

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

Кафедра инфокоммуникаций

Отчет по лабораторной работе №7
Работа с кортежами 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)

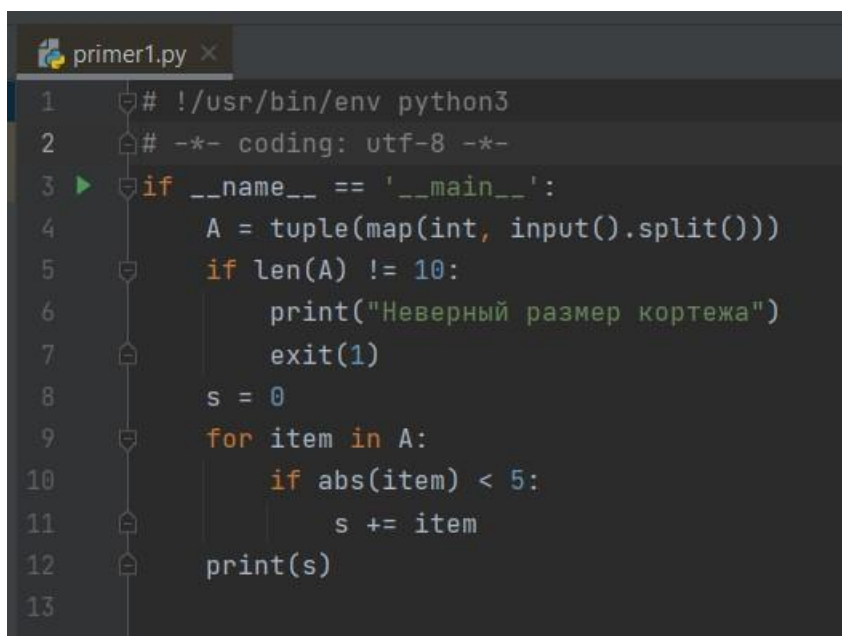
Ставрополь 2021

Цель работы: приобретение навыков по работе с кортежами 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 if __name__ == '__main__':
4     A = tuple(map(int, input().split()))
5     if len(A) != 10:
6         print("Неверный размер кортежа")
7         exit(1)
8     s = 0
9     for item in A:
10         if abs(item) < 5:
11             s += item
12     print(s)
13
```

Рисунок 1. Пример 1

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

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
if __name__ == '__main__':
    U = tuple(map(float, input("Введите значения U: ").split()))
    D = tuple(map(float, input("Введите значения D: ").split()))
    B = tuple(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):
        (k, ) = ((U[i]+D[i]+B[i])/3, )
        S = S + (k, )
    print("Среднее значение температуры по дням недели: ", S)
```

Рисунок 5. Код первого индивидуального задания

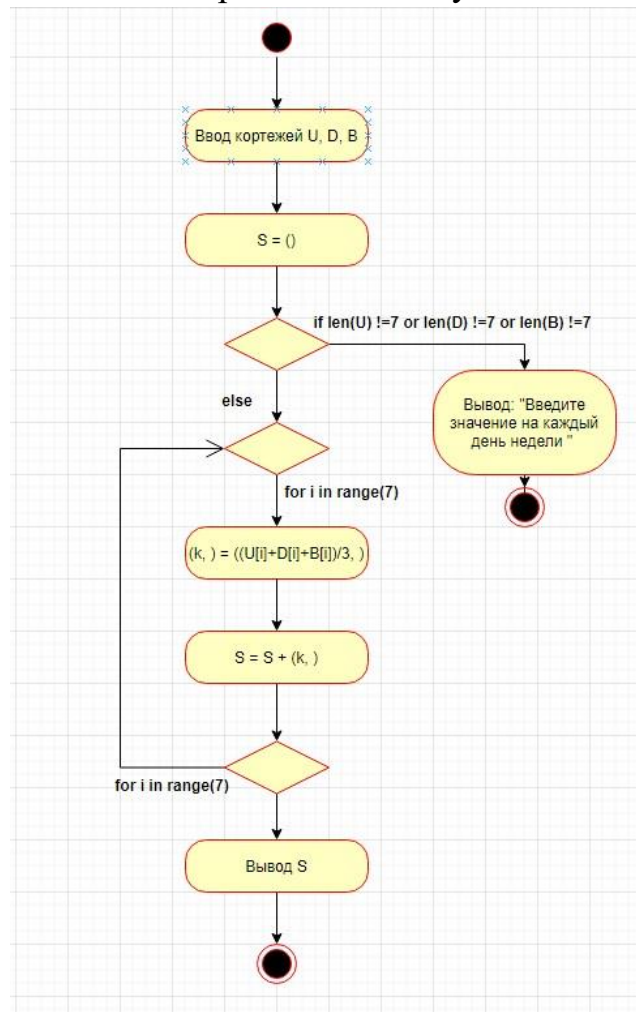


Рисунок 6. Диаграмма для первого задания

```
if __name__ == '__main__':
    a = tuple(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)
    j = tuple(sorted(c, key=lambda x: math.fabs(x), reverse=True))
    print("Сумма чисел после максимального элемента, введённого списка", s)
    print(j)
```

