

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 №8
дисциплины «Основы программной инженерии»

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

(подпись)

Руководитель практики:
Воронкин Р.А., кандидат технических
наук, доцент кафедры
инфокоммуникаций

(подпись)

Отчет защищен с оценкой _____ Дата защиты _____

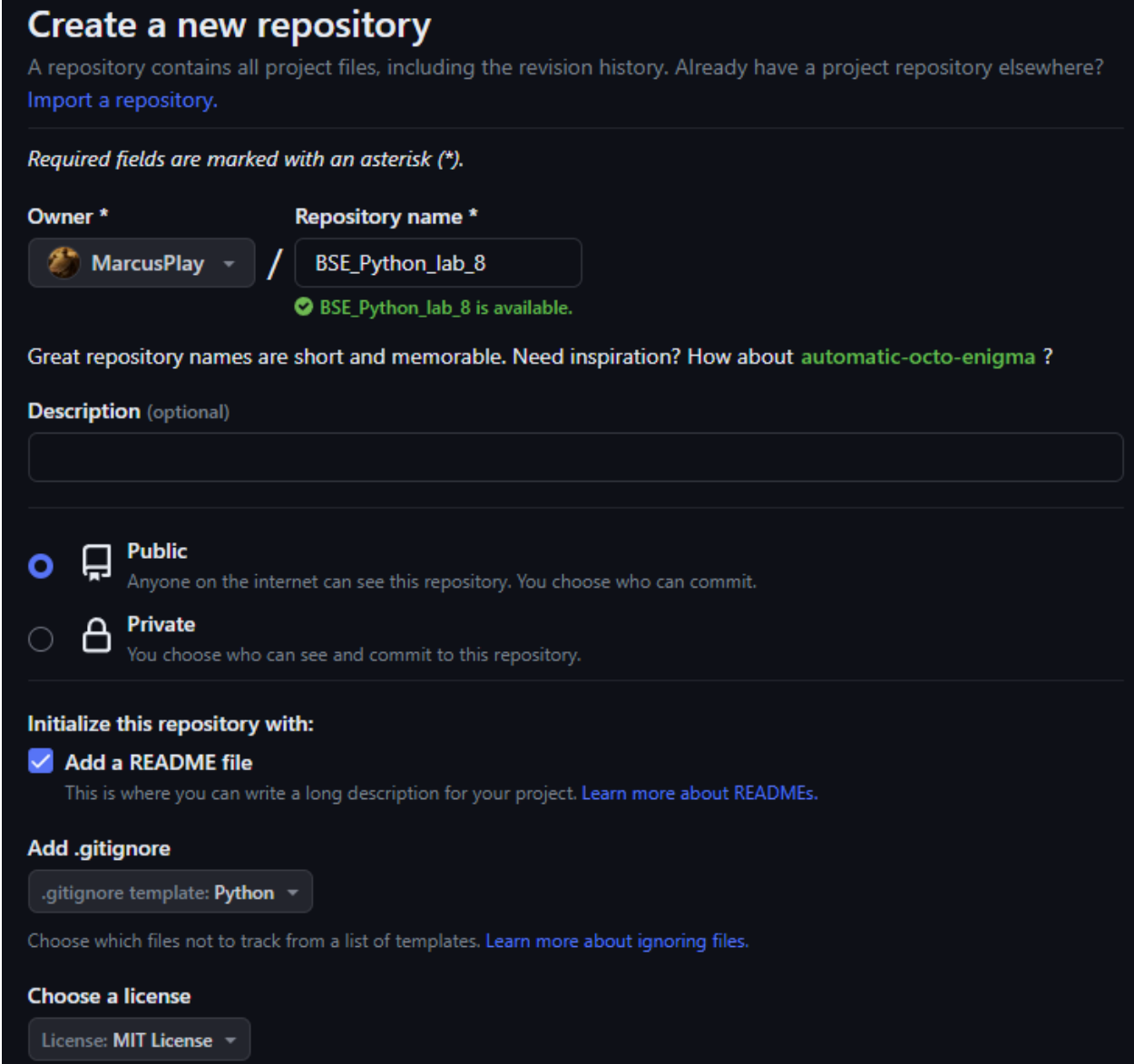
Ставрополь, 2023 г.

