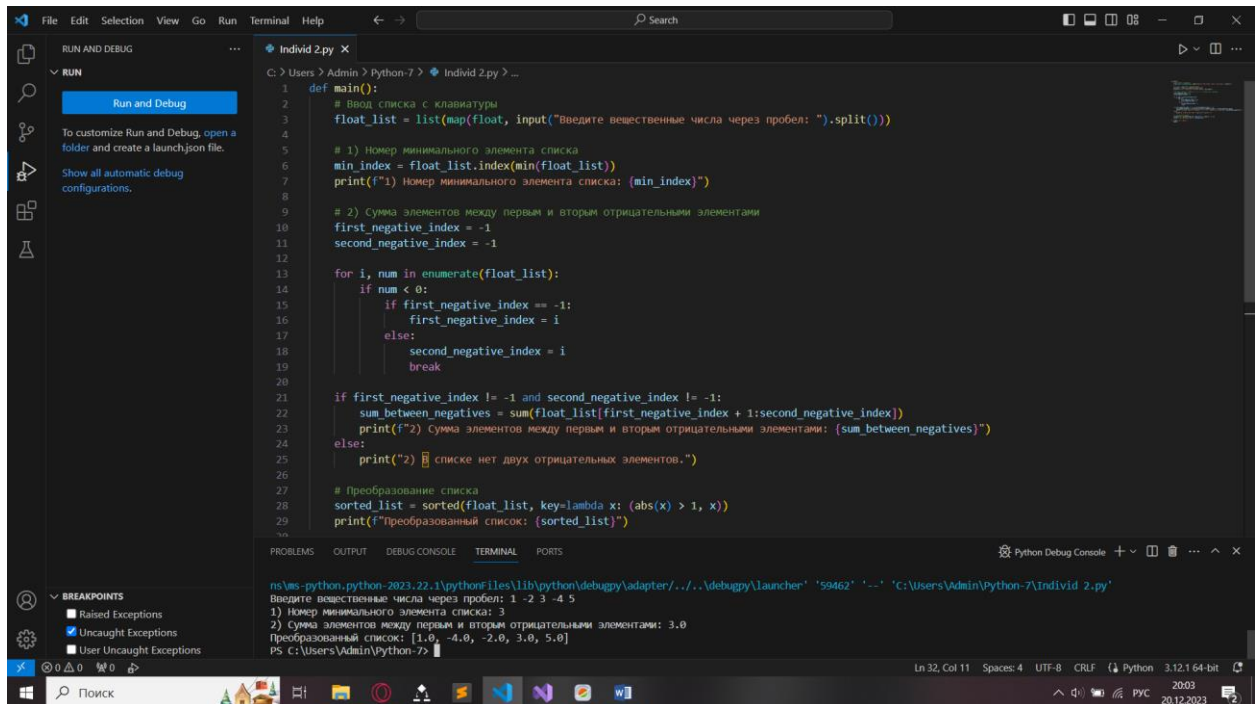
Задание 2. Составить программу с использованием одномерных массивов для решения задачи на переупорядочивание элементов массива. Для сортировки допускается использовать метод sort с

заданным параметром key и объединение нескольких списков.

В списке, состоящем из вещественных элементов, вычислить:

- 1) номер минимального элемента списка;
- 2) сумму элементов списка, расположенных между первым и вторым отрицательными элементами.

Преобразовать список таким образом, чтобы сначала располагались все элементы, модуль которых не превышает 1, а потом все остальные.



```
1 def main():
2     # Ввод списка с клавиатуры
3     float_list = list(map(float, input("Введите вещественные числа через пробел: ").split()))
4
5     # 1) Номер минимального элемента списка
6     min_index = float_list.index(min(float_list))
7     print(f"1) Номер минимального элемента списка: {min_index}")
8
9     # 2) Сумма элементов между первым и вторым отрицательными элементами
10    first_negative_index = -1
11    second_negative_index = -1
12
13    for i, num in enumerate(float_list):
14        if num < 0:
15            if first_negative_index == -1:
16                first_negative_index = i
17            else:
18                second_negative_index = i
19                break
20
21    if first_negative_index != -1 and second_negative_index != -1:
22        sum_between_negatives = sum(float_list[first_negative_index + 1:second_negative_index])
23        print(f"2) Сумма элементов между первым и вторым отрицательными элементами: {sum_between_negatives}")
24    else:
25        print("2) В списке нет двух отрицательных элементов.")
26
27    # Преобразование списка
28    sorted_list = sorted(float_list, key=lambda x: (abs(x) > 1, x))
29    print(f"Преобразованный список: {sorted_list}")
```

Python Debug Console output:

```
ns\ms-python.python-2023.22.1\python\files\lib\python\debugpy\adapter\..\..\debugpy\launcher '59462' '-' 'c:\Users\Admin\Python-7\Individ 2.py'
Введите вещественные числа через пробел: 1 -2 3 -4 5
1) Номер минимального элемента списка: 1
2) Сумма элементов между первым и вторым отрицательными элементами: 3.0
Преобразованный список: [1.0, -4.0, -2.0, 3.0, 5.0]
PS C:\Users\Admin\Python-7>
```

Рисунок 4. Результат работы программы из 2 индивидуального задания

Контрольные вопросы:

1. Что такое списки в языке Python?

