

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное
образовательное учреждение высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Кафедра инфокоммуникаций

Отчет по лабораторной работе №6
Работа со списками в языке Python
по дисциплине «Основы кроссплатформенного программирования»

Выполнил студент группы ИВТ-б-о-20-1

Пушкин Н.С. « » _____ 20__ г.

Подпись студента _____

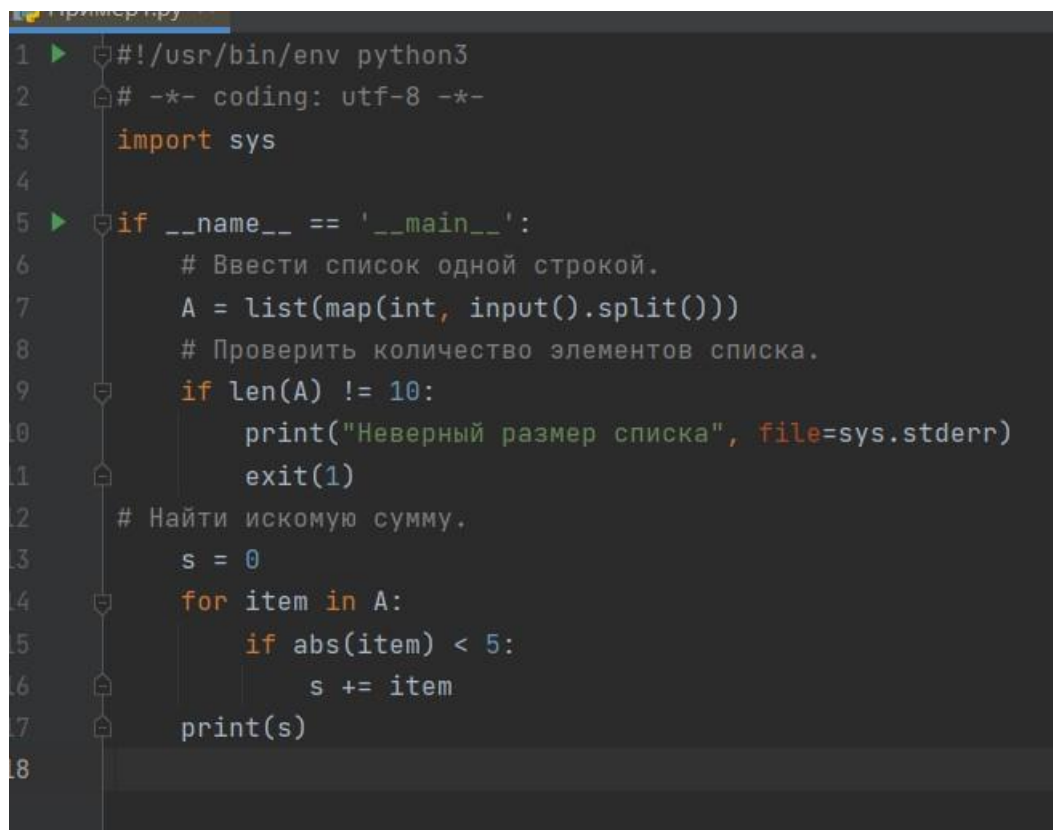
Работа защищена « » _____ 20__ г.

Проверил Воронкин Р.А. _____
(подпись)

Ставрополь 2020

Цель работы: приобретение навыков по работе со списками при написании программ с помощью языка программирования Python версии 3.x.
Ссылка на репозиторий: <https://github.com/NiKiN126>

1. Следуя методическим указаниям, создал новый репозиторий на github, после чего клонировал его и создал в папке репозитория новый проект PyCharm.
2. Проработал все примеры, указанные в методическом документе, сделав для них UML-диаграммы:



