

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 №2.4

**Дисциплины «Основы кросс платформенного
программирования»**

ц

Выполнил:
Якупов Эльдар Алмазович
1 курс, группа ИТС-б-о-21-1,
11.03.02 «Инфокоммуникационные
технологии и системы связи»,
направленность (профиль)
«Инфокоммуникационные системы
сети», очная форма обучения

(подпись)

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

(подпись)

Отчет защищен с оценкой _____ Дата защиты _____

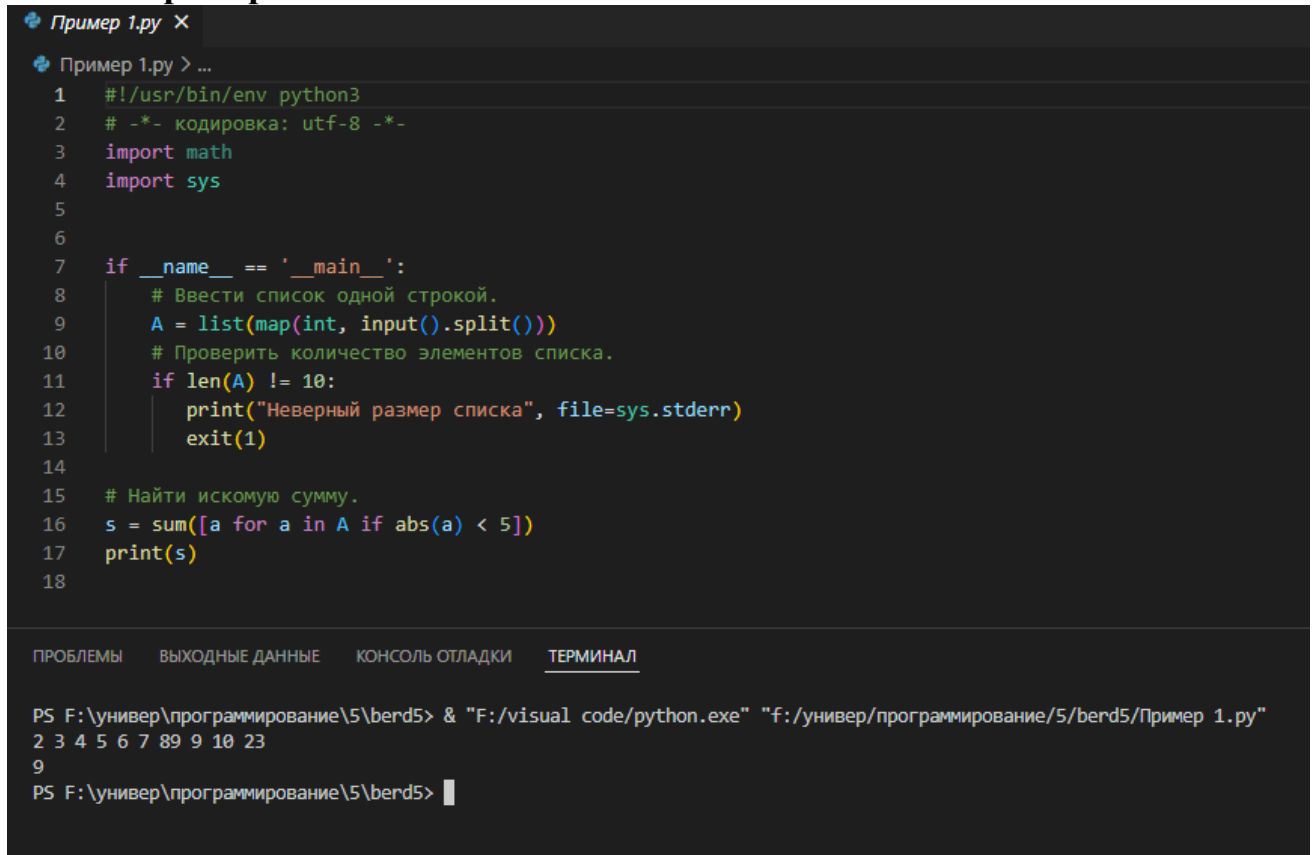
Ставрополь, 2022 г.