Список – это структура данных для хранения объектов различных типов. Размер списка не статичен, его можно изменять. Список по своей природе является изменяемым типом данных. Переменная, определяемая как список, содержит ссылку на структуру в памяти, которая в свою очередь хранит ссылки на какие-либо другие объекты или структуры.

2. Как осуществляется создание списка в Python?

Для создания списка нужно заключить элементы в квадратные скобки: `list_1 = [1, 2, 3, 4,]`. Так же при помощи `list()`: `list_2 = list(1, 2, 3, 4)`

3. Как организовано хранение списков в оперативной памяти? При его создании в памяти резервируется область («контейнер»), в котором хранятся ссылки на другие элементы в памяти. Содержимое контейнера можно менять в отличие от чисел или строк.

4. Каким образом можно перебрать все элементы списка?

Все элементы списка можно перебрать разными способами, при помощи цикла `for` и с помощью `range(len(list))`: `for i in range(len(list)): ...` или при помощи также цикла `for` и с помощью `enumerate(a)`: `for i, item in enumerate(a):`

5. Какие существуют арифметические операции со списками?

Объединение списков при помощи оператора сложения («+») и операция повторения при помощи оператора умножения («*»).

6. Как проверить есть ли элемент в списке?

Проверить есть ли заданный элемент в списке может оператор `in`, если нет заданного элемента `not in`

7. Как определить число вхождений заданного элемента в списке?

Число вхождений заданного элемента в списке может определить метод `count`, который в качестве аргумента принимает искомый элемент.

8. Как осуществляется добавление (вставка) элемента в список?

Добавление элемента в список может осуществляется при помощи метода `append(a)`, который добавляет элемент `a` в конец списка, также можно добавить больше одного элемента методом `extend(a)`.

9. Как выполнить сортировку списка?

Отсортировать массив можно при помощи метода `sort()`, чтобы отсортировать по «убыванию», `sort(reverse=True)`.

10. Как удалить один или несколько элементов из списка?

Удалить элемент по его индексу может метод `pop(i)`.

Удалить элемент по его значению может метод `remove()`.

Оператор `del` может удалять также как метод `remove`, но сразу несколько элементов: `del list[1:3]`.

11. Что такое списковое включение и как с его помощью осуществлять обработку списков?

Списковое включение – способ построения списков, пример: `a = [i for i in range(10)]`. В этом примере создастся массив из 10 элементов: от 0 до 9. Также при помощи данного способа можно осуществлять обработку списков: с элементом `i` проводить арифметические операции, и после цикла писать условие вхождения в список.

12. Как осуществляется доступ к элементам списков с помощью срезов?

С помощью срезов доступ к элементам списков осуществляется так:

- 1) `list[:]` – копия списка;
- 2) `list[0:n]` – первые `n` элементы;
- 3) `list[n:m]` – получить элементы с `n+1` по `m`;
- 4) `list[::n]` – взять элементы списка с шагом `n`;
- 5) `list[n:m:s]` – взять элементы с `n+1` по `m` с шагом `s`.

13. Какие существуют функции агрегации для работы со списками?

Функции агрегации: получить число элементов: `len()`, получить минимальный элемент списка: `min()`, получить максимальный элемент списка: `max()`, получить сумму элементов списка: `sum()`.

14. Как создать копию списка?

Создать копию списка можно при помощи среза `a = b[:]` и при помощи метода `copy()`.

15. Самостоятельно изучите функцию `sorted` языка Python. В чем ее отличие от метода `sort` списков?

Функция `sorted()` возвращает новый отсортированный список, оставляя исходный список неизменным. Она принимает список (или другую итерируемую последовательность) в качестве аргумента и возвращает новый список, содержащий отсортированные элементы. Основное отличие между `sorted()` и `sort()` заключается в том, что `sorted()` возвращает новый отсортированный список, оставляя исходный список неизменным, в то время как `sort()` изменяет сам список, сортируя его элементы на месте.

Вывод: в ходе выполнения лабораторной работы были приобретены навыки по работе со списками при написании программ с помощью языка программирования Python версии 3.x.