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

Рисунок 1. Пример 1

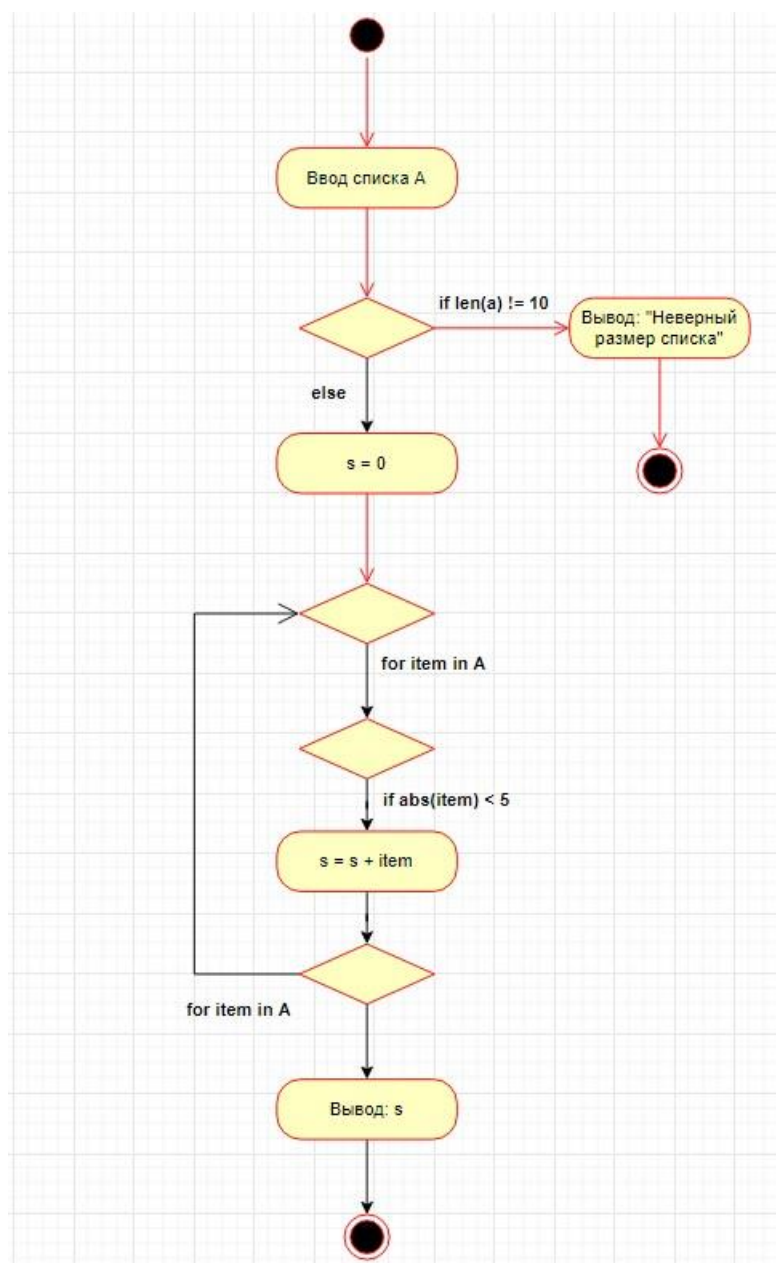


Рисунок 2. Диаграмма для первого примера

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
import sys

if __name__ == '__main__':
    # Ввести список одной строкой.
    a = list(map(int, input().split()))
    # Если список пуст, завершить программу.
    if not a:
        print("Заданный список пуст", file=sys.stderr)
        exit(1)
    # Определить индексы минимального и максимального элементов.
    a_min = a_max = a[0]
    i_min = i_max = 0
    for i, item in enumerate(a):
        if item < a_min:
            i_min, a_min = i, item
        if item >= a_max:
            i_max, a_max = i, item
    # Проверить индексы и обменять их местами.
    if i_min > i_max:
        i_min, i_max = i_max, i_min
    # Посчитать количество положительных элементов.
