

**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 №2.6**

Дисциплина: «Программирование на Python»

Тема: «Работа со словарями в языке Python»

Выполнил: студентка 2 курса

группы ИВТ-б-о-21-1

Яковлева Елизавета  
Андреевна

Ставрополь 2022

## Выполнение работы:

1. Создали репозиторий в GitHub, в который добавили gitignore, который дополнили правилами для работы с IDE PyCharm с ЯП Python, выбрали лицензию MIT, клонировали его на локальный сервер и организовали в соответствии с моделью fit-flow.

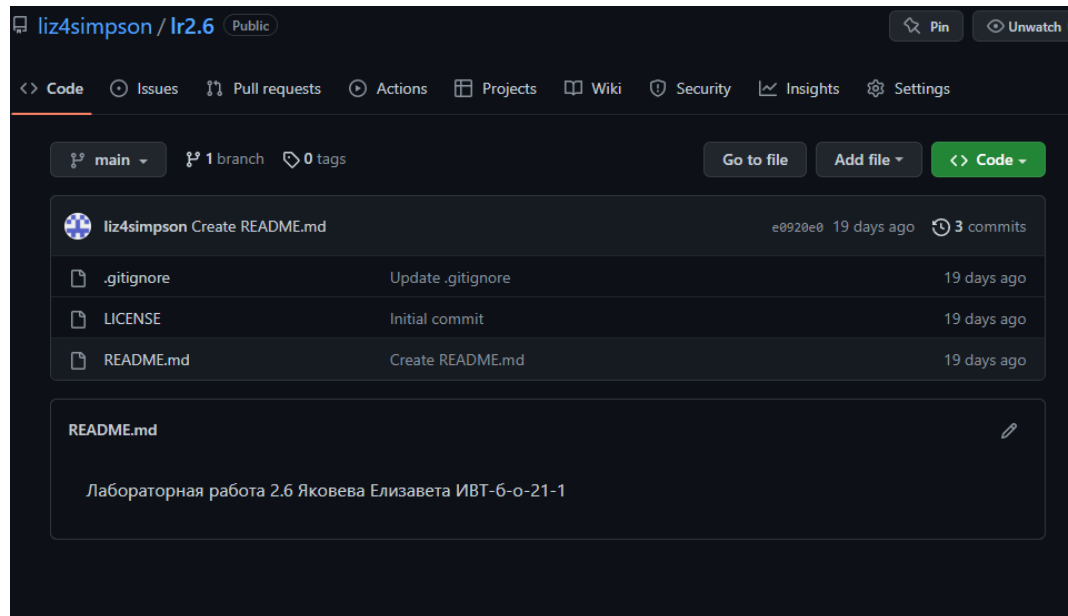


Рисунок 1.1 Созданный репозиторий

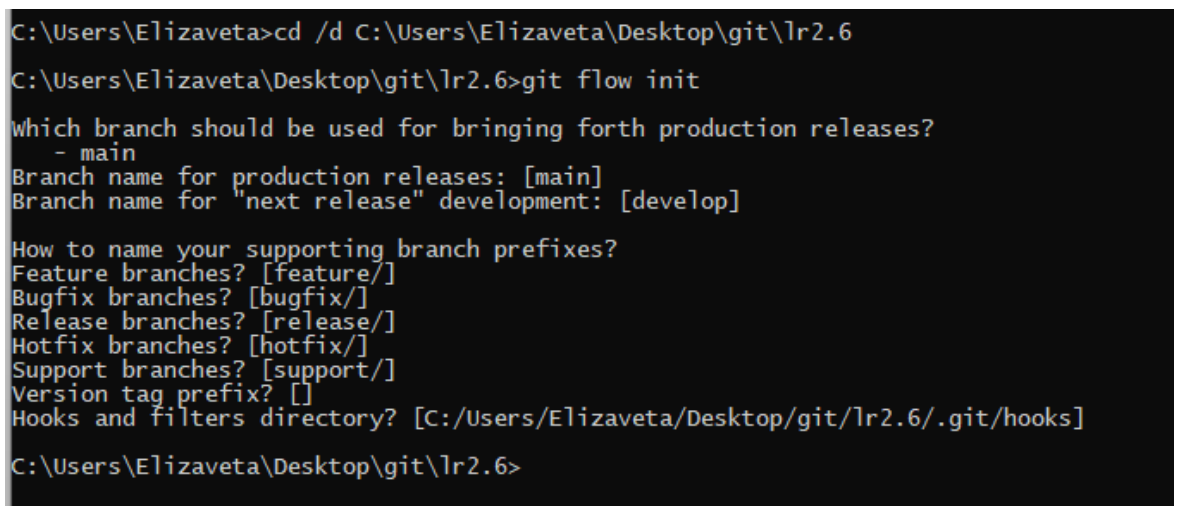
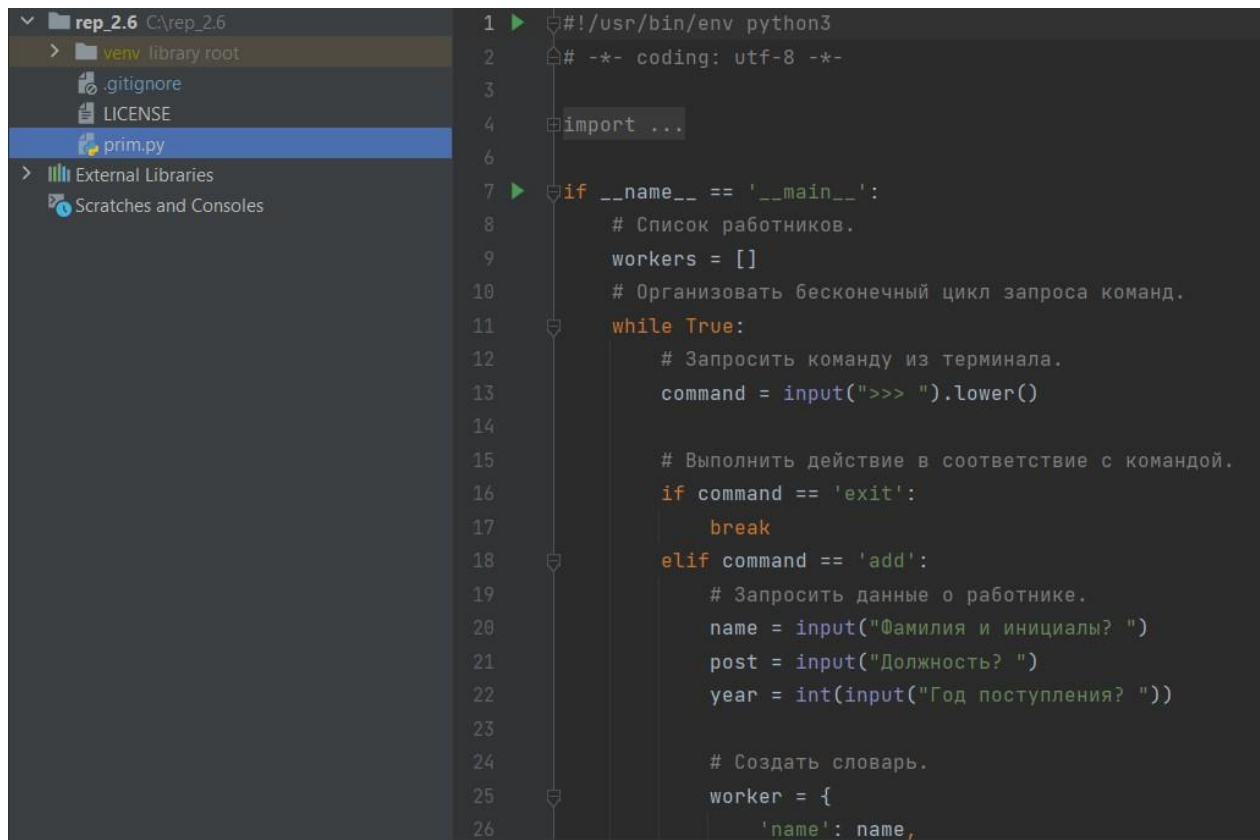


