

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 «Программирование на Python»

Вариант 7

Выполнил:
Зармухамбетов Булат Эльдарович
2 курс, группа ИВТ-б-о-22-1,
09.03.01 «Информатика и
вычислительная техника»,
направленность (профиль)
«Программное обеспечение средств
вычислительной техники и
автоматизированных систем», очная
форма обучения

(подпись)

Руководитель практики:
Воронкин Р. А.

(подпись)

Отчет защищен с оценкой _____ Дата защиты _____

Ставрополь, 2023

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.

Пример 1. Использовать словарь, содержащий следующие ключи: фамилия и инициалы работника; название занимаемой должности; год поступления на работу. Написать программу, выполняющую следующие действия:

- 1) ввод с клавиатуры данных в список, состоящий из заданных словарей;
- 2) записи должны быть размещены по алфавиту; вывод на дисплей фамилий работников, чей стаж работы в организации превышает значение, введенное с клавиатуры;
- 3) если таких работников нет, вывести на дисплей соответствующее сообщение.

Листинг программы:

```
#!/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 счётчик.
    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 счётчик 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('exit - завершить работу с программой.')

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

```

```

>>> help
Список команд:

add - добавить работника;
list - вывести список работников;
select <стаж> - запросить работников со стажем;
exit - завершить работу с программой.
>>> list
+-----+-----+-----+-----+
| № |           Ф.И.О.           |      Должность      |   Год   |
+-----+-----+-----+-----+
>>> select 7
Работники с заданным стажем не найдены
>>> add
Фамилия и инициалы? Зармухамбетов Б.Э.
Должность? Директор
Год поступления? 2015
>>> list
+-----+-----+-----+-----+
| № |           Ф.И.О.           |      Должность      |   Год   |
+-----+-----+-----+-----+
|  1 | Зармухамбетов Б.Э.       |      Директор      |   2015   |
+-----+-----+-----+-----+
>>> select 7
      1: Зармухамбетов Б.Э.
>>>

```

Рисунок 1. Результат №1

```

>>> add
Фамилия и инициалы? Зармухамбетов Б.Э.
Должность? Зам. директора
Год поступления? 2017
>>> select 3
      1: Зармухамбетов Б.Э.
>>>

```

Рисунок 2. Результат №2

Задание 1. 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.

Листинг программы:

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

if __name__ == '__main__':
    school = {
        "1a": 25,
        "1b": 28,
        "2b": 30,
        "6a": 24,
        "7v": 27
    }

    # В классе "1a" изменилось количество учащихся
    school["1a"] = 26

    # В школе появился новый класс
    school["3c"] = 29

    # В школе был расформирован (удален) класс "7v"
    del school["7v"]

    # Вычислим суммарное количество учащихся в школе
    total = sum(school.values())

    print(f"Суммарное количество учащихся в школе: {total}")
```

```
C:\Users\User\PycharmProjects\Python_laba_9\venv
Суммарное количество учащихся в школе: 137

Process finished with exit code 0
```

Рисунок 3. Результат программы

Задание 2. 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.

Листинг программы:

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

```

if __name__ == '__main__':
    # Создание исходного словаря
    original_dict = {1: 'one', 2: 'two', 3: 'three'}

    # Применение метода items() для получения объекта dict_items
    dict_items = original_dict.items()

    # Создание "обратного" словаря
    reversed_dict = {value: key for key, value in dict_items}

    print(reversed_dict)

```

```

C:\Users\User\PycharmProjects\Python_laba_9\
{'one': 1, 'two': 2, 'three': 3}

Process finished with exit code 0

```

Рисунок 4. Результат программы

Индивидуальное задание. Использовать словарь, содержащий следующие ключи: название пункта назначения; номер поезда; время отправления. Написать программу, выполняющую следующие действия: ввод с клавиатуры данных в список, состоящий из словарей заданной структуры; записи должны быть упорядочены по времени отправления поезда; вывод на экран информации о поездах, направляющихся в пункт, название которого введено с клавиатуры; если таких поездов нет, выдать на дисплей соответствующее сообщение.

Листинг программы:

```

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

import sys

if __name__ == '__main__':
    # Создание списка словарей для хранения информации о поездах.
    trains = []

    # Организация бесконечного цикла 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 поезде.

```

```

destination = input('Название пункта назначения? ')
number = input('Номер поезда? ')
departure_time = input('Время отправления? ')