    count = 0
    for item in a[i_min+1:i_max]:
        if item > 0:
            count += 1
    print(count)
```

Рисунок 3. Пример 2

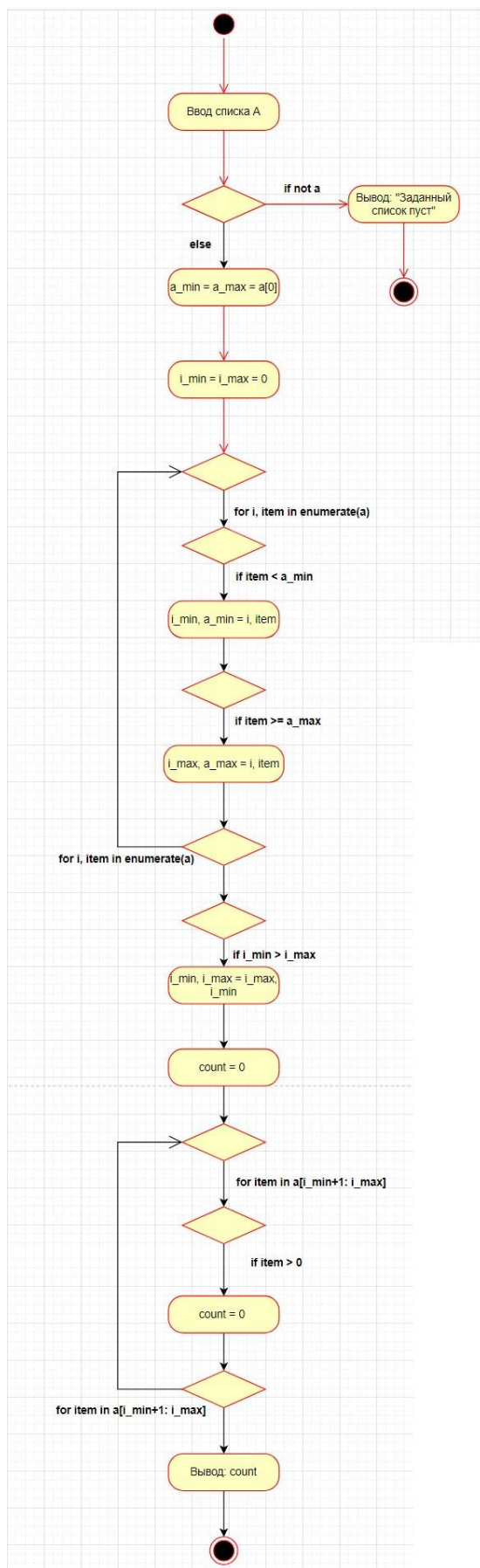


Рисунок 4. Диаграмма для второго примера

3. Затем приступил к выполнению индивидуальных заданий, также сделав для каждого UML-диаграмму:

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
if __name__ == '__main__':
    U = list(map(float, input("Введите значения U: ").split()))
    D = list(map(float, input("Введите значения D: ").split()))
    B = list(map(float, input("Введите значения B: ").split()))
    S = []
    if len(U) != 7 or len(D) != 7 or len(B) != 7:
        print("Введите значение на каждый день недели, то есть 7 значений для каждой величины")
        exit(1)
    for i in range(7):
        S.append((U[i]+D[i]+B[i])/3)
    print("Среднее значение температуры по дням недели: ", S)
```

