МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Департамент цифровых, робототехнических систем и электроники

«Работа с множествами и словарями в языке Python»

Отчет по лабораторной работе № 5
по дисциплине «Программирование на Python»
Вариант 5

Выполнил студент группы ИВТ-б-о-24-1
Грабарь Артемий Павлович
«__» декабря 2025г.
Подпись студента_____
Работа защищена « »_____20__г.
Проверил Воронкин Р.А. _____
(подпись)

Ставрополь 2025

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

Ссылка на GitHub: https://github.com/Arhi258/Laba_5.git

Задание:

1. Создать общедоступный репозиторий на GitHub, в котором будет использована лицензия MIT и язык программирования Python.
2. Выполнить копирование созданного репозитория.
3. Дополнить файл .gitignore необходимыми правилами для работы с IDE PyCharm.
4. Создать проект PyCharm в папке репозитория.
5. Проработать пример лабораторной работы. Создать для него отдельный модуль языка Python. Зафиксировать изменения в репозитории.
6. Привести в отчете скриншоты результатов выполнения примера при различных исходных данных вводимых с клавиатуры.
7. Решить задачу: подсчитайте количество гласных в строке, введенной с клавиатуры с использованием множеств.
8. Зафиксировать сделанные изменения в репозитории.
9. Решить задачу: определите общие символы в двух строках, введенных с клавиатуры.
10. Решить 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.
11. Зафиксировать сделанные изменения в репозитории.
12. Решить 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.

13. Зафиксировать сделанные изменения в репозитории.

14. Привести в отчет скриншоты работы программ индивидуального задания. Задание 1: Определить результат выполнения операций над множествами (рис 1). Считать элементы множества строками. Проверить результат вручную. Задание 2: Составить программу с использованием списков и словарей для решения задачи. Использовать словарь, содержащий следующие ключи: название пункта назначения рейса; номер рейса; тип самолета. Написать программу, выполняющую следующие действия: ввод с клавиатуры данных в список, состоящий из словарей заданной структуры; записи должны быть размещены в алфавитном порядке по названиям пунктов назначения; вывод на экран пунктов назначения и номеров рейсов, обслуживаемых самолетом, тип которого введен с клавиатуры; если таких рейсов нет, выдать на дисплей соответствующее сообщение.

$$
\begin{aligned}
&A = \{c, e, h, n\};\\
&B = \{e, f, k, n, x\};\\
&C = \{b, c, h, p, r, s\};\\
&D = \{b, e, g\};\\
&X = (A/B) \cap (C \cup D);\\
&Y = (A \cap \bar{B}) \cup (C/D).
\end{aligned}
$$

Рисунок 1. Множества и операции над ними

15. Зафиксировать сделанные изменения в репозитории.

16. Добавить отчет по лабораторной работе в формате PDF в папку doc репозитория. Зафиксировать изменения.

17. Отправить сделанные изменения на сервер GitHub.

Ход работы:

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

2. Было выполнено клонирование репозитория (рис 1).

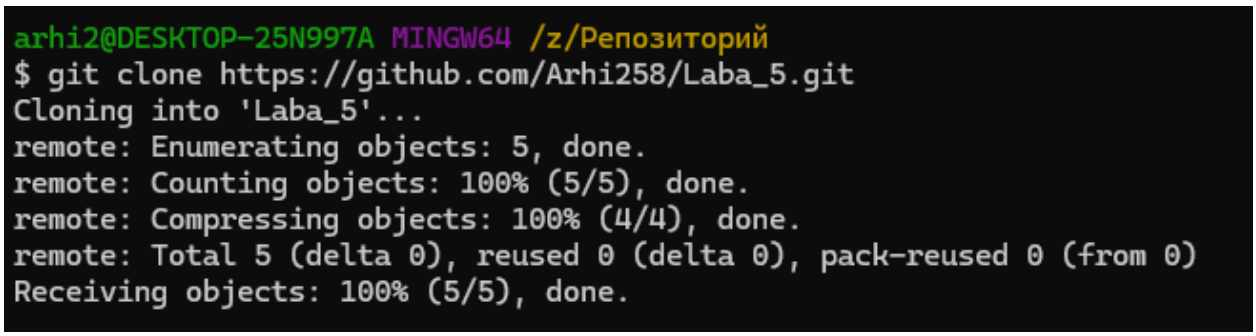

Рисунок 1. Клонирование репозитория

3. Файл .gitignore был дополнен необходимыми правилами для работы с IDE PyCharm.

