

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 №2.5
дисциплины «Основы программной инженерии»

Выполнил:
Баратов Семен Григорьевич
2 курс, группа ПИЖ-б-о-22-1,
09.03.04 «Программная инженерия»,
направленность (профиль) «Разработка
и сопровождение программного
обеспечения», очная форма обучения

(подпись)

Преподаватель:
Воронкин Р.А., канд. тех. наук, доцент,
доцент кафедры инфокоммуникаций

(подпись)

Отчет защищен с оценкой _____ Дата защиты _____

Ставрополь, 2023 г.

Тема: Работа с кортежами в языке Python.

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

Результаты выполнения

1. Создали репозиторий с лицензией MIT, добавили в .gitignore необходимые правила для работы с IDE PyCharm, клонировали репозиторий, организовали репозиторий в соответствии с моделью git-flow.

```
Last login: Tue Oct 24 20:29:02 on ttys000
[itssyoma@MacBook-Air-Sema Основы программной инженерии % git clone https://github.com/itssyoma/megarepo_21.git
Cloning into 'megarepo_21'...
remote: Enumerating objects: 4, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 4 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (4/4), done.
[itssyoma@MacBook-Air-Sema Основы программной инженерии % cd megarepo_21
[itssyoma@MacBook-Air-Sema megarepo_21 % git checkout -b develop
Switched to a new branch 'develop'
[itssyoma@MacBook-Air-Sema megarepo_21 % git branch release
[itssyoma@MacBook-Air-Sema megarepo_21 % git branch develop
fatal: a branch named 'develop' already exists
[itssyoma@MacBook-Air-Sema megarepo_21 % git branch hotfix
[itssyoma@MacBook-Air-Sema megarepo_21 % git branch feature
[itssyoma@MacBook-Air-Sema megarepo_21 %
```

Рисунок 1 – Работа с репозиторием в командной строке.

2. Проработали пример №1.

```
example1.py > ...
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  import sys
5
6
7  if __name__ == '__main__':
8      # Ввести кортеж одной строкой.
9      A = tuple(map(int, input().split()))
10
11     # Проверить количество элементов кортежа.
12     if len(A) != 10:
13         print("Неверный размер кортежа", file=sys.stderr)
14         exit(1)
15     # Найти искомую сумму.
16
17     s=0
18     for item in A:
19         if abs(item) < 5:
20             s += item
21
22     print(s)
23
```

ПРОБЛЕМЫ ВЫХОДНЫЕ ДАННЫЕ КОНСОЛЬ ОТЛАДКИ ТЕРМИНАЛ ПО

```
/usr/local/bin/python3 "/Users/itssyoma/Yandex.Disk.localized/Учеба/
itssyoma@MacBook-Air-Sema megarepo_25 % /usr/local/bin/python3 "/Use
epo_25/example1.py
-4 -6 -2 19 6 1 2 3 4 -23
4
```

Рисунок 2 – Пример №1. Код и результат выполнения программы.

3. Выполнили индивидуальное задание (вариант 1). Известно количество очков, набранных каждой из 20 команд – участниц первенства по футболу. Перечень очков дан в порядке убывания (ни одна пара команд не набрала одинаковое количество очков). Определить, какое место заняла команда, набравшая n очков (естественно, что значение n имеется в перечне). Условный оператор не использовать.

```
task.py > ...
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  import sys
5
6
7  if __name__ == "__main__":
8
9      scores = tuple(int(i) for i in input("Введите баллы 20 команд в поорядке убывания: ").split())
10     if len(scores) < 20:
11         print("Количество команд меньше 20", file=sys.stderr)
12         exit(1)
13     for num1, num2 in zip(scores, scores[1:]):
14         if num2 > num1:
15             print("Баллы расположены не в порядке убывания", file=sys.stderr)
16             exit(1)
17
18     n = int(input("Введите количество баллов, для которого нужно подсчитать место: "))
19     if n not in scores:
20         print("Такого балла нет", file=sys.stderr)
21         exit(1)
22
23     result = {score: position for position, score in enumerate(scores, start=1)}
24
25     print("Команда с данным баллом заняла", result[n], "место")
26
```

