

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 №9**  
**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 со словарями в языке Python.

**Цель работы:** приобретение навыков по работе со словарями 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\_9  
✔ BSE\_Python\_lab\_9 is available.

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

**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.

Create repository

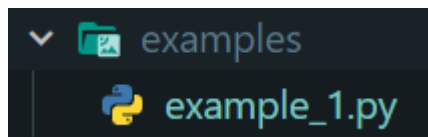
3. Выполните клонирование созданного репозитория.

```
Imran@Kaskad MINGW64 ~/Desktop/Work
$ git clone https://github.com/MarcusPlay/BSE_Python_lab_9.git
Cloning into 'BSE_Python_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.
```

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

```
Imran@Kaskad MINGW64 ~/Desktop/Work/BSE_Python_lab_9 (main)
$ git checkout -b develop
Switched to a new branch 'develop'
```

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



```
Imran@Kaskad MINGW64 ~/Desktop/Work/BSE_Python_lab_9 (develop)
$ git commit -m "added folder examples"
[develop 268b249] added folder examples
1 file changed, 104 insertions(+)
create mode 100644 examples/example_1.py
```

6. Приведите в отчете скриншоты результатов выполнения примера при различных исходных данных, вводимых с клавиатуры.

### Пример 1.

```
#!/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 соответствие 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.
        worker = {
            'name': name,
            'post': post,
            'year': year,
        }

        # Добавить словарь в список.
        workers.append(worker)
        # Отсортировать список в случае необходимости.
        if len(workers) > 1:
            workers.sort(key=lambda item: item.get('name', ''))

    elif command == 'list':
        # Заголовок таблицы.
        line = '+-{}-+-{}-+-{}-+-{}-+'.format(
            '-' * 4,
            '-' * 30,
            '-' * 20,
            '-' * 8
        )
        print(line)
        print(
            '| {:^4} | {:^30} | {:^20} | {:^8} |'.format(
                "№",
                "Ф.И.О.",
                "Должность",
                "Год"
            )
        )
        print(line)

        # Вывести данные о всех сотрудниках.
        for idx, worker in enumerate(workers, 1):
            print(

```

```

        '| {:>4} | {:<30} | {:<20} | {:>8} |'.format(
            idx,
            worker.get('name', ''),
            worker.get('post', ''),
            worker.get('year', 0)
        )
    )

    print(line)

elif command.startswith('select '):
    # Получить текущую дату.
    today = date.today()

    # Разбить команду на части для выделения номера года.
    parts = command.split(' ', maxsplit=1)
    # Получить требуемый стаж.
    period = int(parts[1])

    # Инициализировать счетчик.
    count = 0
    # Проверить сведения работников из списка.
    for worker in workers:
        if today.year - worker.get('year', today.year) >= period:
            count += 1
            print(
                '{:>4}: {}'.format(count, worker.get('name', ''))
            )

    # Если счетчик равен 0, то работники не найдены.
    if count == 0:
        print("Работники с заданным стажем не найдены.")

elif command == 'help':
    # Вывести справку о работе с программой.
    print("Список команд:\n")
    print("add - добавить работника;")
    print("list - вывести список работников;")
    print("select <стаж> - запросить работников со стажем;")
    print("help - отобразить справку;")
    print("exit - завершить работу с программой.")
else:
    print(f"Неизвестная команда {command}", file=sys.stderr)

```

**Вывод:**

```

Imran@Kaskad MINGW64 ~/Desktop/Work/BSE_Python_lab_9/examples (develop)
$ python3 example_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? Magomedov I.V.
Должность? Стажер
Год поступления? 2021
>>> list
+-----+-----+-----+-----+
| №      | Ф.И.О.                | Должность    | Год      |
+-----+-----+-----+-----+
| 1      | Magomedov I.V.        | Стажер       | 2021     |
+-----+-----+-----+-----+
>>> exit

```

7. 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.

**Решение:**

```

if __name__=="__main__":
    classes = {}

    while True:
        command = input("$ ").lower()

        if command == 'exit':
            break

        elif command == 'add' or command == "modified":
            name = input("Введите название класса: ").upper()
            count_stud = int(input("Введите количество учащихся в классе: "))
            classes[name] = count_stud

        elif command == "del":
            name = input("Введите класс, который хотите удалить: ").upper()
            del classes[name]

        elif command == "list":
            line = '+-{}-+-{}-+-{}-+'.format('-' * 4, '-' * 10, '-' * 12)
            print(line)
            print(' | {:^4} | {:^10} | {:^12} |'.format("№", "Название", "Количество"))
            print(line)

            a = 1
            for name, count_stud in classes.items():
                print(' | {:^4} | {:^10} | {:^12} |'.format(a, name, count_stud))
                a += 1
            print(line)

