

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ
ФЕДЕРАЦИИ**

**Федеральное государственное автономное образовательное учреждение
высшего образования**

«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт цифрового развития

Кафедра информационных систем и технологий

Отчет по лабораторной работе №7.

Дисциплина: «Основы программной инженерии»

Выполнил:

Студент группы ПИЖ-б-о-22-1,
направление подготовки: 09.03.04

«Программная инженерия»

ФИО: Джараян Арег Александрович

Проверил:

Воронкин Р. А.

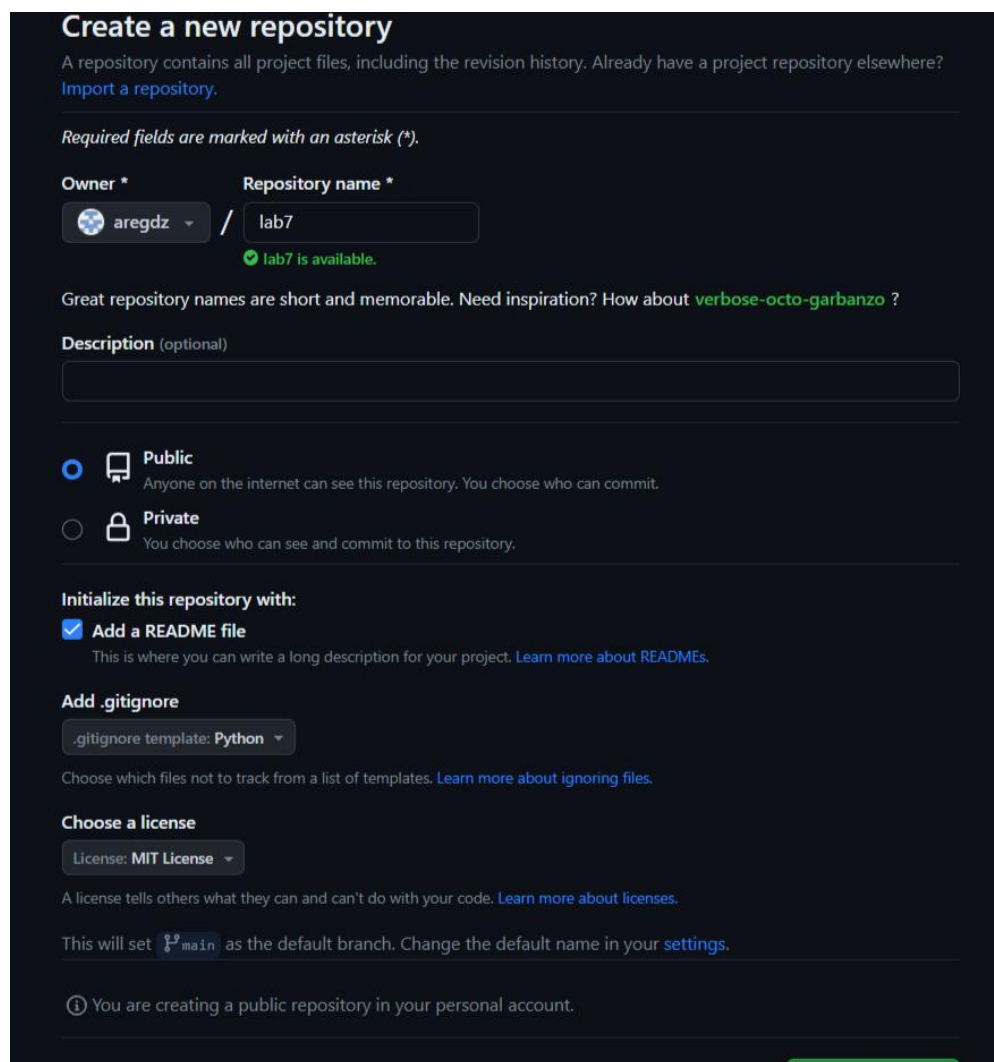
Ставрополь 2022

Тема: Лабораторная работа 2.4 Работа со списками в языке Python

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

Выполнение работы:

1. Изучил теоретический материал работы.
2. Создал репозиторий на git.hub.



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 * aregdz / **Repository name *** lab7

lab7 is available.

Great repository names are short and memorable. Need inspiration? How about **verbose-octo-garbanzo** ?

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
.gitignore template: Python

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

Choose a license
License: MIT License

A license tells others what they can and can't do with your code. [Learn more about licenses.](#)

This will set `main` as the default branch. Change the default name in your [settings](#).

ⓘ You are creating a public repository in your personal account.