Тема: Работа с кортежами в языке Python

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

Методика и порядок выполнения работы


1. Изучить теоретический материал работы.
2. Создать общедоступный репозиторий на GitHub, в котором будет использована лицензия MIT и язык программирования Python.



Create a new repository

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


Required fields are marked with an asterisk ().*


Owner *  **MarcusPlay** / **Repository name ***

✔ BSE_Python_lab_8 is available.

Great repository names are short and memorable. Need inspiration? How about [automatic-octo-enigma](#) ?

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:

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

Add .gitignore

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

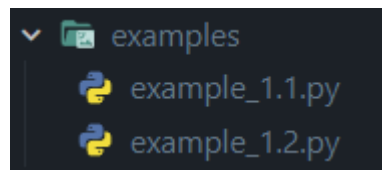
Choose a license

3. Выполните клонирование созданного репозитория.

```
Admin@Genuine MINGW64 ~/Desktop/СКФУ/Основное образование/Основы программной инженерии/Python/Лабораторная работа 8
$ git clone https://github.com/MarcusPlay/BSE_Python_lab_8.git
Cloning into 'BSE_Python_lab_8'...
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 5 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (5/5), done.
```

4. Организуйте свой репозиторий в соответствии с моделью ветвления git-flow.

5. Проработайте примеры лабораторной работы. Создайте для каждого примера отдельный модуль языка Python. Зафиксируйте изменения в репозитории.



```
Admin@Genuine MINGW64 ~/Desktop/СКФУ/Основное образование/Основы программной инженерии/Python/Лабораторная работа 8
python_lab_8 (develop)
$ git commit -m "created folder examples"
[develop 33bed51] created folder examples
2 files changed, 38 insertions(+)
create mode 100644 examples/example_1.1.py
create mode 100644 examples/example_1.2.py
```

6. Приведите в отчете скриншоты результатов выполнения каждой из программ примеров при различных исходных данных, вводимых с клавиатуры.

Пример 1.

```

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     # Проверить количество элементов кортежа.
11     if len(A) != 10:
12         print("Неверный размер кортежа", file=sys.stderr)
13         exit(1)
14
15     # Найти искомую сумму.
16     s = 0
17     for item in A:
18         if abs(item) < 5:
19             s += item
20
21     print(s)

```

```

⊗ PS C:\Users\Admin\Desktop\СКФУ\Основное образование\Основы
   овное образование/Основы программной иженерии/Python/Лабора
   1 2 3 4 5 6
   Неверный размер кортежа
● PS C:\Users\Admin\Desktop\СКФУ\Основное образование\Основы
   овное образование/Основы программной иженерии/Python/Лабора
   1 2 3 4 5 6 7 8 9 10
   10
   PS C:\Users\Admin\Desktop\СКФУ\Основное образование\Основы
   овное образование/Основы программной иженерии/Python/Лабора
   1 20 3 40 5 40 30 203 40 10
   4

```

Пример 2.

```

1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
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)

```

```

⊗ 1 2 3 4 5 6
   Неверный размер списка
PS C:\Users\Admin\Desktop\СКФУ\Основное образование\Основы программной
   овное образование/Основы программной иженерии/Python/Лабораторная работ
● 1 2 3 4 5 6 7 8 9 10
   10
PS C:\Users\Admin\Desktop\СКФУ\Основное образование\Основы программной
● овное образование/Основы программной иженерии/Python/Лабораторная работ
  1 20 3 40 5 40 30 203 40 10
  4