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

Ход работы:

Создал новый репозиторий <https://github.com/Blekroyt/berd5.git> и начал отработку примеров

Пример 1:



The image shows a code editor window titled 'Пример 1.py' containing a Python script. The script prompts the user to enter a list of numbers. Below the code editor is a terminal window showing the command to run the script and the resulting output.

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

ПРОБЛЕМЫ ВЫХОДНЫЕ ДАННЫЕ КОНСОЛЬ ОТЛАДКИ ТЕРМИНАЛ

```
PS F:\универ\программирование\5\berd5> & "F:/visual code/python.exe" "f:/универ/программирование/5/berd5/Пример 1.py"
2 3 4 5 6 7 89 9 10 23
9
PS F:\универ\программирование\5\berd5>
```

Рисунок 1. Работа программы «Пример 1»

Пример 2

```
Пример 2.py X
Пример 2.py > ...
1  #!/usr/bin/env python3
2  # -*- кодировка: utf-8 -*-
3  import math
4  import sys
5
6
7  if __name__ == '__main__':
8      # Ввести список одной строкой.
9      a = list(map(int, input().split()))
10     # Если список пуст, завершить программу.
11     if not a:
12         print("Заданный список пуст", file=sys.stderr)
13         exit(1)
14
15     # Определить индексы минимального и максимального элементов.
16     a_min = a_max = a[0]
17     i_min = i_max = 0
18     for i, item in enumerate(a):
19         if item < a_min:
```

ПРОБЛЕМЫ ВЫХОДНЫЕ ДАННЫЕ КОНСОЛЬ ОТЛАДКИ ТЕРМИНАЛ

PS F:\универ\программирование\5\berd5> & "F:/visual code/python.exe" "f:/универ/программирование/5/berd5/Пример 2.py"

21 23123 34 241 123 32 42 3421 1231 232

0

PS F:\универ\программирование\5\berd5> & "F:/visual code/python.exe" "f:/универ/программирование/5/berd5/Пример 2.py"

12 23 423 212 343 21 341 34 2 21 23

5

PS F:\универ\программирование\5\berd5> █

Рисунок 2. Работа программы «Пример 2»

Задание 1:

```
Задание 1.py X
Задание 1.py > ...
6
7  if __name__ == '__main__':
8      # Ввести список одной строкой.
9      A = list(map(int, input().split()))
10     F = []
11     R = []
12     # Если список пуст, завершить программу.
13     if not A:
14         print("Заданный список пуст", file=sys.stderr)
15         exit(1)
16     for x in A:
17         if 8 < x and x < 18:
18             F.append(x)
19             t=0
20             m =10
21     for i in F:
22         if i % 10 == 0:
23             t +=1
24     R.append(i)
```

ПРОБЛЕМЫ ВЫХОДНЫЕ ДАННЫЕ КОНСОЛЬ ОТЛАДКИ ТЕРМИНАЛ

PS F:\универ\программирование\5\berd5> & "F:/visual code/python.exe" "f:/универ/программирование/5/berd5/Задание 1.py"

2 4 6 7 9 10 12 14 16 17

[17]