Рисунок 1 – создание репозитория

3. Клонировал репозиторий.

```
MINGW64:/c/Users/aregd/OneDrive/Рабочий стол/git7
cd
aregd@DESKTOP-5KV9QA9 MINGW64 ~
$ cd "C:\Users\aregd\OneDrive\Рабочий стол\git7"
aregd@DESKTOP-5KV9QA9 MINGW64 ~/OneDrive/Рабочий стол/git7
$ git clone https://github.com/aregdz/lab7.git
Cloning into 'lab7'...
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.
aregd@DESKTOP-5KV9QA9 MINGW64 ~/OneDrive/Рабочий стол/git7
$
```

Рисунок 2 – клонирование репозитория 4.

Дополнить файл gitignore необходимыми правилами.

```
.gitignore – Блокнот
Файл Правка Формат Вид Справка
# Created by .ignore support plugin (hsz.mobi)
### Python template
# Byte-compiled / optimized / DLL files
__pycache__/
*.py[cod]
*$py.class

# C extensions
*.so

# Distribution / packaging
.Python
env/
build/
develop-eggs/
dist/
downloads/
```

Рисунок 3 - - .gitignore для IDE PyCharm

5. Организовать свой репозиторий в соответствии с моделью ветвления git-flow.

```
aregd@DESKTOP-5KV9QA9 MINGW64 ~/OneDrive/Рабочий стол/git 5/lab5 (main)
$ git checkout -b develop
Switched to a new branch 'develop'
aregd@DESKTOP-5KV9QA9 MINGW64 ~/OneDrive/Рабочий стол/git 5/lab5 (develop)
$ |
```

Рисунок 4 – создание ветки develop

6. Составить программу с использованием одномерных массивов для решения задачи. Номер варианта необходимо получить у преподавателя. Решить индивидуальное задание как с использованием циклов, так и с использованием List Comprehensions.

Ввести список A из 10 элементов, найти сумму элементов, больших 3 и меньших 8 и вывести ее на экран.

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

Рисунок 5 – задание 1

```
1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4 import sys
5
6 if __name__ == '__main__':
7     A = list(map(int, input().split()))
8     if len(A) != 10:
9         print("Неверный размер списка", file=sys.stderr)
10        exit(1)
11
12    # Найти и вывести сумму элементов
13    s = sum([a for a in A if a > 3 and a < 8])
14    print(s)
```

Рисунок 6 – задание 1

```
C:\Users\aregd\AppData\Local\Programs\Python\Python311\python.exe
1 2 3 4 5 6 7 8 9 0
22
```

Рисунок 7 – результат выполнения задания 1

7. В списке, состоящем из вещественных элементов, вычислить:

1.максимальный элемент списка;

2.сумму элементов списка, расположенных до последнего положительного элемента.

Сжать список, удалив из него все элементы, модуль которых находится в интервале $[a, b]$. Освободившиеся в конце списка элементы заполнить нулями.

```
1 # !/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4 import sys
5
6 if __name__ == '__main__':
7     A = list(map(float, input("Введите элементы списка: ").split()))
8     a = int(input("Введите значения a: "))
9     b = int(input("Введите значения b: "))
10    print(f"Максимальный элемент списка - {max(A)}")
11    l = 0
12    for i, item in enumerate(A):
13        if item > 0:
14            l = i
15
16    s = sum(A[:l+1])
17    print(f"Сумма элементов списка, расположенных до последнего положительного элемента - {s}")
18
19    d = [i for i in A if abs(i)>a and abs(i)<b]
20    print(f"Сжатый список - {d}")
21
22    la = len(A)
23    ld = len(d)
24    while ld != la:
25        d.append(0)
26        ld = len(d)
27    print(f"Список с добавленными 0 - {d}")
```

Рисунок 8 – решение задания 1

```
C:\Users\aregd\AppData\Local\Programs\Python\Python311\python.exe "C:\Users\aregd\OneDrive\Рабочий стол\git7\zadanie2.py"
Введите элементы списка: 12.2 34.5 2 -5 -8 3 5 6 3 6 7 3 32 5464 435 -345 345 2435 -2435 5 6 7 8 9 0 345 567
Введите значения a: 5
Введите значения b: 400
Максимальный элемент списка - 5464.0
Сумма элементов списка, расположенных до последнего положительного элемента - 6951.200000000001
Сжатый список - [12.2, 34.0, -8.0, 6.0, 6.0, 7.0, 32.0, -345.0, 345.0, 6.0, 7.0, 8.0, 9.0, 345.0]
Список с добавленными 0 - [12.2, 34.0, -8.0, 6.0, 6.0, 7.0, 32.0, -345.0, 345.0, 6.0, 7.0, 8.0, 9.0, 345.0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
Process finished with exit code 0
```

Рисунок 9 – результат выполнения задания 1

8. Проработал примеры из методички.

```

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

```

Рисунок 10 – пример 1

```

C:\Users\aregd\AppData\Local\Programs\
1 2 3 4 5 6 7 8 9 0
10

Process finished with exit code 0
|

