

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ
ФЕДЕРАЦИИ
Федеральное государственное автономное
образовательное учреждение высшего
образования
«СЕВЕРОКАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ
УНИВЕРСИТЕТ»**

**Кафедра
инфокоммуникаций
Институт цифрового
развития**

ОТЧЁТ
по лабораторной работе №7
Дисциплина: «Основы программной инженерии»
Тема: «Работа со списками в языке Python»

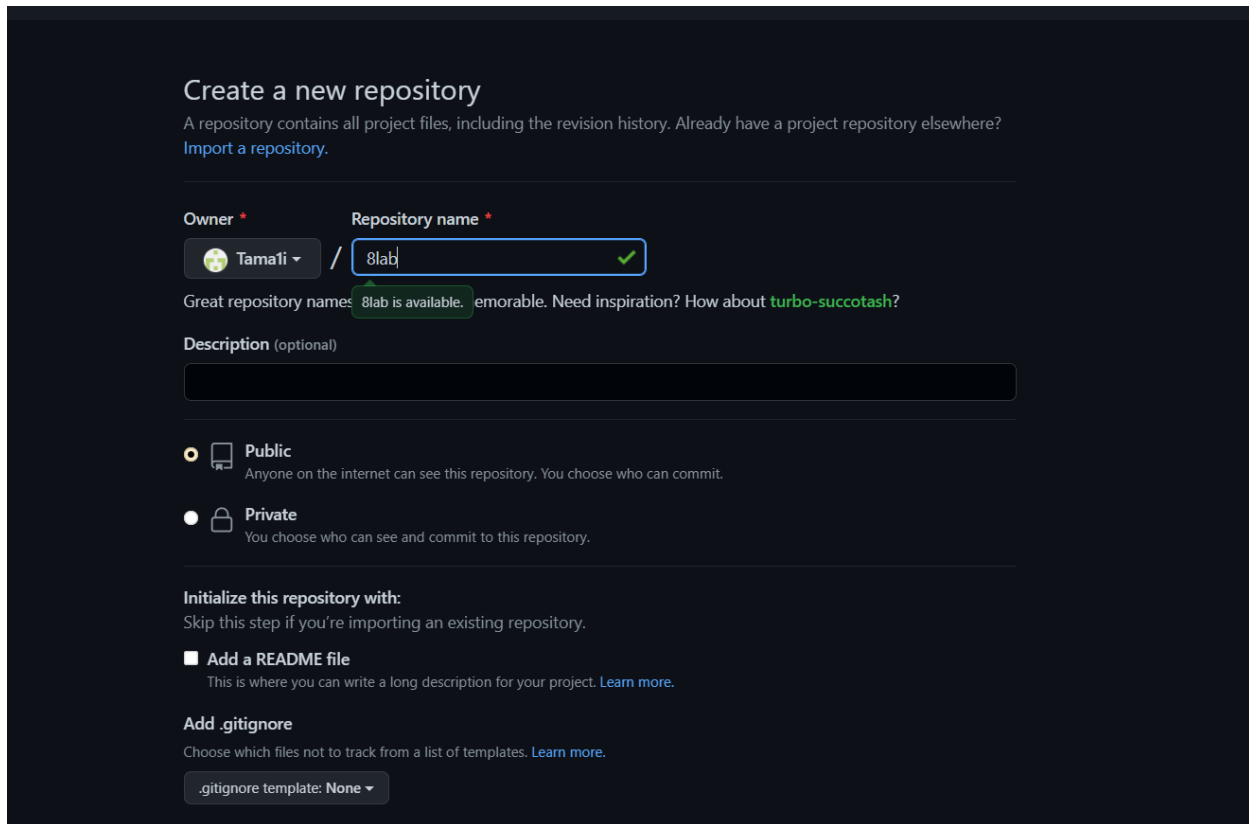
Выполнил: студент 2
курса группы Пиж-б-о-
21-1

Рязанцев Матвей
Денисович

Ставрополь 2022

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

Выполнение работы



Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Owner * Repository name *

Tama1i / 8lab ✓

Great repository names: 8lab is available. memorable. Need inspiration? How about [turbo-succotash?](#)

Description (optional)

☐ Public
Anyone on the internet can see this repository. You choose who can commit.

☐ Private
You choose who can see and commit to this repository.

Initialize this repository with:
Skip this step if you're importing an existing repository.