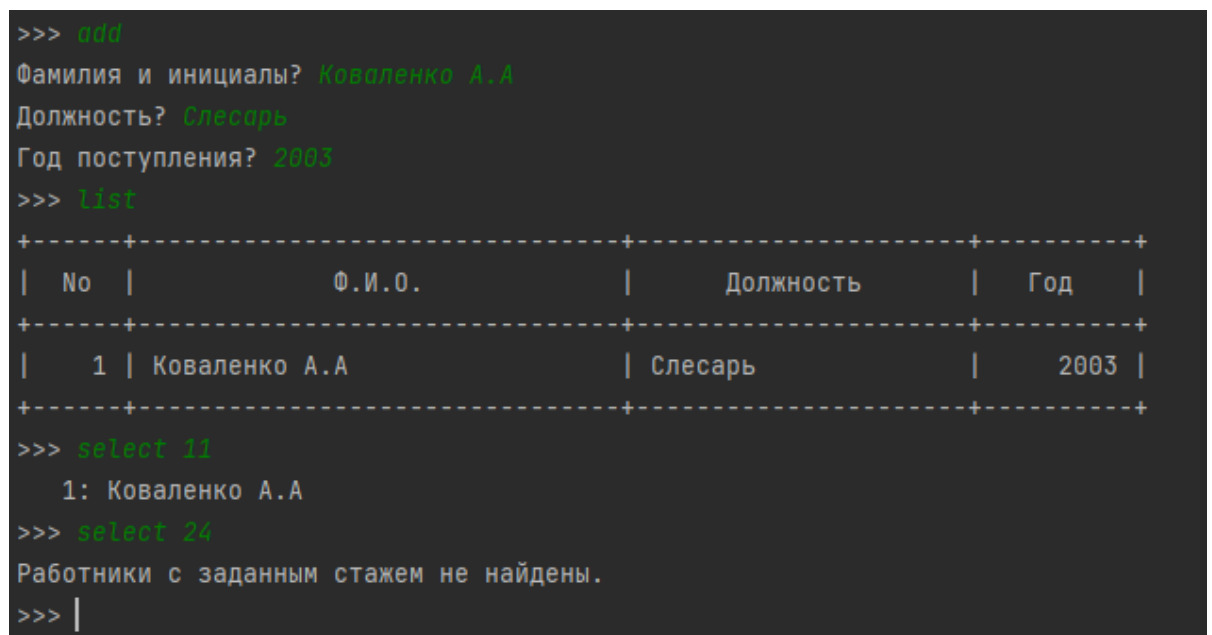
Рисунок 1.2 Организация репозитория в соответствии с моделью ветвления git-flow

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



```
1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4 import ...
5
6
7 if __name__ == '__main__':
8     # Список работников.
9     workers = []
10    # Организовать бесконечный цикл запроса команд.
11    while True:
12        # Запросить команду из терминала.
13        command = input(">>> ").lower()
14
15        # Выполнить действие в соответствие с командой.
16        if command == 'exit':
17            break
18        elif command == 'add':
19            # Запросить данные о работнике.
20            name = input("Фамилия и инициалы? ")
21            post = input("Должность? ")
22            year = int(input("Год поступления? "))
23
24            # Создать словарь.
25            worker = {
26                'name': name,
```

Рисунок 2.1 Создание проекта в PyCharm



```
>>> add
Фамилия и инициалы? Коваленко А.А
Должность? Слесарь
Год поступления? 2003
>>> list
+-----+-----+-----+-----+
| No |          Ф.И.О.          |      Должность      |   Год   |
+-----+-----+-----+-----+
|  1 | Коваленко А.А          |      Слесарь        |   2003   |
+-----+-----+-----+-----+
>>> select 11
      1: Коваленко А.А
>>> select 24
Работники с заданным стажем не найдены.
>>> |
```