```

Рисунок 11 – пример выполнения первого примера

```

1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  import sys
5
6  if __name__ == '__main__':
7      # Ввести список одной строкой.
8      a = list(map(int, input().split()))
9      # Если список пуст, завершить программу.
10     if not a:
11         print("Заданный список пуст", file=sys.stderr)
12         exit(1)
13
14     # Определить индексы минимального и максимального элементов.
15     a_min = a_max = a[0]
16     i_min = i_max = 0
17     for i, item in enumerate(a):
18         if item < a_min:
19             i_min, a_min = i, item
20         if item >= a_max:
21             i_max, a_max = i, item
22
23     # Проверить индексы и обменять их местами.
24     if i_min > i_max:
25         i_min, i_max = i_max, i_min
26
27     # Посчитать количество положительных элементов.
28     count = 0
29     for item in a[i_min + 1:i_max]:
30         if item > 0:

```

Рисунок 12 – пример 2

```

C:\Users\aregd\AppData\Local\Programs
1 2 23 4 5 6 46 2435
6

Process finished with exit code 0

```

Рисунок 13 – пример выполнения примера 2

9. Зафиксировал все изменения в github в ветке develop.

```

aregd@DESKTOP-5KV9QA9 MINGW64 ~/OneDrive/Рабочий стол/git7/lab7 (develop)
$ git add .

aregd@DESKTOP-5KV9QA9 MINGW64 ~/OneDrive/Рабочий стол/git7/lab7 (develop)
$ git status
On branch develop
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   PyCharm/zadanie1(1).py
        new file:   PyCharm/zadanie1.py
        new file:   PyCharm/zadanie2.py

aregd@DESKTOP-5KV9QA9 MINGW64 ~/OneDrive/Рабочий стол/git7/lab7 (develop)
$ git commit -m"Сохранение изменений"
[develop e79bbd2] Сохранение изменений
3 files changed, 45 insertions(+)
create mode 100644 PyCharm/zadanie1(1).py
create mode 100644 PyCharm/zadanie1.py
create mode 100644 PyCharm/zadanie2.py

```

Рисунок 14 – фиксация изменений в ветку develop

10. Слил ветки.

```

aregd@DESKTOP-5KV9QA9 MINGW64 ~/OneDrive/Рабочий стол/git7/lab7 (develop)
$ git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.

aregd@DESKTOP-5KV9QA9 MINGW64 ~/OneDrive/Рабочий стол/git7/lab7 (main)
$ git merge develop
Updating f380191..63b7060
Fast-forward
 PyCharm/primer1.py      | 13 ++++++++
 PyCharm/primer2.py      | 27 ++++++++
 PyCharm/zadanie1(1).py | 10 +++++
 PyCharm/zadanie1.py     | 13 +++++
 PyCharm/zadanie2.py     | 22 +++++
5 files changed, 85 insertions(+)
create mode 100644 PyCharm/primer1.py
create mode 100644 PyCharm/primer2.py
create mode 100644 PyCharm/zadanie1(1).py
create mode 100644 PyCharm/zadanie1.py
create mode 100644 PyCharm/zadanie2.py

aregd@DESKTOP-5KV9QA9 MINGW64 ~/OneDrive/Рабочий стол/git7/lab7 (main)
$

```

Рисунок 15 – сливание ветки develop в ветку main

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, как было уже сказано выше, 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.

3. Как организовано хранение списков в оперативной памяти?

При создании списка 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?

С помощью цикла:

```
my_list = ['один', 'два', 'три', 'четыре', 'пять']  
for elem in my_list:  
    print(elem)
```

5. Какие существуют арифметические операции со списками?

Списки можно сложить используя “+”.

6. Как проверить есть ли элемент в списке?

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

7. Как определить число вхождений заданного элемента в списке?

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

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

Можно указать индекс списка, куда нужно вставить новый элемент.

Также можно воспользоваться `append(один элемент)` и `extend(сразу несколько элементов)`.

9. Как выполнить сортировку списка?

Можно воспользоваться методом `sort`.

10. Как удалить один или несколько элементов из списка?

Удалить элемент можно, написав его индекс в методе `pop`. Если не указывать индекс, то функция удалит последний элемент. Элемент можно удалить с помощью метода `remove`. Можно удалить несколько элементов с помощью оператора среза и `del`. Можно удалить все элементы из списка с помощью метода `clear`.

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

List Comprehensions чаще всего на русский язык переводят как абстракция списков или списковое включение, является частью синтаксиса языка, которая предоставляет простой способ построения списков.

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?

Используя метод copy().

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

Основное отличие между sorted и sort заключается в том, что sorted создает новый отсортированный список, оставляя исходный список без изменений, в то время как sort сортирует сам список, изменяя его. Выбор между ними зависит от ваших потребностей и того, нужно ли вам сохранить оригинальный порядок элементов.