☒ Add a README file
This is where you can write a long description for your project. [Learn more.](#)

Add .gitignore
Choose which files not to track from a list of templates. [Learn more.](#)

.gitignore template: None ▼

Рисунок 1 -создание репозитория

```
D:\gite\8lab>git flow init

Which branch should be used for bringing forth production releases?
- main
Branch name for production releases: [main]
Branch name for "next release" development: [develop]

How to name your supporting branch prefixes?
Feature branches? [feature/]
Bugfix branches? [bugfix/]
Release branches? [release/]
Hotfix branches? [hotfix/]
Support branches? [support/]
Version tag prefix? []
Hooks and filters directory? [D:/gite/8lab/.git/hooks]

D:\gite\8lab>
```

Рисунок 2 - Организация репозитория по модели ветвления git glow

```

D:\gite\7lab>git branch
* develop
  main

D:\gite\7lab>git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.

D:\gite\7lab>git merge develop
Updating 9c8c639..c370396
Fast-forward
 7lab.docx | Bin 0 -> 378856 bytes
 idz/PythonApplication1.py | 22 ++++++
 idz/PythonApplication2.py | 40 ++++++
 ob/ob1.py | 19 ++++++
 ob/ob1_1.py | 20 ++++++
 ob/ob2.py | 33 ++++++
 6 files changed, 134 insertions(+)
 create mode 100644 7lab.docx
 create mode 100644 idz/PythonApplication1.py
 create mode 100644 idz/PythonApplication2.py
 create mode 100644 ob/ob1.py
 create mode 100644 ob/ob1_1.py
 create mode 100644 ob/ob2.py

```

Рисунок 2 – слияние веток

```

pythonApplication3.py*  X
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

import sys

if __name__ == '__main__':
    # Ввести список одной строкой.
    A = list(map(int, input().split()))
    # Проверить количество элементов списка.
    if len(A) != 10:
        print("Неверный размер списка", file=sys.stderr)
        exit(1)

    # Найти искомую сумму.
    s = sum([a for a in A if abs(a) < 5])
    print(s)

```

Рисунок 3 - Код программы пример 1 с помощью списковых включений следующим образом

```
C:\Users\ryaza\AppData\Local\Programs\Pyt
1 2 4 5 6 7 8 9 10 3 6
10
Press any key to continue . . .
```

Рисунок 4 – результат работы программы пример 1

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

import sys

if __name__ == '__main__':
    # Ввести список одной строкой.
    A = list(map(int, input().split()))
    # Проверить количество элементов списка.
    if len(A) != 10:
        print("Неверный размер списка", file=sys.stderr)
        exit(1)

    # Найти искомую сумму.
    s = 0
    for item in A:
        if abs(item) < 5:
            s += item
    print(s)
```

Рисунок 5 – код программы пример 1

```
C:\Users\ryaza\AppData\Local\Programs\Pyt
1 2 3 4 5 6 7 8 9 10
10
Press any key to continue . . .
```

Рисунок 6 – результат работы программы

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

if __name__ == '__main__':
    a = ((int(input("team1 ")), "team1"), (int(input("team2 ")), "team2"), (int(input("team3 ")), "team3"))
    m = [input(" 1 "), input(" 2 "), input(" 3 ")]
    p = True

    for i, val in enumerate(m):
        for j in a:
            if m[i] == j[1]:
                m[i] = int(j[0])

    for i in range(0, 1):
        if int(m[i]) < int(m[i+1]):
            p = False

    if p == False:
        print("ne pravilno")
    else:
        print("pravilno")
```

Рисунок 7 – код программы идз

```
C:\Users\ryaza\AppData\Local\Programs\Python\Python310\python.exe
team1 22
team2 21
team3 23
 1 team3
 2 team1
 3 team2
pravilno
Press any key to continue . . .
```

Рисунок 8 – результат работы прораммы

Контрольные вопросы

1. Что такое кортежи в языке Python?

Кортеж (tuple) – это неизменяемая структура данных, которая по своему подобию очень похожа на список.

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 придутся как нельзя кстати. Кортежи в памяти занимают меньший объем по сравнению со списками. Кортежи работают быстрее, чем списки

3. Как осуществляется создание кортежей?

a = ()

b = tuple()

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.

6. Какую роль играют кортежи в множественном присваивании?

Используя множественное присваивание, можно провернуть интересный трюк: обмен значениями между двумя переменными.

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 $T2 = T1[i:j]$

здесь

- T2 – новый кортеж, который получается из кортежа T1;

- T1 – исходный кортеж, для которого происходит срез;
- i, j – соответственно нижняя и верхняя границы среза.

Фактически берутся ко вниманию элементы, лежащие на позициях i, i+1, ..., j-1. Значение j определяет позицию за последним элементом среза.

8. Как выполняется конкатенация и повторение кортежей?

Для кортежей можно выполнять операцию конкатенации, которая обозначается символом +.

$$T3 = T1 + T2$$

9. Как выполняется обход элементов кортежа?

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

10. Как проверить принадлежность элемента кортежу?

Проверка вхождения элемента в кортеж - оператор in.

11. Какие методы работы с кортежами Вам известны?

index(), count().

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

Доступно.

13. Как создать кортеж с помощью спискового включения.

Так же, как и список.