

**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 №1**

Дисциплина: «Программирование на Python»

Тема: «Работа со словарями в языке Python»

Выполнил: студент 2 курса

группы ИВТ-б-о-21-1

Уланбекова Айканыш Уланбековна

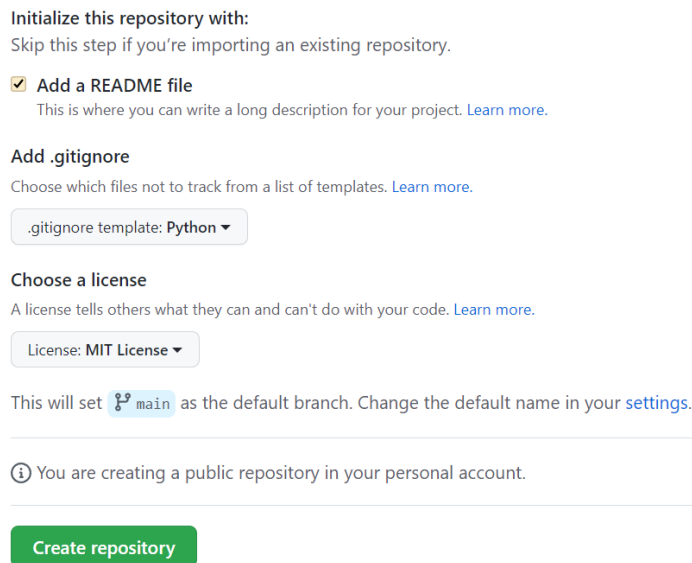
Ставрополь 2022

## Работа со словарями в языке Python

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

### Порядок выполнения работы:

1. Создать общедоступный репозиторий на GitHub, в котором будет использована лицензия MIT и язык программирования Python.



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: Python ▼

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

License: MIT License ▼

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

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

Create repository

Рисунок 1. Создание репозитория

2. Выполните клонирование созданного репозитория.

```
c:\Users\User>cd c:\Users\User\Desktop\2 кypc Python\lab 9
c:\Users\User\Desktop\2 кypc Python\lab 9>git clone https://github.com/aikanyshkaukanbekova/lab-9.git
Cloning into 'lab-9'...
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.
c:\Users\User\Desktop\2 кypc Python\lab 9>
```

Рисунок 2. Клонирование репозитория

3. Дополните файл .gitignore необходимыми правилами для работы с IDE PyCharm.

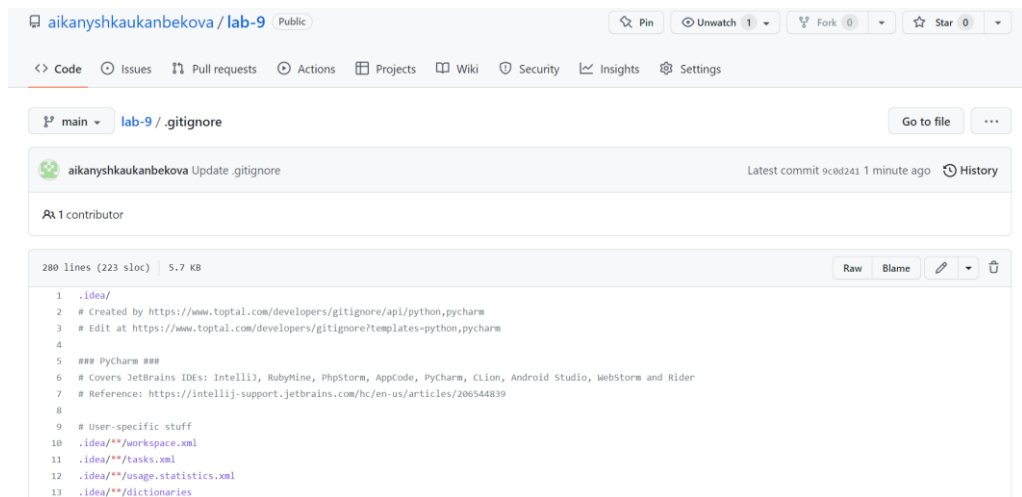


Рисунок 3. Дополнение файла .gitignore

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

```
C:\Users\User>cd C:\Users\User\Desktop\2 кypc Python\lab 9\lab-9
C:\Users\User\Desktop\2 кypc Python\lab 9\lab-9>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/User/Desktop/2 кypc Python/lab 9/lab-9/.git/hooks]
C:\Users\User\Desktop\2 кypc Python\lab 9\lab-9>_
```

Рисунок 4. Организован модель ветвления git flow

5. Проработайте пример лабораторной работы. Создайте для него отдельный модуль языка Python. Зафиксируйте изменения в репозитории.