Рисунок 2.2 Результат выполнения программы

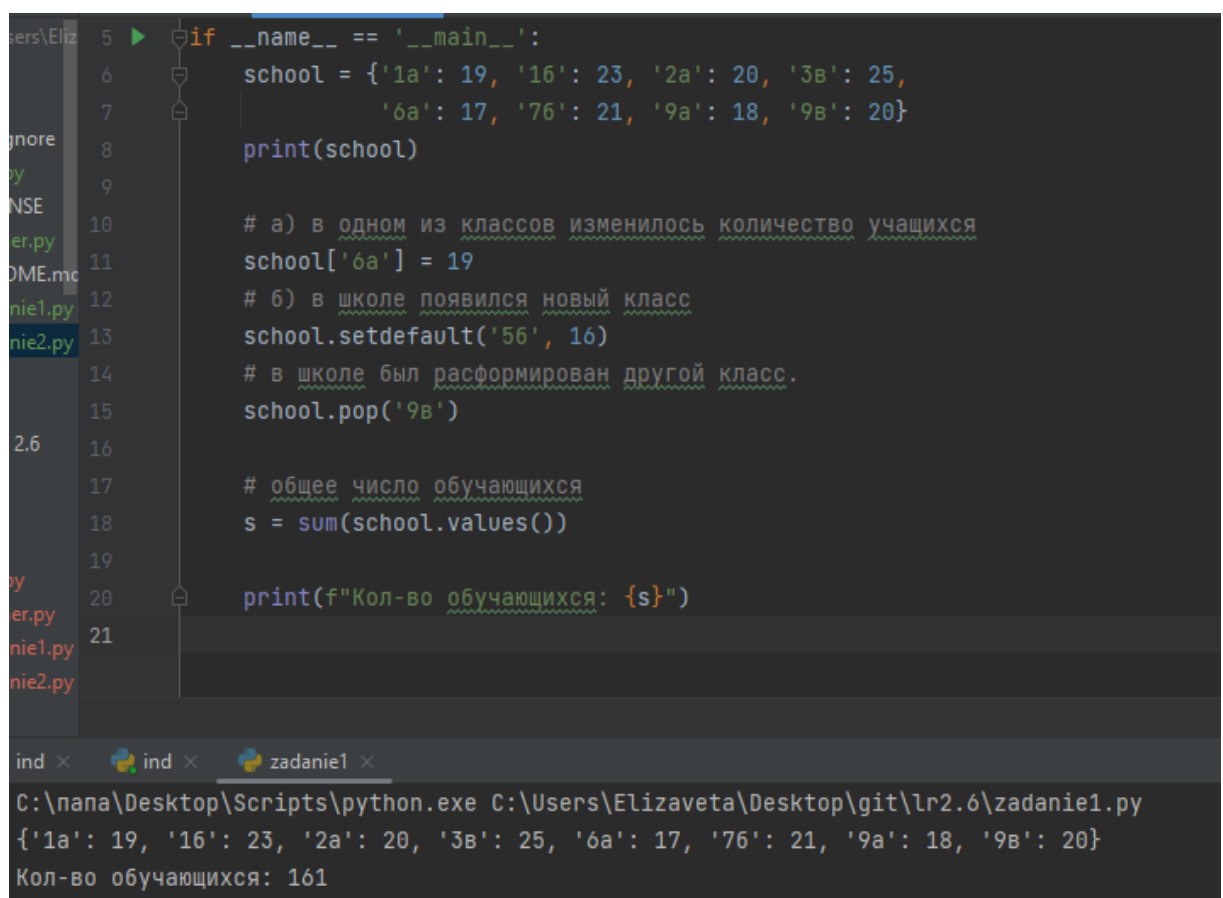
### 3. Выполнила задания

Решите задачу: создайте словарь, связав его с переменной `school`, и наполните данными, которые бы отражали количество учащихся в разных классах (1а, 1б, 2б, 6а, 7в и т. п.). Внесите изменения в словарь согласно следующему:

а) в одном из классов изменилось количество учащихся, б) в школе появился новый класс,

с) в школе был расформирован (удален) другой класс. Вычислите общее количество учащихся в школе.