Рисунок 7. Код второго индивидуального задания

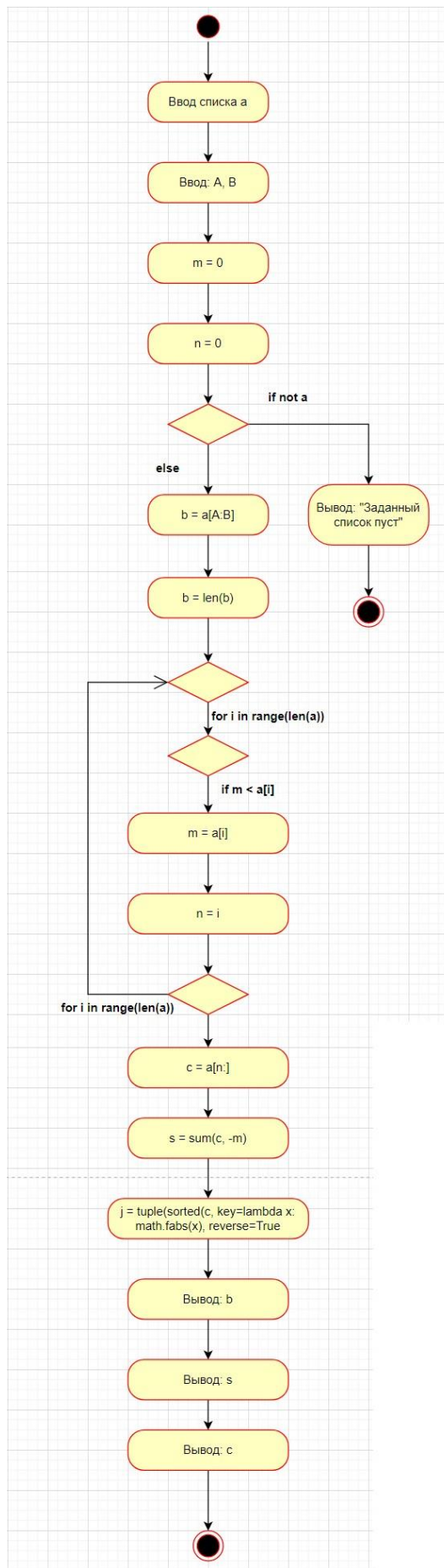


Рисунок 8. Диаграмма для второго задания Контрольные вопросы:

1. Что такое кортежи в языке Python?

Кортеж (tuple) – это неизменяемая структура данных, которая по своему подобию очень похожа на список. Список – это изменяемый тип данных. Т. е. если у нас есть список `a = [1, 2, 3]` и мы хотим заменить второй элемент с 2 на 15, то мы можем это сделать, напрямую обратившись к элементу списка. С кортежем мы не можем производить такие операции, т. к. элементы его изменять нельзя.

2. Каково назначение кортежей в языке Python?

Существует несколько причин, по которым стоит использовать кортежи вместо списков. Одна из них – это обезопасить данные от случайного изменения. Если мы получили откуда-то массив данных, и у нас есть желание поработать с ним, но при этом непосредственно менять данные мы не собираемся, тогда, это как раз тот случай, когда кортежи придется как нельзя кстати. Используя их в данной задаче, мы дополнительно получаем сразу несколько бонусов – во-первых, это экономия места. Дело в том, что кортежи в памяти занимают меньший объем по сравнению со списками. Во-вторых – прирост производительности, который связан с тем, что кортежи работают быстрее, чем списки (т. е. на операции перебора элементов и т. п. будет тратиться меньше времени). Важно также отметить, что кортежи можно использовать в качестве ключа у словаря. 3. Как осуществляется создание кортежей?

Для создания пустого кортежа можно воспользоваться одной из следующих команд.

```
>>> a = ()
```

```
>>> b = tuple()
```

Кортеж с заданным содержанием создается также как список, только вместо квадратных скобок используются круглые.

4. Как осуществляется доступ к элементам кортежа?

Доступ к элементам кортежа осуществляется также как к элементам списка – через указание индекса.

5. Зачем нужна распаковка (деструктуризация) кортежа? Обращение по индексу, это не самый удобный способ работы с кортежами. Дело в том, что кортежи часто содержат значения разных типов, и помнить, по какому индексу что лежит — очень непросто. Но есть способ лучше! Как мы кортеж собираем, так его можно и разобрать:

```
name_and_age = ('Bob', 42)
(name, age) = name_and_age
name # 'Bob' age # 42
```

Именно таким способом принято получать и сразу разбирать значения, которые возвращает функция (если таковая возвращает несколько значений, конечно).

6. Какую роль играют кортежи в множественном присваивании?

Благодаря тому, что кортежи легко собирать и разбирать, в Python удобно делать такие вещи, как множественное присваивание:

```
(a, b, c) = (1, 2, 3)
a # 1 b # 2 c # 3
```

7. Как выбрать элементы кортежа с помощью среза?

С помощью операции взятия среза можно получить другой кортеж.

Общая форма операции взятия среза для кортежа следующая

```
T2 = T1[i:j]
```

8. Как выполняется конкатенация и повторение кортежей?

Для кортежей можно выполнять операцию конкатенации, которая обозначается символом `+`. В простейшем случае для конкатенации двух кортежей общая форма операции следующая:

$$T3 = T1 + T2$$

Кортеж может быть образован путем операции повторения, обозначаемой символом `*`. При использовании в выражении общая форма операции следующая: $T2 = T1 * n$

9. Как выполняется обход элементов кортежа?

Элементы кортежа можно последовательно просмотреть с помощью операторов цикла `while` или `for`.

10. Как проверить принадлежность элемента кортежу?

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

11. Какие методы работы с кортежами Вам известны?

Метод `index()` – чтобы получить индекс (позицию) элемента в кортеже.

Метод `count()` – чтобы определить количество вхождений заданного элемента в кортеж.

12. Допустимо ли использование функций агрегации таких как `len()`, `sum()` и т. д. при работе с кортежами?

Да.

13. Как создать кортеж с помощью спискового включения

В отличие от выражения `[a for a in A ...]`, которое на выходе дает нам список, выражение `(a for a in A ...)` дает на выходе специальный объект генератора, а не кортеж. Для преобразования генератора в кортеж необходимо воспользоваться вызовом `tuple()`

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