

**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 ПРАКТИЧЕСКОЙ РАБОТЕ №2.4
дисциплины
«Основы кроссплатформенного программирования»**

Выполнила студентка группы
ИВТ-б-о-21-1
Яковлева Е.А. « » _____ 20__ г.
Подпись студента _____
Работа защищена
« » _____ 20__ г.

Проверил доцент
Кафедры инфокоммуникаций,
старший преподаватель
Воронкин Р.А.

(подпись)

Ставрополь, 2022 г.

Тема: Работа со списками в языке Python

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

Порядок выполнения работы:

1. Создала общедоступный репозиторий на GitHub, в котором использованы лицензия MIT и язык программирования Python.

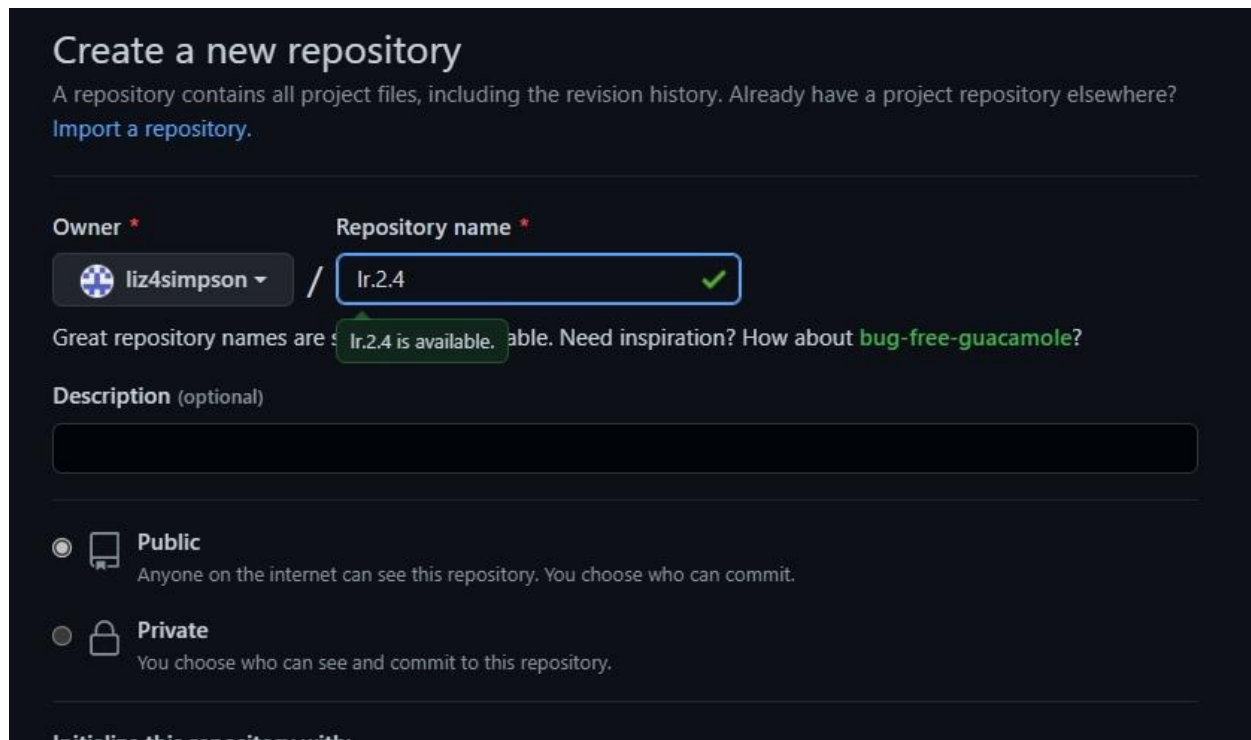


Рисунок 1. Создание репозитория

Выполнила клонирование созданного репозитория.