Рисунок 3 – Задание. Код программы.

```
itssyoma@MacBook-Air-Sema megarepo_25 % /usr/local/bin/python3 "/Users/itssyoma/Yandex.Disk.localized/Y
Основы программной инженерии/megarepo_25/task.py"
Введите баллы 20 команд в поорядке убывания: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
Баллы расположены не в порядке убывания
itssyoma@MacBook-Air-Sema megarepo_25 % /usr/local/bin/python3 "/Users/itssyoma/Yandex.Disk.localized/Y
Основы программной инженерии/megarepo_25/task.py"
Введите баллы 20 команд в поорядке убывания: 1 2 3 4
Количество команд меньше 20
itssyoma@MacBook-Air-Sema megarepo_25 % /usr/local/bin/python3 "/Users/itssyoma/Yandex.Disk.localized/Y
Основы программной инженерии/megarepo_25/task.py"
Введите баллы 20 команд в поорядке убывания: 89 76 74 71 65 62 58 57 51 49 48 43 41 39 37 35 33 31 20 11
Введите количество баллов, для которого нужно подсчитать место: 74
Команда с данным баллом заняла 3 место
```

Рисунок 4 – Задание. Результат выполнения программы.

Ответы на контрольные вопросы

1. Что такое списки в языке Python?

Списки в языке Python - это упорядоченные изменяемые коллекции объектов, которые могут содержать любые типы данных

2. Каково назначение кортежей в языке Python?

Кортежи в языке Python - это упорядоченные неизменяемые коллекции объектов, которые могут содержать любые типы данных.

3. Как осуществляется создание кортежей?

Кортежи могут быть созданы с помощью круглых скобок и запятых, например: `my_tuple = (1, "hello", True)`.

4. Как осуществляется доступ к элементам кортежа?

Доступ к элементам кортежа осуществляется с помощью индексов, начиная с 0, например: `my_tuple[0]` вернет первый элемент кортежа.

5. Зачем нужна распаковка (деструктуризация) кортежа?

Распаковка (деструктуризация) кортежа позволяет присвоить каждому элементу кортежа отдельную переменную, например: `a, b, c = my_tuple`.

6. Какую роль играют кортежи в множественном присваивании?

Кортежи позволяют выполнить множественное присваивание значений переменным, например: `a, b = b, a`.

7. Как выбрать элементы кортежа с помощью среза?

Элементы кортежа можно выбирать с помощью срезов, например: `my_tuple[1:3]` вернет подкортеж, содержащий второй и третий элементы.

8. Как выполняется конкатенация и повторение кортежей?

Кортежи можно конкатенировать с помощью оператора `+` и повторять с помощью оператора `*`, например: `my_tuple1 + my_tuple2` и `my_tuple * 3`.

9. Как выполняется обход элементов кортежа?

Обход элементов кортежа можно выполнить с помощью цикла `while` или `for`, например: `for item in my_tuple: print(item)`.

10. Как проверить принадлежность элемента кортежу?

Чтобы проверить принадлежность элемента кортежу, можно использовать оператор `in`, например: `if "hello" in my_tuple: print("Found")`.

11. Какие методы работы с кортежами Вам известны?

Некоторые методы работы с кортежами: `count()` - возвращает количество элементов с заданным значением, `index()` - возвращает индекс первого элемента с заданным значением.

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

Да, функции агрегации такие как `len()`, `sum()` и т. д. могут быть использованы при работе с кортежами.

13. Как создать кортеж с помощью спискового включения.

Кортеж можно создать с помощью спискового включения, например:
`my_tuple = tuple([x**2 for x in range(5)])`.