Рисунок 5. Код первого индивидуального задания

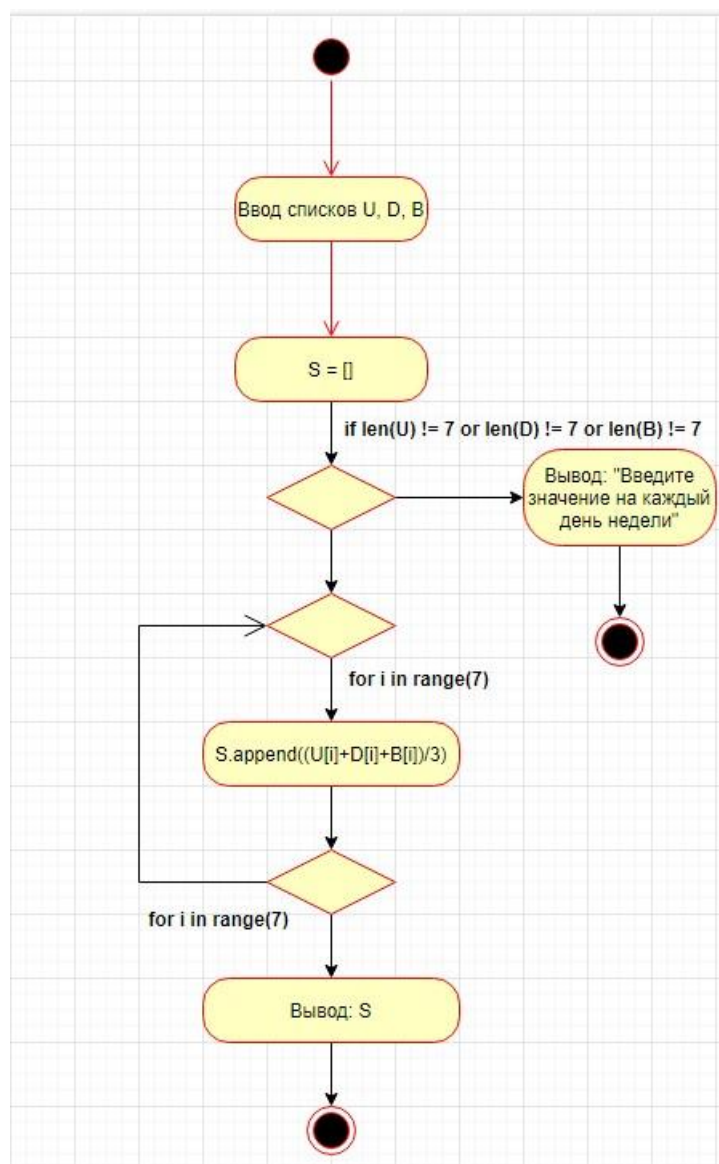


Рисунок 6. Диаграмма для первого задания

```

#!/usr/bin/env python3
# -*- coding: utf-8 -*-
import sys
import math

if __name__ == '__main__':
    a = list(map(float, input("Введите элементы списка: ").split()))
    A = int(input("Введите минимальное значение диапазона (A): "))
    B = int(input("Введите максимальное значение диапазона (B): "))
    m = 0
    n = 0
    if not a:
        print("Заданный список пуст", file=sys.stderr)
        exit(1)
    b = a[A:B]
    b = len(b)
    print("Количество чисел в диапазоне от A до B = ", b)
    for i in range(len(a)):
        if m < a[i]:
            m = a[i]
            n = i
    c = a[n:]
    s = sum(c, -m)
    c.sort(key=lambda x: math.fabs(x), reverse=True)
    print("Сумма чисел после максимального элемента, введённого списка", s)
    print(c)