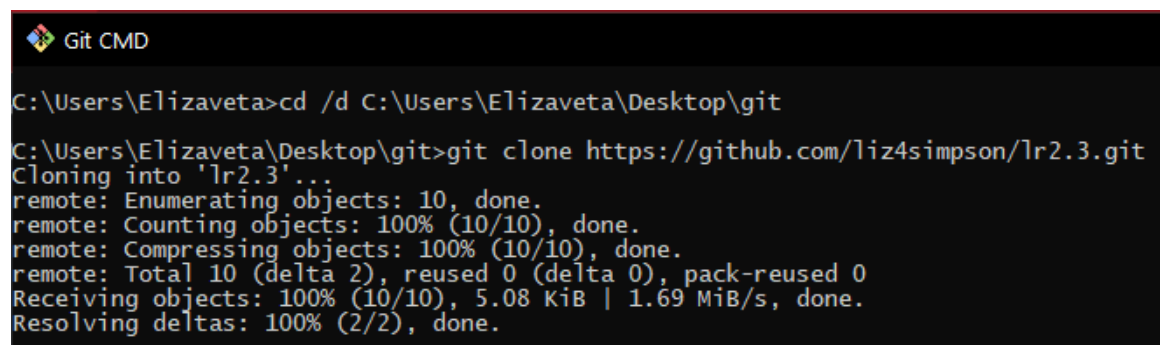


Рисунок 2. Клонирование репозитория

Организовала свой репозиторий в соответствии с моделью ветвления git-flow.

```

C:\Users\Elizaveta>cd /d C:\Users\Elizaveta\Desktop\git\lr2.4
C:\Users\Elizaveta\Desktop\git\lr2.4>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? [C:/Users/Elizaveta/Desktop/git/lr2.4/.git/hooks]

```

Рисунок 3. Организация репозитория согласно модели ветвления get-flow

2. Создала проект PyCharm в папке репозитория, проработала примеры из лабораторной работы.