```

```

$ add
Введите название класса: 1a
Введите количество учащихся в классе: 34
$ add
Введите название класса: 2в
Введите количество учащихся в классе: 20
$ list
+-----+-----+-----+
|  №   |  Название  |  Количество  |
+-----+-----+-----+
|  1   |    1А     |    34       |
|  2   |    2В     |    20       |
+-----+-----+-----+
$ modified
Введите название класса: 1a
Введите количество учащихся в классе: 40
$ list
+-----+-----+-----+
|  №   |  Название  |  Количество  |
+-----+-----+-----+
|  1   |    1А     |    40       |
|  2   |    2В     |    20       |
+-----+-----+-----+
$ del
Введите класс, который хотите удалить: 2в
$ list
+-----+-----+-----+
|  №   |  Название  |  Количество  |
+-----+-----+-----+
|  1   |    1А     |    40       |
+-----+-----+-----+

```

8. Зафиксируйте сделанные изменения в репозитории.

```
Imran@Kaskad MINGW64 ~/Desktop/Work/BSE_Python_lab_9 (develop)
$ git commit -m "added task_1.py"
[develop d89d1e1] added task_1.py
1 file changed, 40 insertions(+)
create mode 100644 task_1.py
```

9. 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.

```
task_2.py
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  # Создайте словарь, где ключами являются числа, а значениями – строки. Примените к нему метод items(),
5  # с помощью полученного объекта dict_items создайте новый словарь, "обратный" исходному,
6  # т. е. ключами являются строки, а значениями – числа.
7
8  if __name__ == "__main__":
9      numbers_v1 = {1:"one", 2:"two", 3:"three"}
10     numbers_v2 = {}
11
12     for a, b in numbers_v1.items():
13         numbers_v2[b] = a
14
15     print(numbers_v1)
16     print(numbers_v2)
```

```
Imran@Kaskad MINGW64 ~/Desktop/Work/BSE_Python_lab_9 (develop)
$ python3 task_2.py
{1: 'one', 2: 'two', 3: 'three'}
{'one': 1, 'two': 2, 'three': 3}
```

10. Зафиксируйте сделанные изменения в репозитории.

```
Imran@Kaskad MINGW64 ~/Desktop/Work/BSE_Python_lab_9 (develop)
$ git commit -m "added task_2.py"
[develop b853e14] added task_2.py
1 file changed, 17 insertions(+)
create mode 100644 task_2.py
```

11. Приведите в отчете скриншоты работы решения индивидуального задания.

Использовать словарь, содержащий следующие ключи: фамилия, имя; номер телефона; дата рождения (список из трех чисел). Написать программу, выполняющую следующие действия: ввод с клавиатуры данных в список, состоящий из словарей заданной структуры; записи должны быть размещены



по алфавиту; вывод на экран информации о людях, чьи дни рождения приходятся на месяц, значение которого введено с клавиатуры; если таких нет, выдать на дисплей соответствующее сообщение.

### Решение:

```
Imran@Kaskad MINGW64 ~/Desktop/Work/BSE_Python_lab_9 (develop)
$ python3 individual_task.py
~<| add
Введите Фамилию и имя: магомедов имран
Введите Номер телефона: 8929-345-23-32
Введите дату рождения (пример: 05 07 2004): 05 07 2004
~<| add
Введите Фамилию и имя: Темаев идрис
Введите Номер телефона: 8928-567-34-34
Введите дату рождения (пример: 05 07 2004): 02 07 2005
~<| add
Введите Фамилию и имя: Гойалиев Слутан
Введите Номер телефона: 8928-234-45-67
Введите дату рождения (пример: 05 07 2004): 03 08 2004
~<| list
Введите число месяца (1 - 12):
7
+-----+-----+-----+-----+
| №      | Название           | Номер телефона    | Возраст   |
+-----+-----+-----+-----+
| 2      | магомедов имран    | 8929-345-23-32    | 5 7 2004  |
+-----+-----+-----+-----+
| 3      | Темаев идрис       | 8928-567-34-34    | 2 7 2005  |
+-----+-----+-----+-----+
~<| exit
```

```

if __name__ == "__main__":
    users = []

    while True:
        command = input("~<| ").lower()

        if command == 'exit':
            break

        elif command == 'add':
            name = input('Введите Фамилию и Имя: ')
            phone_number = input('Введите Номер телефона: ')
            year = list(map(int, input('Введите дату рождения (пример: 05 07 2004): ').split()))

            user = {
                'name': name,
                'phone_number': phone_number,
                'year': year,
            }

            users.append(user)

            if len(users) > 1:
                users.sort(key=lambda item: item.get('name', ''))

        elif command == 'list':
            print('Введите число месяца (1 - 12): ')
            while True:
                num = int(input())
                if num < 1 or num > 12:
                    print("Значение введено неправильно! Попробуйте еще раз.")
                else:
                    break

            line = '+-{}--{}--{}--{}--{}--'.format('-' * 4, '-' * 20, '-' * 18, '-' * 10)
            print(line)
            print('| {:^4} | {:^20} | {:^18} | {:^10} |'.format(
                "№",
                "Название",
                "Номер телефона",
                "Дата"
            ))
            print(line)

            for idx, user in enumerate(users, 1):
                if user['year'][1] == num:
                    print(
                        '| {:^4} | {:^20} | {:^18} | {:^10} |'.format(
                            idx,
                            user['name'],
                            user['phone_number'],
                            ' '.join(map(str, user['year']))
                        )
                    )
                    print(line)