4. В папке репозитория был создан проект PyCharm.

5. Были проработаны примеры, приведенные в лабораторной работе. Для каждого примера был создан отдельный модуль Python. Изменения были зафиксированы в репозитории.

6. Каждый из примеров был проверен на работоспособность (рис 2, 3). Полный исходный код программы приведен в Приложении 1, 2.

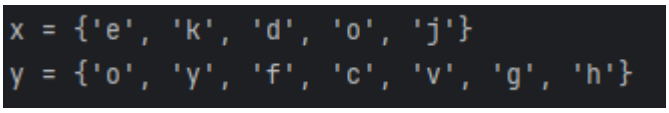

Рисунок 2. Пример 1

```
>>> add
Фамилия и инициалы? Грабарь А.П.
Должность? Студент
Год поступления? 2024
>>> add
Фамилия и инициалы? Верникова П.П.
Должность? Уборщица
Год поступления? 2012
>>> list
+------+--------------------------------+----------------------+----------+
|  №   |             Ф.И.О.             |      Должность       |   Год    |
+------+--------------------------------+----------------------+----------+
|    1 | Верникова П.П.                 | Уборщица             |     2012 |
|    2 | Грабарь А.П.                   | Студент              |     2024 |
+------+--------------------------------+----------------------+----------+
>>> help
Список команд:

add - добавить работника;
list - вывести список работников;
select <стаж> - запросить работников со стажем;
help - отобразить справку;
exit - завершить работу с прогруммой.
>>> select 5
   1: Верникова П.П.
>>> exit

Process finished with exit code 0
```

Рисунок 3. Пример 2

7. Была решена задача: “подсчитайте количество гласных в строке, введенной с клавиатуры с использованием множеств.”. Результат работы программы приведен на рисунке 4. Полный исходный код программы приведен в Приложении 3.

```
Введите строку: ааппп орор ег
Количество гласных в вашей строке = 5
```

Рисунок 4. Результат выполнения задания

8. Была решена задача: “ определите общие символы в двух строках, введенных с клавиатуры.”. Результат работы программы приведен на рисунке 5. Полный исходный код программы приведен в Приложении 4.

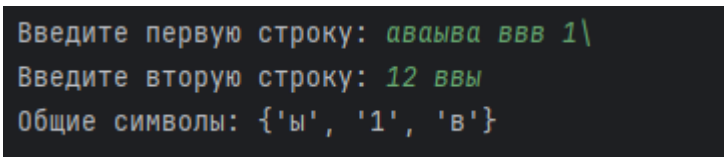

Рисунок 5. Результат выполненного задания

9. Была решена задача: “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.”. Результат работы программы приведен на рисунке 6. Полный исходный код программы приведен в Приложении 5.