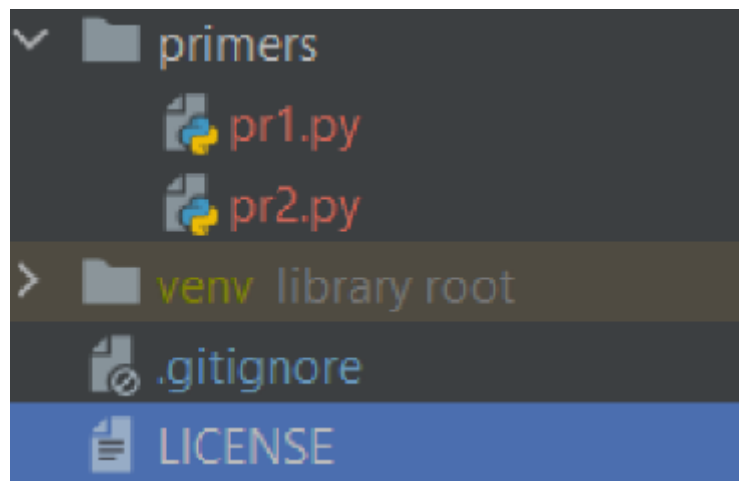


Рисунок 4. Проработанные примеры

```

1 5 4 6 8 7 9 5 1 3
9

```

Рисунок 5. Результат выполнения программы

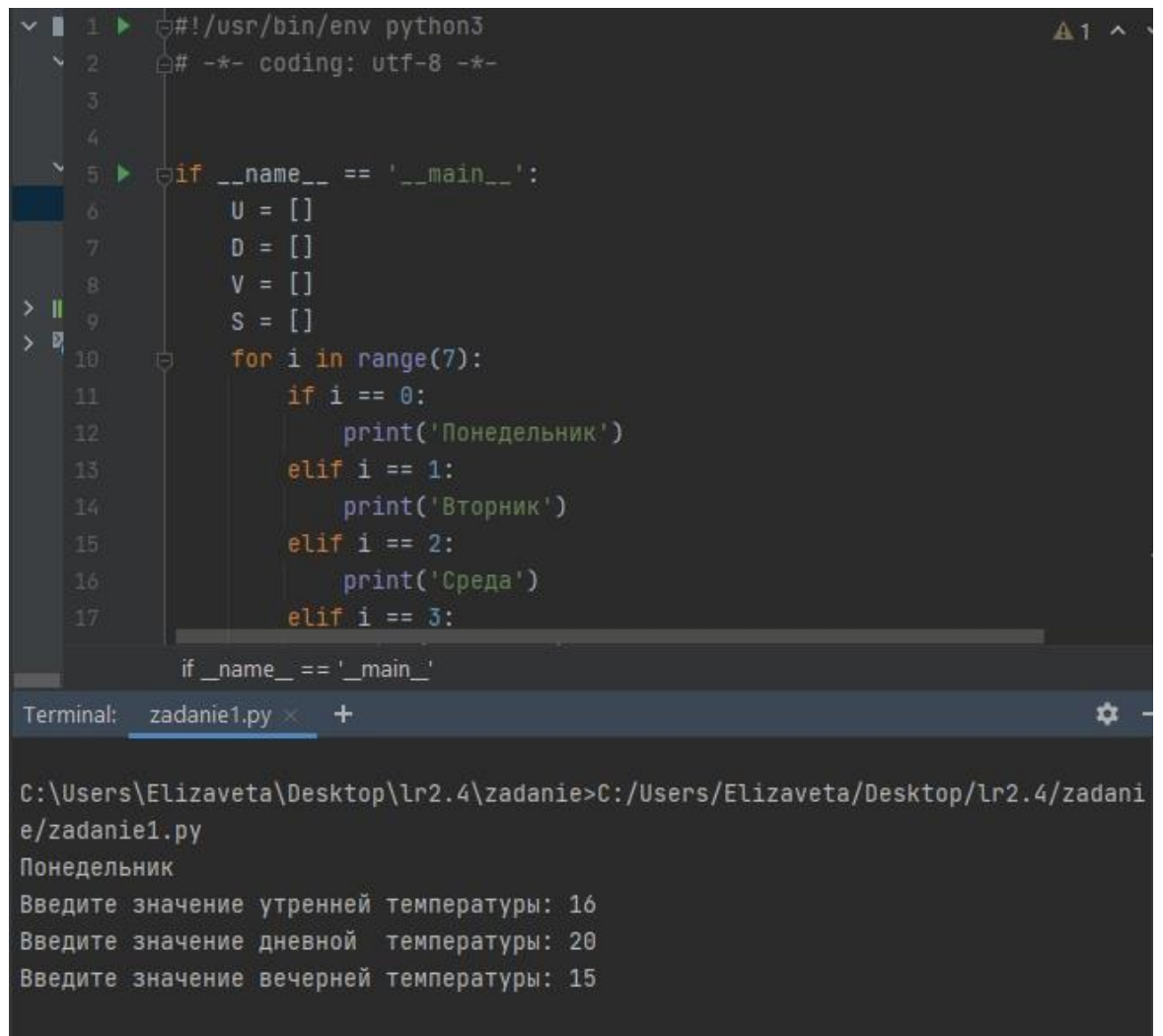
```

2 3 6 5 4 8 9 4 8 7
13

```

Рисунок 6. Результат выполнения программы

3. Выполнила индивидуальное задание (в22)



The image shows a code editor window with a Python script and a terminal window below it. The script defines lists U, D, V, and S, and a loop that prints days of the week based on a range of 7. The terminal shows the execution of the script, which prints 'Понедельник' and prompts for temperature values.

```
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4
5  if __name__ == '__main__':
6      U = []
7      D = []
8      V = []
9      S = []
10     for i in range(7):
11         if i == 0:
12             print('Понедельник')
13         elif i == 1:
14             print('Вторник')
15         elif i == 2:
16             print('Среда')
17         elif i == 3:
```

if __name__ == '__main__'

Terminal: zadanie1.py x +

C:\Users\Elizaveta\Desktop\lr2.4\zadanie>C:/Users/Elizaveta/Desktop/lr2.4/zadanie/zadanie1.py