```

12. Зафиксируйте сделанные изменения в репозитории.

```
Imran@Kaskad MINGW64 ~/Desktop/Work/BSE_Python_lab_9 (develop)
$ git commit -m "added individual task"
[develop e7fa711] added individual task
1 file changed, 70 insertions(+), 1 deletion(-)
```

13. Добавьте отчет по лабораторной работе в формате PDF в папку doc репозитория. Зафиксируйте изменения.

14. Выполните слияние ветки для разработки с веткой main/master.

```
Imran@Kaskad MINGW64 ~/Desktop/Work/BSE_Python_lab_9 (main)
$ git merge develop
Updating fc3d88a..e7fa711
Fast-forward
 examples/example_1.py | 104 +++++++++++++++++++++++++++++++++++++
 individual_task.py    |  71 +++++++++++++++++++++++++++++++++
 task_1.py             |  40 +++++
 task_2.py             |  18 +++++
 4 files changed, 233 insertions(+)
 create mode 100644 examples/example_1.py
 create mode 100644 individual_task.py
 create mode 100644 task_1.py
 create mode 100644 task_2.py
```

15. Отправьте сделанные изменения на сервер GitHub.

```
Imran@Kaskad MINGW64 ~/Desktop/Work/BSE_Python_lab_9 (main)
$ git push
Enumerating objects: 22, done.
Counting objects: 100% (22/22), done.
Delta compression using up to 12 threads
Compressing objects: 100% (18/18), done.
Writing objects: 100% (21/21), 5.42 KiB | 2.71 MiB/s, done.
Total 21 (delta 7), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (7/7), done.
To https://github.com/MarcusPlay/BSE_Python_lab_9.git
fc3d88a..e7fa711 main -> main
```

16. Отправьте адрес репозитория GitHub на электронный адрес преподавателя.

### Контрольные вопросы:

1. Что такое словари в языке Python?

Словари — это изменяемый тип данных в Python, предназначенный для хранения пар ключ-значение. Ключи в словаре уникальны, и они используются для доступа к соответствующим значениям.

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

Да, функция len() может использоваться для определения количества элементов в словаре. Она возвращает количество ключей в словаре.

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

Для обхода словарей можно использовать циклы for. Например:

```
for key in my_dict:  
    print(key, my_dict[key])
```

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

Значение из словаря можно получить, используя квадратные скобки или метод get(). Например:

```
value = my_dict["key"]
```

или

```
value = my_dict.get("key")
```

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

Значение в словаре можно установить, используя квадратные скобки. Например:

```
my_dict["key"] = value
```

## **6. Что такое словарь включений?**

Словарь включений (или dictionary comprehension) — это синтаксический сахар для создания словарей в одну строку. Пример:

```
my_dict = {key: value for key, value in iterable}
```

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

Функция zip() используется для объединения нескольких итерируемых объектов в кортежи. Пример:

```
names = ["Alice", "Bob", "Charlie"]
```

```
ages = [25, 30, 35]
```

```
zipped_data = zip(names, ages)
```

```
result = dict(zipped_data)
```

Результат: {'Alice': 25, 'Bob': 30, 'Charlie': 35}

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

Модуль datetime предоставляет классы для работы с датой и временем. Некоторые возможности включают создание объектов datetime, форматирование и парсинг дат, арифметические операции с датами, работу с часовыми поясами и многое другое. Примеры:

```
from datetime import datetime, timedelta
```

```
# Создание объекта datetime
```

```
current_time = datetime.now()
```

```
# Форматирование даты в строку
```

```
formatted_time = current_time.strftime("%Y-%m-%d  
%H:%M:%S")
```

```
# Арифметические операции с датами
```

```
new_time = current_time + timedelta(days=1)
```

Эти ответы могут помочь вам подготовиться к защите работы. Убедитесь, что вы также понимаете основные концепции и умеете объяснить выбранные примеры.