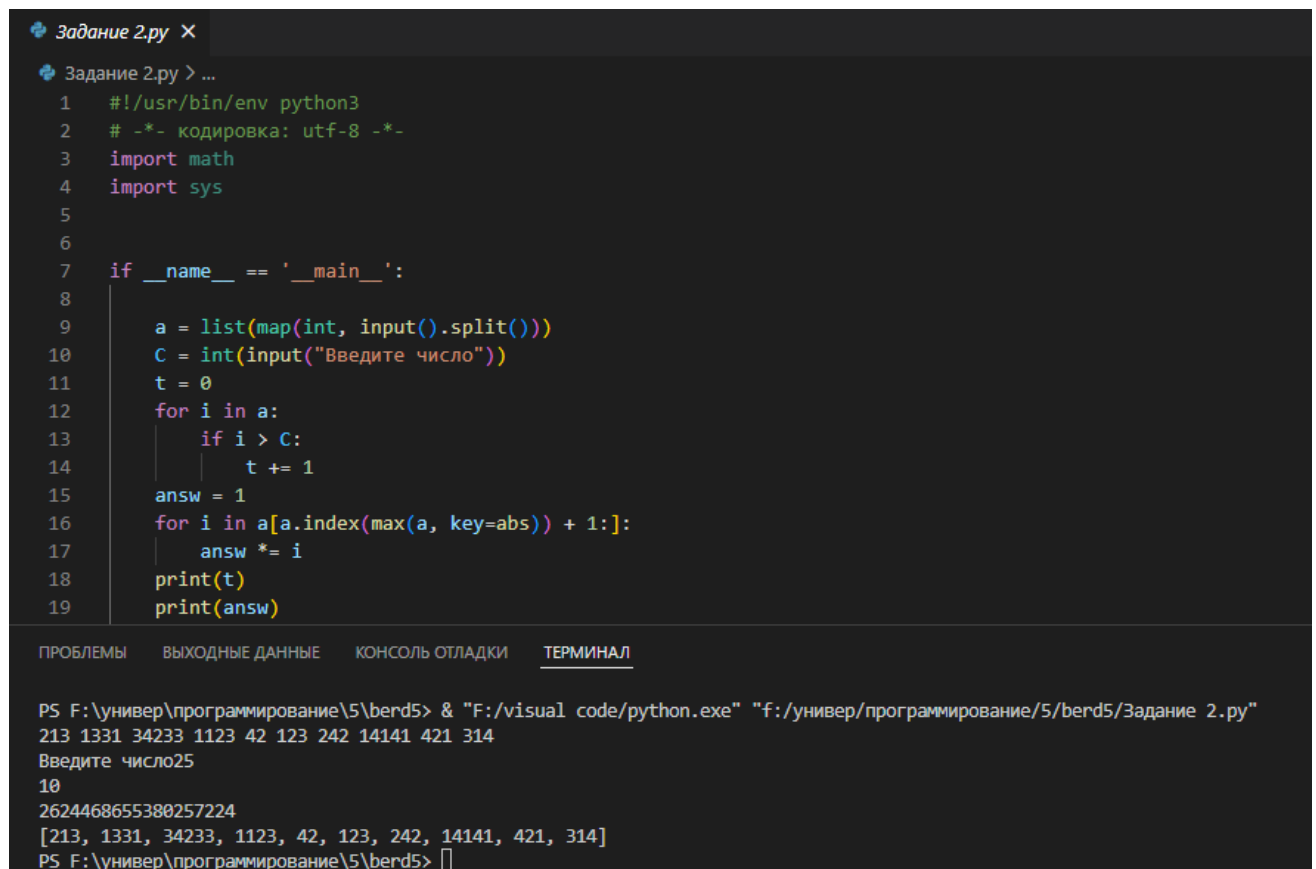
1

10

PS F:\универ\программирование\5\berd5> █

Рисунок 3. Работа программы «Задание 1»

Задание 2:



```
Задание 2.py > ...
1  #!/usr/bin/env python3
2  # -*- кодировка: utf-8 -*-
3  import math
4  import sys
5
6
7  if __name__ == '__main__':
8
9      a = list(map(int, input().split()))
10     c = int(input("Введите число"))
11     t = 0
12     for i in a:
13         if i > c:
14             t += 1
15     answ = 1
16     for i in a[a.index(max(a, key=abs)) + 1:]:
17         answ *= i
18     print(t)
19     print(answ)

ПРОБЛЕМЫ  ВЫХОДНЫЕ ДАННЫЕ  КОНСОЛЬ ОТЛАДКИ  ТЕРМИНАЛ

PS F:\универ\программирование\5\berd5> & "F:/visual code/python.exe" "f:/универ/программирование/5/berd5/Задание 2.py"
213 1331 34233 1123 42 123 242 14141 421 314
Введите число25
10
2624468655380257224
[213, 1331, 34233, 1123, 42, 123, 242, 14141, 421, 314]
PS F:\универ\программирование\5\berd5> 
```

Рисунок 4. Работа программы «Задание 2»

Контрольные вопросы

1. Что такое списки в языке Python?