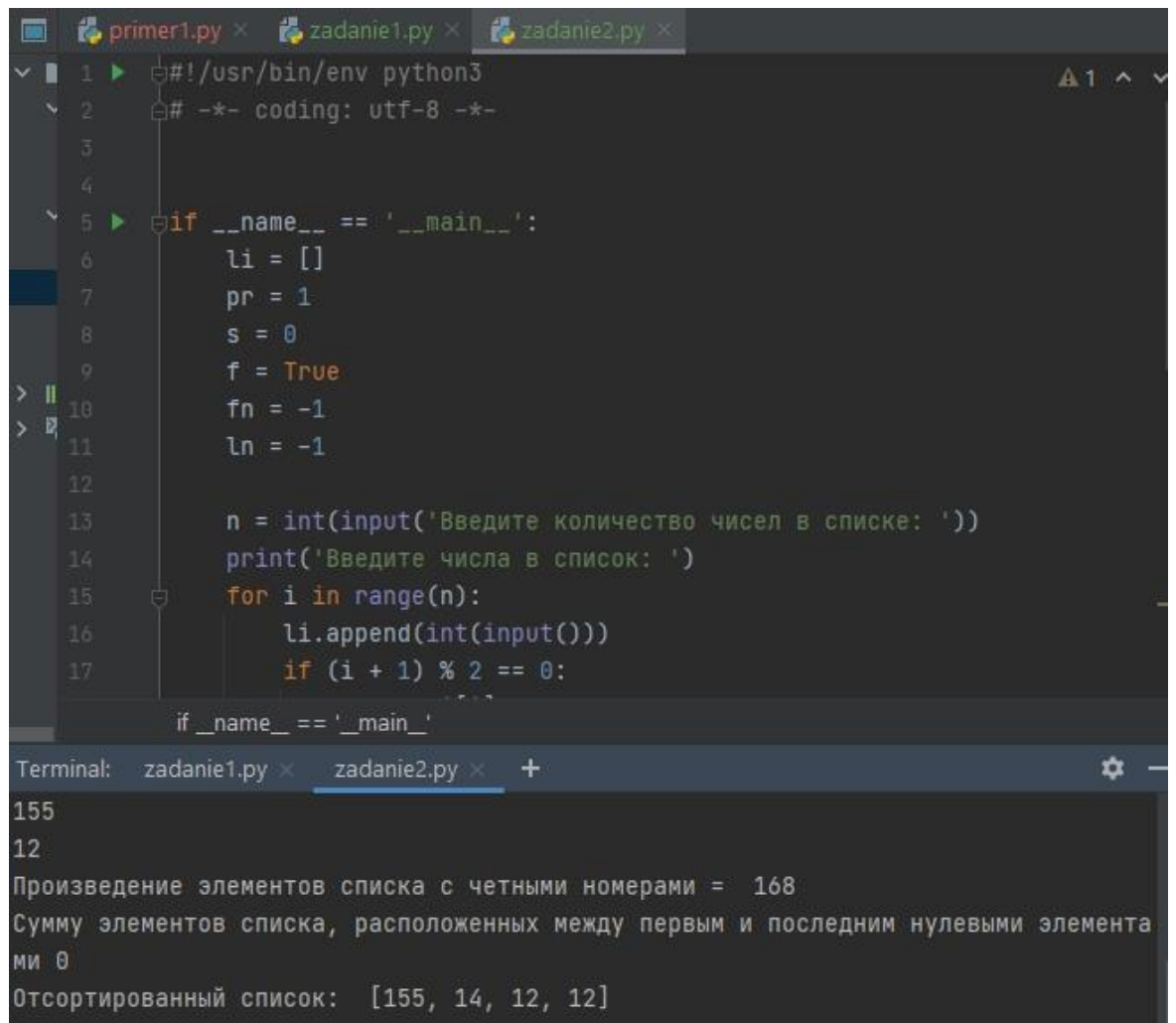
Понедельник

Введите значение утренней температуры: 16

Введите значение дневной температуры: 20

Введите значение вечерней температуры: 15

Рисунок 7. Индивидуальное задание



The image shows a code editor with three tabs: `primer1.py`, `zadanie1.py`, and `zadanie2.py`. The `zadanie1.py` tab is active, displaying the following Python code:

```
1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4
5 if __name__ == '__main__':
6     li = []
7     pr = 1
8     s = 0
9     f = True
10    fn = -1
11    ln = -1
12
13    n = int(input('Введите количество чисел в списке: '))
14    print('Введите числа в список: ')
15    for i in range(n):
16        li.append(int(input()))
17        if (i + 1) % 2 == 0:
```