# Создать словарь.
train = {
    'destination': destination,
    'number': number,
    'departure_time': departure_time
}

# Добавить словарь в список.
trains.append(train)

# Отсортировать список по времени отправления поезда.
trains.sort(key=lambda item: item.get('departure_time', ''))

elif command == 'list':
    # Заголовок таблицы.
    line = '+-{}-+{}-+{}-+'.format(
        '-' * 20,
        '-' * 15,
        '-' * 20
    )
    print(line)
    print(
        '| {:^20} | {:^15} | {:^20} |'.format(
            "Пункт назначения",
            "Номер поезда",
            "Время отправления"
        )
    )
    print(line)

    # Вывести информацию о поездах.
    for idx, train in enumerate(trains, 1):
        print(
            '| {:<20} | {:^15} | {:^20} |'.format(
                train.get('destination', ''),
                train.get('number', ''),
                train.get('departure_time', '')
            )
        )
    print(line)

elif command.startswith('select '):
    # Получить название пункта назначения из команды.
    parts = command.split(' ', maxsplit=1)
    destination = parts[1]

    # Поиск поездов с заданным пунктом назначения.
    selected_trains = [train for train in trains if
train['destination'] == destination]

    if selected_trains:
        # Вывести информацию о найденных поездах в виде таблицы.
        line = '+-{}-+{}-+{}-+'.format(
            '-' * 20,
            '-' * 15,
            '-' * 20
        )
        print(line)
        print(
            '| {:^20} | {:^15} | {:^20} |'.format(

```

```

        "Пункт назначения",
        "Номер поезда",
        "Время отправления"
    )
    )
    print(line)
    for train in selected_trains:
        print(
            '| {:<20} | {:^15} | {:^20} |'.format(
                train.get('destination', ''),
                train.get('number', ''),
                train.get('departure_time', '')
            )
        )
    print(line)
else:
    print(f'Поездов в пункт "{destination}" не найдено')

elif command == 'help':
    # Вывести справку о работе с программой.
    print('Список команд:\n')
    print('add - добавить информацию о поезде;')
    print('list - вывести список всех поездов;')
    print('select <пункт_назначения> - запросить информацию о поездах
в заданном пункте назначения;')
    print('exit - завершить работу с программой.')
else:
    print(f'Неизвестная команда "{command}"!', file=sys.stderr)

```

Рисунок 5. Тест №1