Список (*list*) – 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:
`my_list = [1, 2, 3, 4, 5]`

3. Как организовано хранение списков в оперативной памяти?

При создании списка в памяти резервируется область, которую можно условно назвать некоторым “контейнером”, в котором хранятся ссылки на другие элементы данных в памяти. В отличие от таких типов данных как число или строка, содержимое “контейнера” списка можно менять.

4. Каким образом можно перебрать все элементы списка? Читать

элементы списка можно с помощью следующего цикла:

```
my_list = ['один', 'два', 'три', 'четыре', 'пять']
for elem in my_list:
    print(elem)
```

5. Какие существуют арифметические операции со списками?

Для объединения списков можно использовать оператор сложения (+).

Список можно повторить с помощью оператора умножения (*):

6. Как проверить есть ли элемент в списке?

Для того, чтобы проверить, есть ли заданный элемент в списке Python необходимо использовать оператор `in` :

7. Как определить число вхождений заданного элемента в списке?

Ответ: Метод *count* можно использовать для определения числа сколько раз данный элемент встречается в списке:

8. Как осуществляется добавление (вставка) элемента в список? Метод *insert* можно использовать, чтобы вставить элемент в список. Метод *append* можно использовать для добавления элемента в список.

9. Как выполнить сортировку списка?

Для сортировки списка нужно использовать метод *sort*.

Для сортировки списка в порядке убывания необходимо вызвать метод *sort* с аргументом *reverse=True*.

10. Как удалить один или несколько элементов из списка? Удалить элемент можно, написав его индекс в методе *pop*: Если не указывать индекс, то функция удалит последний элемент. Элемент можно удалить с помощью метода *remove*.

Оператор `del` можно использовать для тех же целей:

Можно удалить несколько элементов с помощью оператора среза:

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

List Comprehensions чаще всего на русский язык переводят как абстракция списков или списковое включение, является частью синтаксиса языка, которая предоставляет простой способ построения списков.

В языке Python есть две очень мощные функции для работы с

коллекциями: `map` и `filter`. Они позволяют использовать функциональный стиль программирования, не прибегая к помощи циклов, для работы с такими типами как `list`, `tuple`, `set`, `dict` и т.п. Списковое включение позволяет обойтись без этих функций.

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

Слайсы (срезы) являются очень мощной составляющей *Python*, которая позволяет быстро и лаконично решать задачи выборки элементов из списка.

Слайс задается тройкой чисел, разделенных запятой: *start:stop:step*. *Start*— позиция с которой нужно начать выборку, *stop*— конечная позиция, *step*— шаг. При этом необходимо помнить, что выборка не включает элемент определяемый *stop*.

13. Какие существуют функции агрегации для работы со списками? Для работы со списками *Python* предоставляет следующие функции

`len(L)` - получить число элементов в списке `L` .

`min(L)` - получить минимальный элемент списка `L` .

`max(L)` - получить максимальный элемент списка `L` .

`sum(L)` - получить сумму элементов списка `L` , если список `L` содержит только числовые значения.

14. Как создать копию списка? Воспользоваться командой `copy.copy(x)`

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

Функция `sorted()` в Python возвращает отсортированный список из элементов в итерируемом объекте. `list.sort()` на 13% быстрее, чем `sorted()`.

Вывод: приобрёл навыки по работе со списками при написании программ с помощью языка программирования Python версии 3.x.