Решите задачу: создайте словарь, где ключами являются числа, а значениями – строки. Примените к нему метод `items()`, с помощью полученного объекта `dict_items` создайте новый словарь, "обратный" исходному, т. е. ключами являются строки, а значениями – числа.



```
5 if __name__ == '__main__':
6     school = {'1a': 19, '1б': 23, '2а': 20, '3в': 25,
7             '6а': 17, '7б': 21, '9а': 18, '9в': 20}
8     print(school)
9
10    # а) в одном из классов изменилось количество учащихся
11    school['6а'] = 19
12    # б) в школе появился новый класс
13    school.setdefault('5б', 16)
14    # в школе был расформирован другой класс.
15    school.pop('9в')
16
17    # общее число обучающихся
18    s = sum(school.values())
19
20    print(f"Кол-во обучающихся: {s}")
21
```

Output:

```
C:\nana\Desktop\Scripts\python.exe C:\Users\Elizaveta\Desktop\git\lr2.6\zadanie1.py
{'1a': 19, '1б': 23, '2а': 20, '3в': 25, '6а': 17, '7б': 21, '9а': 18, '9в': 20}
Кол-во обучающихся: 161
```

Рисунок 3.1 Вывод программы задания

```
1 ▶ #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4
5 ▶ if __name__ == '__main__':
6     nums = {1: 'one', 2: 'two', 3: 'three', 4: 'four', 5: 'five'}
7     print(nums)
8
9     # инвертирование словаря
10    new_nums = {n: d for d, n in nums.items()}
11    print(new_nums)

```

f\_\_name\_\_ == '\_\_main\_\_'

ind × ind × zadanie2 ×

C:\nana\Desktop\Scripts\python.exe C:\Users\Elizaveta\Desktop\git\lr2.6\zadanie2.py

{1: 'one', 2: 'two', 3: 'three', 4: 'four', 5: 'five'}

{'one': 1, 'two': 2, 'three': 3, 'four': 4, 'five': 5}

Рисунок 3.2 Вывод программы задания

4. (26 вариант). Выполнила индивидуальное задание.

```
>>> add
Пункт назначения Москва
Номер поезда 12
Время отправления 12:00
>>> add
Пункт назначения Ставрополь
Номер поезда 48
Время отправления 11:30
>>> list
+-----+-----+-----+-----+
| № | Пункт назначения | Номер поезда | Время отправления |
+-----+-----+-----+-----+
| 1 | Ставрополь | 48 | 11:30 |
| 2 | Москва | 12 | 12:00 |
+-----+-----+-----+-----+
>>> select Москва
1: Номер поезда - Москва, время отправления - 12

```

Рисунок 4.1 Вывод программы индивидуального задания

5. Сделала коммит, выполнила слияние с веткой main, и запустила изменения в уд. репозиторий.

```

C:\Users\Elizaveta\Desktop\git\lr2.6>git add .
C:\Users\Elizaveta\Desktop\git\lr2.6>git commit -m "programs"
[develop 13e31a7] programs
4 files changed, 244 insertions(+)
create mode 100644 project/ind.py
create mode 100644 project/primer.py
create mode 100644 project/zadanie1.py
create mode 100644 project/zadanie2.py
C:\Users\Elizaveta\Desktop\git\lr2.6>git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
C:\Users\Elizaveta\Desktop\git\lr2.6>git push
Everything up-to-date

```

Рисунок 4.1 Коммит и пуш изменений и переход на ветку main

```

C:\Users\Elizaveta\Desktop\git\lr2.6>git merge develop
Updating e0920e0..13e31a7
Fast-forward
 project/ind.py      | 109 ++++++
 project/primer.py   | 104 ++++++
 project/zadanie1.py | 20 ++++++
 project/zadanie2.py | 11 ++++++
4 files changed, 244 insertions(+)
create mode 100644 project/ind.py
create mode 100644 project/primer.py
create mode 100644 project/zadanie1.py
create mode 100644 project/zadanie2.py