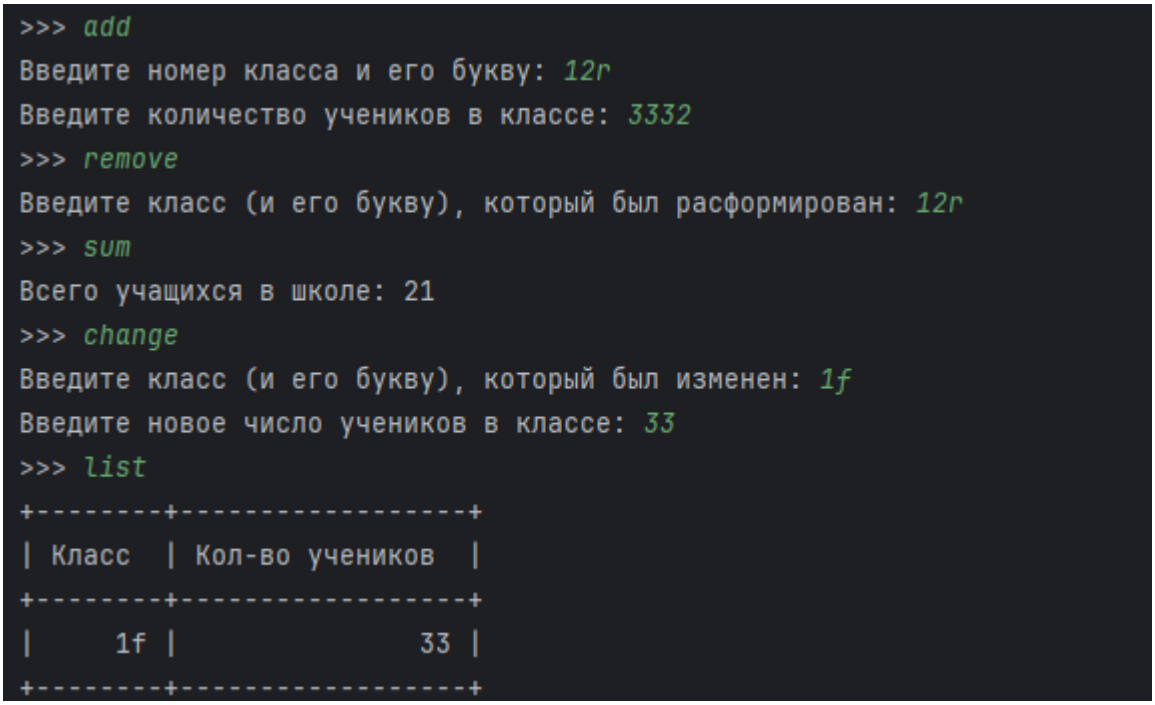

Рисунок 6. Результат выполненного задания

10. Была решена задача: 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.”. Результат работы программы приведен на рисунке 7. Полный исходный код программы приведен в Приложении 6.

```
{'один': 1, 'два': 2, 'три': 3, 'четыре': 4}
```

Рисунок 7. Результат выполненного задания

11. Была решена задача 1. Результат работы программы приведен на рисунке 8. Полный исходный код программы приведен в Приложении 7.

```
{'c', 'h'}
{'h', 'p', 'r', 's', 'c'}
```

Рисунок 8. Результат выполненного задания

12. Была решена задача 2. Результат работы программы приведен на рисунке 9. Полный исходный код программы приведен в Приложении 8.

```
>>> add
Пункт назначения: Пермь
Номер рейса: 12
Тип самолета: Ен7
>>> add
Пункт назначения: Архангельск
Номер рейса: 32
Тип самолета: ро5
>>> list
+-------------------------------+----------------+----------------------+
|        Пункт назначения       |   Номер рейса  |     Тип самолета     |
+-------------------------------+----------------+----------------------+
|                  архангельск |             32 |                  ро5 |
|                        пермь |             12 |                  ен7 |
+-------------------------------+----------------+----------------------+
>>> check_flight
Введите тип самолета: еН7
Вам подходит рейс 12 который направляется в пермь
```

Рисунок 9. Результат выполненного задания

Контрольные вопросы:

1. Множество – неупорядоченная совокупность уникальных значений. В качестве элементов могут выступать любые неизменяемые объекты, такие как числа, символы, строки.

2. Создать множество можно с помощью фигурных скобок или вызова set().

3. Чтобы проверить, есть ли элемент в множестве используется in, для проверки отсутствия not in.

4. Перебор всех элементов множества осуществляется с помощью цикла for и in.

5. Set comprehensions – компактный способ создания множеств.

6. Чтобы добавить новый элемент в множество используется метод add.

7. Функции для удаления элементов: remove – удаление элемента с генерацией исключения в случае, если такого элемента нет; discard – удаление элемента без генерации исключения, если элемент отсутствует; pop – удаление первого элемента, генерируется исключение при попытке удаления из пустого множества. Если необходимо убрать все элементы, используется метод clear().

8. Метод union – объединение элементов двух множеств. Метод update – добавить все элементы одного множества к другому. Функция intersection – поиск общих элементов для двух разных множеств. Метод difference – вычисление разности двух разных множеств.