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

import sys
from datetime import date

if __name__ == '__main__':
    # Список работников.
    workers = []

    # Организовать бесконечный цикл запроса команд.
    while True:
        # Запросить команду из терминала.
        command = input(">>> ").lower()

        # Выполнить действие в соответствии с командой.
        if command == 'exit':
            break
        elif command == 'add':
            # Запросить данные о работнике.
            name = input("Фамилия и инициалы? ")
            post = input("Должность? ")
            year = int(input("Год поступления? "))

            # Создать словарь.
            if __name__ == '__main__':
                while True:
                    if command == 'exit':
                        break
                    elif command == 'add':
                        name = input("Фамилия и инициалы? ")
                        post = input("Должность? ")
                        year = int(input("Год поступления? "))
                        # Создать словарь.
```

Рисунок 5. Пример лаб работы

```

"C:\Users\User\Desktop\2 курс Python\Lab 9\Lab-9\venv\Scripts\python.exe" "C:\Users\User\Desktop\2 курс Python\Lab 9\Lab-9\primer 1.py"
>>> help
Список команд:

add - добавить работника;
list - вывести список работников;
select <стаж> - запросить работников со стажем;
help - отобразить справку;
exit - завершить работу с программой.
>>> add
Фамилия и инициалы? Ulanbekova A.U
Должность? Owner
Год поступления? 2005
>>> list
+-----+-----+-----+-----+
| № |      Ф.И.О.      |      Должность      |      Год      |
+-----+-----+-----+-----+
| 1 | Ulanbekova A.U   | Owner               | 2005          |
+-----+-----+-----+-----+
>>> select
Неизвестная команда 170
>>> select 170
Работники с заданным стажем не найдены.
>>>

```

Рисунок 6. Вывод примера

6. 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.

```

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

if __name__ == '__main__':
    school = {"1а": 8, "1б": 12, "2б": 12, "6а": 4, "7в": 19}
    print(school, "количество учащихся в разных классах")
    school['1а'] = 22
    print(school, "1 изменение")
    school['1г'] = 26
    print(school, "новый класс")
    del school["2б"]
    print(school, "класс расформирован")
    k = sum(school.values())
    print(k, "кол-во учеников")

```

Рисунок 7. Выполненное задание 1

7. 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.

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

if __name__ == '__main__':
    slov = {1: "sid", 2: "Mani", 3: "Shiba"}
    print (slov)
    nn = {}
    for k, v in slov.items():
        nn[v] = k
    print (nn)
```

Рисунок 8. Выполненное задание 2

### Индивидуальное задание:

Использовать словарь, содержащий следующие ключи: название начального пункта маршрута; название конечного пункта маршрута; номер маршрута. Написать программу, выполняющую следующие действия: ввод с клавиатуры данных в список, состоящий из словарей заданной структуры; записи должны быть упорядочены по номерам маршрутов; вывод на экран информации о маршруте, номер которого введен с клавиатуры; если таких маршрутов нет, выдать на дисплей соответствующее сообщение.

```

primer 1.py x zadanie 1.py x zadanie 2.py x indiv.py x
1  ▶  #!/usr/bin/env python3
2  # -*- coding^ utf-8 -*-
3
4
5  import sys
6
7  ▶  if __name__ == '__main__':
8      workers = []
9
10     while True:
11         command = input(">>> ").lower()
12
13         if command == 'exit':
14             break
15
16         elif command == 'add':
17
18             start = input("Название начального пункта маршрута? ")
19             finish = input("Название конечного пункта маршрута? ")
20             number = int(input("Номер маршрута? "))
21
22             worker = {
23                 'start': start,
24                 'finish': finish,

```

```

                'number': number,
            }

            workers.append(worker)

            if len(workers) > 1:
                workers.sort(key=lambda item: item.get('number', ''))
        elif command == 'list':
            line = '+-{}-+-{}-+-{}-+-{}-+'.format(
                '-' * 4,
                '-' * 30,
                '-' * 20,
                '-' * 8
            )
            print(line)
            print(
                '| {:^4} | {:^30} | {:^20} | {:^8} |'.format(
                    "№",
                    "Начальный пункт",
                    "Конечный пункт",
                    "Номер маршрута"
                )
            )
            print(line)

```

```

        for idx, worker in enumerate(workers, 1):
            print(
                '| {:>4} | {:<30} | {:<20} | {:>8} |'.format(
                    idx,
                    worker.get('start', ''),
                    worker.get('finish', ''),
                    worker.get('number', 0)
                )
            )
            print(line)

    elif command.startswith('select '):

        parts = command.split(' ', maxsplit=1)
        count = 0
        period = int(parts[1])

        for worker in workers:
            if worker.get('number') == period:
                count += 1
                print(
                    '{:>4} - {}'.format(worker.get('start', ''),
                                         worker.get('finish', ''))

        if count == 0:
            print("Маршрут с таким номером не найден.")
    elif command == 'help':

        print("Список команд:\n")
        print("add - добавить маршрут;")
        print("list - вывести список маршрутов;")
        print("select <номер маршрута> - запросить данные о маршруте;")
        print("help - отобразить справку;")
        print("exit - завершить работу с программой.")
    else:
        print(f"Неизвестная команда {command}", file=sys.stderr)

```

Рисунки 9. Выполненное индивидуальное задание

8. Сделала коммит, выполнил слияние с веткой main, и запустил изменения в уд. репозиторий.

```

C:\Users\User\Desktop\2 крс Python\lab 9\lab-9>git add .
C:\Users\User\Desktop\2 крс Python\lab 9\lab-9>git commit -m "added programs + modidied .gitignore"
[develop ac3500f] added programs + modidied .gitignore
4 files changed, 208 insertions(+)
 create mode 100644 indiv.py
 create mode 100644 primer 1.py
 create mode 100644 zadanie 1.py
 create mode 100644 zadanie 2.py
C:\Users\User\Desktop\2 крс Python\lab 9\lab-9>git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.

```

Рисунок 10. Коммит и пуш изменений и переход на ветку main

**Ответы на контрольные вопросы:**

## **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