```

Номер поезда? 101
Время отправления? 11:00
>>> add
Название пункта назначения? stavorpol
Номер поезда? 202
Время отправления? 12:00
>>> list
+-----+-----+-----+
| Пункт назначения | Номер поезда | Время отправления |
+-----+-----+-----+
| stavorpol       | 101         | 11:00             |
| stavorpol       | 202         | 12:00             |
+-----+-----+-----+
>>> select stavorpol
+-----+-----+-----+
| Пункт назначения | Номер поезда | Время отправления |
+-----+-----+-----+
| stavorpol       | 101         | 11:00             |
| stavorpol       | 202         | 12:00             |
+-----+-----+-----+
>>> add
Название пункта назначения? saint p
Номер поезда? 228
Время отправления? 14:00
>>> list
+-----+-----+-----+
| Пункт назначения | Номер поезда | Время отправления |
+-----+-----+-----+
| stavorpol       | 101         | 11:00             |
| stavorpol       | 202         | 12:00             |
| saint p         | 228         | 14:00             |
+-----+-----+-----+
>>> select saint p
+-----+-----+-----+
| Пункт назначения | Номер поезда | Время отправления |
+-----+-----+-----+
| saint p         | 228         | 14:00             |
+-----+-----+-----+

```

Рисунок 6. Тест №2.1

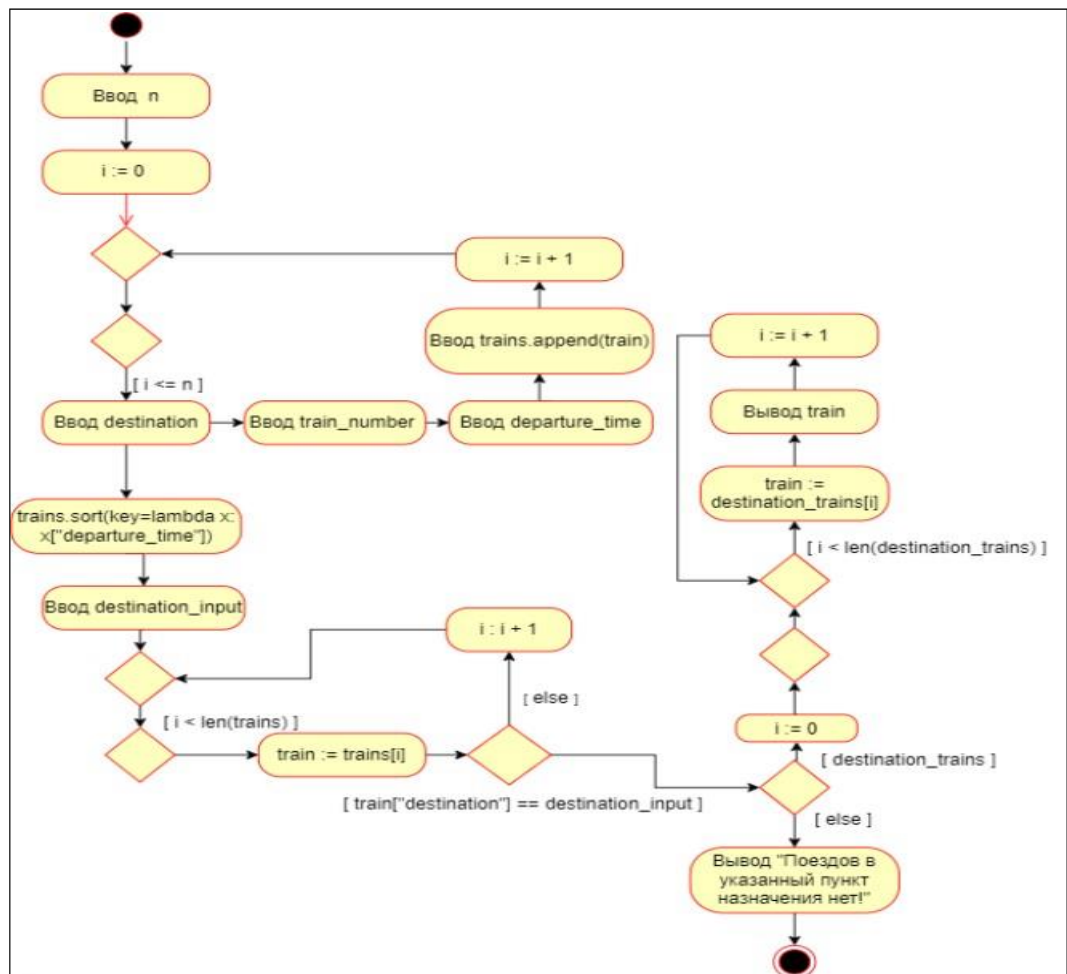


Рисунок 8. UML-диаграмма для индивидуального задания

Вывод: в ходе выполнения данной лабораторной работы были приобретены навыки взаимодействия со словарями при написании программ с помощью языка программирования Python версии 3.x.

Ответы на контрольные вопросы

1. Словари в Python - это неупорядоченные коллекции объектов, которые хранятся в парах ключ-значение.

2. Да, функция `len()` может быть использована для определения количества элементов в словаре.

3. Методы обхода словарей включают использование цикла `for` для перебора ключей или значений, методов `keys()`, `values()` и `items()`.

4. Значения из словаря можно получить по ключу с помощью оператора доступа к элементам `[]` или метода `get()`.

5. Значение в словаре по ключу можно установить с помощью оператора доступа к элементам `[]` или метода `setdefault()`.

6. Словарь включений (dictionary comprehensions) - это способ создания нового словаря на основе итерации по другому объекту.

7. Функция `zip()` используется для объединения элементов из нескольких итерируемых объектов в кортежи. Например:

```
a = [1, 2, 3]
```

```
b = ['a', 'b', 'c']
```

```
zipped = zip(a, b)
```

```
print(list(zipped)) # [(1, 'a'), (2, 'b'), (3, 'c')]
```

8. Модуль `datetime` обладает функционалом для работы с датой, временем, интервалами времени, форматированием и парсингом даты и времени, а также для работы с часовыми поясами. Некоторые из его классов и методов включают `datetime`, `date`, `time`, `timedelta`, `strftime()`, `strptime()` и многие другие.