9. Команда a.issubset(b) определяет, является ли b подмножеством a. Команда a.issuperset(b) определяет, является ли множество a надмножеством b.

10. frozenset – множество, которое не поддается изменению.

11. Для преобразования множества в строку используется конкантенация текстовых значений, которую обеспечивает функция join. Функция dict преобразует множество в словарь. Функция list позволяет преобразовать множество в список.

12. Словарь – изменяемый неупорядоченный набор элементов "ключ: значение".

13. С помощью функции len() можно вывести размер словаря.

14. Обход словарей осуществляется через цикл, в основном цикл for.

15. Метод get() позволяет получить элемент по его ключу.

16. Метод setdefault() позволяет добавить элемент в словарь.

17. Словарь включений – краткий способ создания словарей с помощью генераторных выражений.

18. Функция zip() объединяет элементы из нескольких итерируемых объектов (список, кортеж и т.д.) в один итератор кортежей, где каждый кортеж содержит элементы из одной позиции всех свободных объектов. Эту функцию можно использовать при создании словаря из двух списков.

19. Модуль datetime предоставляет классы для удобной работы с датами и временем. Основные классы модуля: date (работа с датой), time (работа с временем), datetime (комбинация даты и времени), timedelta (представляет разницу между двумя моментами времени).

Вывод: в результате работы были получены навыки по работе с множествами и словарями при написании программ с помощью языка программирования Python версии 3.x.

Приложения.

Приложение 1. Исходный код:

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

if __name__ == "__main__":
    # Определим уникальное множество
    u = set("abcdefghijklmnopqrstuvwxyz")

    a = {"b", "c", "h", "o"}
    b = {"d", "f", "g", "o", "v", "y"}
    c = {"d", "e", "j", "k"}
    d = {"a", "b", "f", "g"}

    x = (a.intersection(b)).union(c)
    print(f"x = {x}")

    # НАйдем дополнения множеств
    bn = u.difference(b)
    cn = u.difference(c)

    y = (a.difference(d).union(cn.difference(bn)))
    print(f"y = {y}")
```

Приложение 2. Исходный код:

```
#!/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.
            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 0б 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 прогруммой.")

        else:
            print(f"Неизвестная команда {command}", file=sys.stderr)
```

Приложение 3. Исходный код:

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

if __name__ == "__main__":
    # Перевод всей строки в нижний регистр
```

```
    a = list(input("Введите строку: ").lower())
    vowels = {'у', 'е', 'ё', 'ы', 'а', 'о', 'э', 'я', 'и', 'ю'}

    common = set(a) & vowels

    count = 0
    for i in common:
        count += a.count(i)

    print(f"Количество гласных в вашей строке = {count}")
```

Приложение 4. Исходный код:

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

if __name__ == '__main__':
    a = set(input("Введите первую строку: ").replace(" ", ""))
    b = set(input("Введите вторую строку: ").replace(" ", ""))

    common = a & b
    print(f"Общие символы: {common}")
```

Приложение 5. Исходный код:

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

import sys


if __name__ == '__main__':
    school = {}

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

        # Выход из программы
        if command == "exit":
            break

        # Добавление класса
        elif command == "add":
            class_school = input("Введите номер класса и его букву: ").lower()
            number_of_students = int(input(
                "Введите количество учеников в классе: "
            ))

            school[class_school] = number_of_students

        # Удаление класса
        elif command == "remove":
            class_remove = input(
                "Введите класс (и его букву), который был расформирован: "
            ).lower()
```

```
    if class_remove not in school:
        print("Такого класса нет!")
    else:
        del school[class_remove]

# Вывод суммы учеников в школе
elif command == "sum":
    all_students = sum(school.values())
    print(f"Всего учащихся в школе: {all_students}")

# Изменение количества учеников
elif command == "change":
    class_change = input(
        "Введите класс (и его букву), который был изменен: "
    )

    if class_change not in school:
        print("Такого класса нет!")
    else:
        school[class_change] = int(input(
            "Введите новое число учеников в классе: "
        ))

# Вывод всех классов в школе
elif command == "list":
    # Заголовок таблицы
    line = '+--------+------------------+'
    print(line)
    print(
        '| {:^6} | {:^16} |'.format(
            'Класс',
            'Кол-во учеников'
        )
    )
    print(line)

    # Вывод данных о классах и количестве учеников
    for class_school, number_of_students in school.items():
        print(
            '| {:>6} | {:>16} |'.format(
                class_school,
                number_of_students
            )
        )
        print(line)

# Вывод списка команд
elif command == "help":
    print("list - вывод всех классов в школе")
    print("help - вывод списка команд")
    print("change - изменение количества учениклв в классе")
    print("sum - вывод общего количества учеников в школе")
```

```
            print("remove - удаление класса")
            print("add - добавление нового класса")
            print("exit - завершение работы программы")

        # Действие при неверной команде
        else:
            print(f"Неизвестная команда {command}", file=sys.stderr)
```