```


Рисунок 4.2 Слияние ветки main с develop

```

C:\Users\Elizaveta\Desktop\git\lr2.6>git push
Enumerating objects: 8, done.
Counting objects: 100% (8/8), done.
Delta compression using up to 4 threads
Compressing objects: 100% (7/7), done.
Writing objects: 100% (7/7), 3.41 KiB | 698.00 KiB/s, done.
Total 7 (delta 2), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (2/2), completed with 1 local object.
To https://github.com/liz4simpson/lr2.6.git
e0920e0..13e31a7 main -> main

```

Рисунок 4.3 Пуш изменений на удаленный сервер


liz4simpson programs
13e31a7 10 minutes ago
🕒 4 commits

📁 project	programs	10 minutes ago
📄 .gitignore	Update .gitignore	19 days ago
📄 LICENSE	Initial commit	19 days ago
📄 README.md	Create README.md	19 days ago

README.md

Лабораторная работа 2.6 Яковева Елизавета ИВТ-6-о-21-1

Рисунок 4.4 Изменения на удаленном сервере

Контр. вопросы и ответы на них:

**1. Что такое словари в языке Python?**

Словари в Python — это изменяемые отображения ссылок на объекты, доступные по ключу.

**2. Может ли функция len() быть использована при работе со словарями.**

Функция len() возвращает длину (количество элементов) в объекте. Аргумент может быть последовательностью, такой как строка, байты, кортеж, список или диапазон или коллекцией (такой как словарь, множество или неизменяемое множество).

**3. Какие методы обхода словарей Вам известны?**

Самый очевидный вариант обхода словаря — это попытаться напрямую запустить цикл for по объекту словаря, так же как мы делаем это со списками, кортежами, строками и любыми другими итерируемыми объектами.

for something in currencies:

print(something)

**4. Какими способами можно получить значения из словаря по ключу?**

С помощью метода .get()

**5. Какими способами можно установить значение в словаре по ключу?**

С помощью функции dict.update()

**6. Что такое словарь включений?**

Словарь включений аналогичен списковым включениям, за исключением того, что он создаёт объект словаря вместо списка.

**7. Самостоятельно изучите возможности функции zip() приведите примеры ее использования.**

Функция zip() в Python создает итератор, который объединяет элементы из нескольких источников данных. Эта функция работает со списками, кортежами, множествами и словарями для создания списков или кортежей, включающих все эти данные.

Предположим, что есть список имен и номером сотрудников, и их нужно объединить в массив кортежей. Для этого можно использовать функцию `zip()`. Вот пример программы, которая делает именно это:

```
employee_numbers = [2, 9, 18, 28]
employee_names = ["Дима", "Марина", "Андрей", "Никита"]
zipped_values = zip(employee_names, employee_numbers)
```

```
zipped_list = list(zipped_values)
print(zipped_list)
```

Функция `zip` возвращает следующее:

```
[('Дима', 2), ('Марина', 9), ('Андрей', 18), ('Никита', 28)]
```

## **8. Самостоятельно изучите возможности модуля `datetime`. Каким функционалом по работе с датой и временем обладает этот модуль?**

`Datetime` — важный элемент любой программы, написанной на Python. Этот модуль позволяет управлять датами и временем, представляя их в таком виде, в котором пользователи смогут их понимать.

`datetime` включает различные компоненты. Так, он состоит из объектов следующих типов:

- `date` — хранит дату
  - `time` — хранит время
  - `datetime` — хранит дату и время
- Как получить текущие дату и время?

```
import datetime
```

```
dt_now = datetime.datetime.now()
print(dt_now)
```

Результат:

```
2022-09-11 15:43:32.249588
```

### **Получить текущую дату:**

```
from datetime import date
current_date = date.today()
print(current_date)
```

Результат:

```
2022-09-11
```

### **Получить текущее время:**



```
import datetime
```

```
current_date_time = datetime.datetime.now()
```

```
current_time = current_date_time.time() print(current_time)
```

**Результат:**

15:51:05.627643