```

Рисунок 7. Код второго индивидуального задания

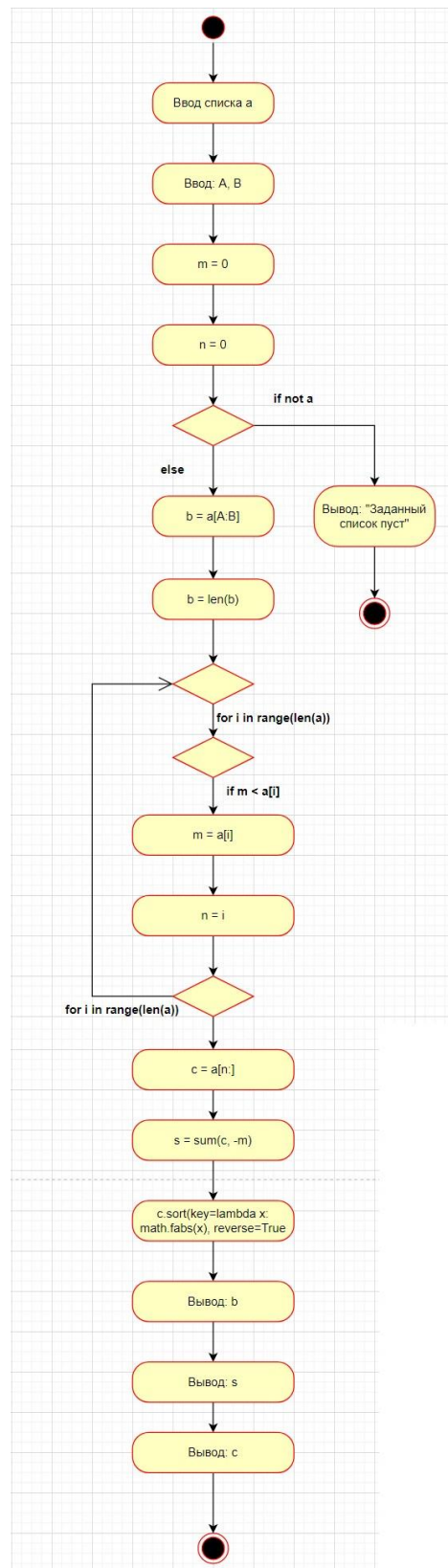


Рисунок 8. Диаграмма для второго задания Контрольные

вопросы:

1. Что такое списки в языке Python?

Список (list) – это структура данных для хранения объектов различных типов. Список очень похож на массив, только в нем можно хранить объекты 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?

Как уже было сказано выше, список является изменяемым типом данных. При его создании 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 i in my_list: print(i)
```

5. Какие существуют арифметические операции со списками?

Для объединения списков можно использовать оператор сложения (+):

```
list_1 = [1, 2, 3] list_2 = [4, 5, 6]
```

Список можно повторить с помощью оператора умножения (*): list_1 = [1, 2, 3]*2

6. Как проверить есть ли элемент в списке?

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

```
lst = [3, 5, 2, 4, 1]
```

if 3 in lst:

print("Список содержит число 3") else:

print("Список не содержит число 3")

7. Как определить число вхождений заданного элемента в списке?

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

```
lst = [1, 2, 2, 3, 3] print(lst.count(3))
```

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

Метод insert можно использовать, чтобы вставить элемент в начало списка:

```
my_list = [1, 2, 3, 4, 5] my_list.insert(1, 'Привет')
```

Метод append можно использовать для добавления элемента в конец списка.

9. Как выполнить сортировку списка?

Для сортировки списка нужно использовать метод sort или sorted.

10. Как удалить один или несколько элементов из списка?

Удалить элемент можно, написав его индекс в методе pop.

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

Списковое включение является частью синтаксиса языка, которая предоставляет простой способ построения списков:

```
>>> a = [i for i in range(int(input()))]
```

```
7
```

```
>>> print(a)
```

```
[0, 1, 2, 3, 4, 5, 6]
```

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.

С функциями:

```
>>> b = list(filter(lambda x: x % 2 == 0, a))
```

Через включения:

```
>>> b = [i for i in a if i % 2 == 0]
```

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?

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?

Для создания копии списка необходимо использовать либо метод copy, либо использовать оператор среза.

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

Для сортировки достаточно вызвать функцию сортировки Python `sorted()`, которая вернёт новый отсортированный список. Также можно использовать метод списков `list.sort()`, который изменяет исходный список. Обычно это не так удобно, как использование `sorted()`, но если не нужен исходный список, то так будет немного эффективнее. Ещё одно отличие заключается в том, что метод `list.sort()` определён только для списков, в то время как `sorted()` работает со всеми итерируемыми объектами.

Вывод: в ходе выполнения лабораторной работы, изучил основы по работе с элементами списков языка python.