```

7. Приведите в отчете скриншоты работы программ решения индивидуального задания.

```
individual_task_1.py > ...
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  # В начале кортежа записано несколько равных между собой элементов. Определить
5  # количество таких элементов и вывести все элементы, следующие за последним из них.
6  # Рассмотреть возможность того, что весь массив заполнен одинаковыми элементами.
7  # Условный оператор не использовать.
8
9  if __name__=="__main__":
10     t = tuple(map(int, input("Введите кортеж значений через пробел: ").split()))
11
12     count = 0
13     index = 0
14
15     while index < (len(t) - 1) and t[index] == t[index + 1]:
16         count += 1
17         index += 1
18
19     print(count + 1, t[index + 1:])
Введите кортеж значений через пробел: 1 1 1 1 1 5 5 4 4 3 4
5 (5, 5, 4, 4, 3, 4)
Введите кортеж значений через пробел: 1 1 1 1 1 1
6 ()
```

8. Зафиксируйте сделанные изменения в репозитории.

```
$ git commit -m "added individual tasks"
[develop 49d7ebf] added individual tasks
1 file changed, 20 insertions(+)
create mode 100644 individual_task_1.py
```

9. Добавьте отчет по лабораторной работе в *формате PDF* в папку *doc* репозитория. Зафиксируйте изменения.
10. Выполните слияние ветки для разработки с веткой *main / master*.
11. Отправьте сделанные изменения на сервер GitHub.
12. Отправьте адрес репозитория GitHub на электронный адрес преподавателя.

Контрольные вопросы

1. Списки в языке Python — это упорядоченные коллекции элементов, которые могут содержать объекты различных типов данных. Они представляют собой изменяемую структуру данных, которая позволяет добавлять, удалять и изменять элементы.

2. Кортежи в Python — это упорядоченные коллекции элементов, подобные спискам, но они являются неизменяемыми. Это значит, что один раз созданный кортеж нельзя изменить, добавить или удалить элементы. Кортежи используются для представления неизменяемых последовательностей данных.

3. Для создания кортежа в Python, вы можете использовать круглые скобки и перечислить элементы через запятые. Например:

```
my_tuple = (1, 2, 3)
```

4. Доступ к элементам кортежа осуществляется по индексу, начиная с 0. Например, чтобы получить доступ к первому элементу кортежа, вы можете использовать `my_tuple[0]`.

5. Распаковка (деструктуризация) кортежа позволяет присвоить значения элементов кортежа переменным. Например:

```
a, b, c = my_tuple
```

6. Кортежи играют важную роль в множественном присваивании, как показано в предыдущем ответе. Вы можете присваивать значения нескольким переменным одновременно, используя кортеж.

7. Для выбора элементов кортежа с помощью среза, вы можете использовать следующий синтаксис:

```
subset = my_tuple[start:end]
```

8. Для конкатенации (соединения) и повторения кортежей, вы можете использовать операторы `+` и `*` соответственно. Например:

```
concatenated_tuple = tuple1 + tuple2
```

```
repeated_tuple = my_tuple * 3
```

9. Обход элементов кортежа можно выполнить с помощью цикла `for`. Например:

```
for item in my_tuple:  
    print(item)
```

10. Чтобы проверить принадлежность элемента кортежу, вы можете использовать оператор `in`. Например:

```
if item in my_tuple:  
    print("Item is in the tuple")
```

11. Некоторые методы работы с кортежами включают `count()` для подсчета вхождений элемента и `index()` для поиска индекса элемента в кортеже.

12. Да, можно использовать функции агрегации, такие как `len()`, `sum()`, `max()`, `min()`, для работы с кортежами, поскольку они являются итерируемыми структурами данных.

13. Для создания кортежа с помощью спискового включения, вы можете использовать выражение внутри круглых скобок. Например:

```
my_tuple = tuple([x * 2 for x in range(5)])
```

Это создаст кортеж, содержащий удвоенные значения от 0 до 8.