Приложение 6. Исходный код:

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

if __name__ == '__main__':
    line = {1: 'один', 2: 'два', 3: 'три', 4: 'четыре'}

    new_line = {}
    for key, value in line.items():
        new_line[value] = key

    print(new_line)
```

Приложение 7. Исходный код:

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

if __name__ == '__main__':
    u = set('cehnfkxbprsg')

    a = {'c', 'e', 'h', 'n'}
    b = {'e', 'f', 'k', 'n', 'x'}
    c = {'b', 'c', 'h', 'p', 'r', 's'}
    d = {'b', 'e', 'g'}

    no_b = u.difference(b)

    x = a.difference(b).intersection(c.union(d))
    y = a.intersection(no_b).union(c.difference(d))

    print(x)
    print(y)
```

Приложение 8. Исходный код:

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

import sys


if __name__ == '__main__':
    # Список рейсов
    flights = []

    while True:
        command = input(">>> ").lower()
```

```
# Команда завершения
if command == 'exit':
    break

# Команда добавления рейса
elif command == 'add':
    destination = input("Пункт назначения: ").lower()
    number = int(input("Номер рейса: "))
    aircraft_type = input("Тип самолета: ").lower()

    flight = {
        'destination': destination,
        'number': number,
        'aircraft_type': aircraft_type
    }

    # Добавить словарь в список
    flights.append(flight)
    # Сортировка списка
    if len(flights) > 1:
        flights.sort(key=lambda item: item.get('destination', ''))

# Вывод всех рейсов
elif command == 'list':
    line = '+-{}-+-{}-+-{}-+'.format(
        '-' * 30,
        '-' * 15,
        '-' * 20
    )
    print(line)
    print(
        '| {:^30} | {:^15} | {:^20} |'.format(
            "Пункт назначения",
            "Номер рейса",
            "Тип самолета"
        )
    )
    print(line)

    for flight in flights:
        print(
            '| {:>30} | {:>15} | {:>20} |'.format(
                flight.get('destination', ''),
                flight.get('number', ''),
                flight.get('aircraft_type', '')
            )
        )

    print(line)

# Вывод рейсов с данным типом самолета
elif command == 'check_flight':
```

```
        key_aircraft = input("Введите тип самолета: ").lower()
        count = 0

        # Проверка, есть ли такой тип самолета
        for flight in flights:
            if flight.get("aircraft_type") == key_aircraft:
                count += 1
                print(
                    f"Вам подходит рейс {flight.get('number', '')}"
                    f" который направляется в "
                    f"{flight.get('destination', '')}"
                )

        # Проверка, если нет такого типа самолета
        if count == 0:
            print(
                "Похоже, рейсов с данным типом самолета нет,"
                " но вы можете посмотреть другие рейсы"
            )

    # Справка о работе с программой
    elif command == 'help':
        print("Список команд:\n")
        print("add - добавить рейс")
        print("list - вывести список рейсов")
        print("exit - завершить работу с программой")
        print("help - вывести список команд")
        print(
            "check_flight - проверить, "
            "есть ли рейс с данным типом самолета"
        )

    else:
        print(f"Неизвестная команда {command}", file=sys.stderr)
```