Below the code editor is a terminal window with the following output:

```
Terminal:  zadanie1.py  zadanie2.py  +
155
12
Произведение элементов списка с четными номерами = 168
Сумму элементов списка, расположенных между первым и последним нулевыми элементами 0
Отсортированный список: [155, 14, 12, 12]
```

Рисунок 8. Индивидуальное задание

4. Сделала коммит, выполнила слияние с веткой `main`, и запустила изменения в удаленный репозиторий.

```
C:\Users\Elizaveta\Desktop\lr2.4>git add .
C:\Users\Elizaveta\Desktop\lr2.4>git commit -m "fine"
[develop dec9fc4] fine
4 files changed, 108 insertions(+)
create mode 100644 primer/primer1.py
create mode 100644 primer/primer2.py
create mode 100644 zadanie/zadanie1.py
create mode 100644 zadanie/zadanie2.py
```

Рисунок 9. Фиксация и коммит файлов

```
C:\Users\Elizaveta\Desktop\lr2.4>git merge develop
Updating cbd8295..dec9fc4
Fast-forward
 primer/primer1.py | 20 +++++
 primer/primer2.py | 17 +++++
 zadanie/zadanie1.py | 38 +++++
 zadanie/zadanie2.py | 33 +++++
 4 files changed, 108 insertions(+)
 create mode 100644 primer/primer1.py
 create mode 100644 primer/primer2.py
 create mode 100644 zadanie/zadanie1.py
 create mode 100644 zadanie/zadanie2.py
```

Рисунок 10. Слияние ветки develop с main

```
C:\Users\Elizaveta\Desktop\lr2.4>git push
Enumerating objects: 9, done.
Counting objects: 100% (9/9), done.
Delta compression using up to 4 threads
Compressing objects: 100% (8/8), done.
Writing objects: 100% (8/8), 1.79 KiB | 19.00 KiB/s, done.
Total 8 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), done.
To https://github.com/liz4simpson/lr2.4.git
 cbd8295..dec9fc4 main -> main
```

Рисунок 11. Отправка изменений на удаленный репозиторий







 liz4simpson fine	dec9fc4 3 hours ago	 3 commits
 primer	fine	3 hours ago
 zadanie	fine	3 hours ago
 .gitignore	Update .gitignore	3 hours ago
 LICENSE	Initial commit	3 hours ago

Рисунок 12. Изменения на удаленном репозитории

Вывод: в ходе лабораторной работы были приобретены навыки по работе со списками при написании программ.

Ответы на контрольные вопросы:

1. Что такое списки в языке Python?

Список (list) – это структура данных для хранения объектов различных типов.

2. Как осуществляется создание списка в Python?

Для создания списка нужно заключить элементы в квадратные скобки.

3. Как организовано хранение списков в оперативной памяти?

Список является изменяемым типом данных. При его создании 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?

`for elem in my_list:`

5. Какие существуют арифметические операции со списками?

`+, *`

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?

`list.count('элемент')`

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

Метод `insert` можно использовать, чтобы вставить элемент в список.

9. Как выполнить сортировку списка?

`list.sort()`

10. Как удалить один или несколько элементов из списка?

Удалить элемент можно, написав его индекс в методе `pop`.

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?

`list[::]`

13. Какие существуют функции агрегации для работы со списками?

Для работы со списками Python предоставляет следующие функции:

- `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?

Для создания копии списка необходимо использовать либо метод `copy`, либо использовать оператор среза

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

Отличие заключается в том, что метод `list.sort()` определён только для списков, в то время как `sorted()` работает со всеми итерируемыми объектами