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

Кафедра инфокоммуникаций

Отчет по лабораторной работе №8

Работа со словарями в языке Python.

по дисциплине «Технологии программирования и алгоритмизации»

Выполнил студент группы ИВТ-б-о-20-1
Лысенко И.А. « »_____20__г.
Подпись студента_____
Работа защищена « »_____20__г.
Проверила Воронкин Р.А. _____
(подпись)

Ставрополь 2020

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

Указания к работе:

Ход работы:

1. Был создан следующий репозиторий: https://github.com/IsSveshuD/lab_8
2. Был проработан пример.

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

import ...

if __name__ == '__main__':
    workers = []

    while True:
        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)
```

Рисунок 1 – Пример

3. Было выполнено следующее 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.

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


import sys

if __name__ == '__main__':
    workers = []

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

        if command == 'exit':
            break

        elif command == 'add':

            start = input("Название начального пункта маршрута? ")
            finish = input("Название конечного пункта маршрута? ")
            number = int(input("Номер маршрута? "))

            worker = {
                'start': start,
                '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)
```

Рисунок 2 – Задание 1

Ответы на вопросы

1. Что такое словари в языке Python?

Словарь – это изменяемый(как список) неупорядоченный (в отличие от строк, списков и кортежей) набор элементов "ключ: значение".

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

Да, может.

3. Какие методы обхода словарей Вам известны?

Метод перебора элементов в цикле for, методы clear(), copy(), fromkeys(), get(), pop(), popitem(), setdefault(), update(), items(), keys() и values().

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

Методом get(), keys() и values() можно получить значение словаря по ключу.

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

Методом setdefault (), fromkeys() можно получить значение словаря по ключу.

6. Что такое словарь исключений?

Исключения (exceptions) — ещё один тип данных в python. Исключения необходимы для того, чтобы сообщать программисту об ошибках. Самый простой пример исключения — деление на ноль.

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. Пример:

```
employee_numbers = [3, 12, 17, 20]
employee_names = ["Алёна", "Александр", "Иван", "Юля"]


zipped_values = zip(employee_names, employee_numbers)
zipped_list = list(zipped_values)
```

print(zipped_list)

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.

Вывод: Были приобретены и проработаны навыки по работе со словарями при написании программ с помощью языка программирования Python версии 3.x, а также методы dstetime() и zip().