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 практической работе №2.6

Дисциплины «Программирование на Python»

Выполнил:

Пустяков Андрей Сергеевич

2 курс, группа ИВТ-б-о-22-1,

09.03.01 «Информатика и вычислительная техника (профиль) «Программное обеспечение средств вычислительной
техники и автоматизированных систем», очная форма обучения

_____

(подпись)

Руководитель практики:

Воронкин Р. А. кандидат технических наук, доцент, доцент кафедры инфокоммуникаций

_____

(подпись)

Ставрополь, 2023 г.

Тема: Работа со словарями в языке Python.

Цель: приобрести навыки по работе со словарями при написании программ на языке программирования Python версии 3.x.

Ход работы:

Создание репозитория, клонирование репозитория, организация репозитория согласно модели ветвления «git-flow» (рис. 1).

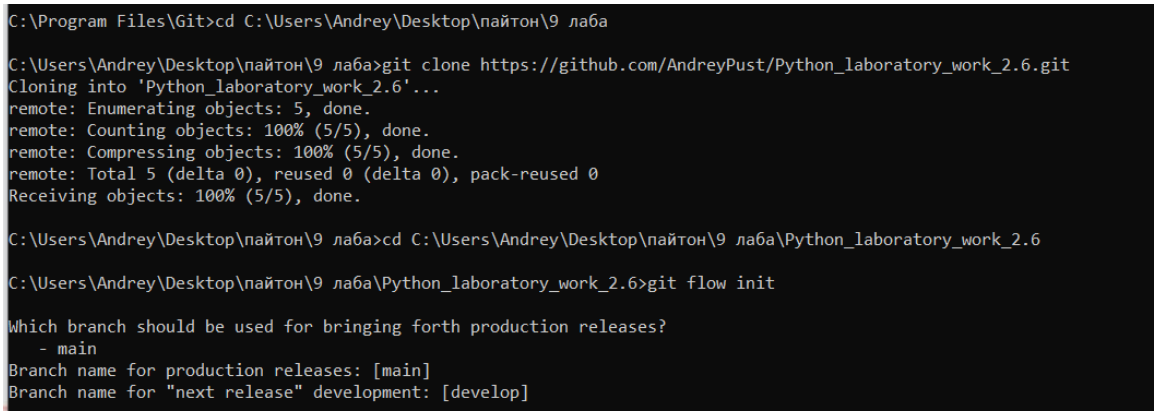

Рисунок 1 – Организация модели ветвления «git-flow».

Проработка примеров лабораторной работы:

Пример 1.

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

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

Код программы примера №1 и результаты работы программы с различными исходными данными (рис. 2, 3).

```
#!/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)
```

Рисунок 2 – Код программы примера №1.

```
>>> 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? Пустяков А.С.
Должность? директор
Год поступления? 2022
>>> add
Фамилия и инициалы? Иванов И.И.
Должность? менеджер
Год поступления? 2022
>>> add
Фамилия и инициалы? Сергеев П.С.
Должность? водитель
Год поступления? 2023
>>> add
Фамилия и инициалы? Смирнов И.И.
Должность? руководитель группы
Год поступления? 2024
>>> list
+------+----------------------------------+----------------------+----------+
|  №   |             Ф.И.О.               |      Должность       |   Год    |
+------+----------------------------------+----------------------+----------+
```

```
>>> add
Фамилия и инициалы? Смирнов И.И.
Должность? руководитель группы
Год поступления? 2024
>>> list
+------+----------------------------------+----------------------+----------+
|  №   |             Ф.И.О.               |      Должность       |   Год    |
+------+----------------------------------+----------------------+----------+
|    1 | Иванов И.И.                      | менеджер             |     2022 |
|    2 | Пустяков А.С.                    | директор             |     2022 |
|    3 | Сергеев П.С.                     | водитель             |     2023 |
|    4 | Смирнов И.И.                     | руководитель группы  |     2024 |
+------+----------------------------------+----------------------+----------+
```

```
Год поступления? 2000
>>> add
Фамилия и инициалы? Петрова Г.И.
Должность? кассир
Год поступления? 2002
>>> add
Фамилия и инициалы? Коршунов И.В.
Должность? кладовщик
Год поступления? 2003
>>> add
Фамилия и инициалы? Нужнов В.М.
Должность? менеджер
Год поступления? 2000
>>> list
+------+--------------------------------+----------------------+----------+
|  №   |             Ф.И.О.             |      Должность       |   Год    |
+------+--------------------------------+----------------------+----------+
|    1 | Коршунов И.В.                  | кладовщик            |     2003 |
|    2 | Нужнов В.М.                    | менеджер             |     2000 |
|    3 | Петрова Г.И.                   | кассир               |     2002 |
|    4 | Пустяков А.С.                  | директор             |     2000 |
+------+--------------------------------+----------------------+----------+
>>> select 23
   1: Нужнов В.М.
   2: Пустяков А.С.
>>> exit
```

Рисунок 3 – Результаты работы программы примера №1.

Необходимо решить задачу: создать словарь, связав его с переменной «school», наполнить его данными, которые отражали бы количество учащихся в разных классах. Далее внести в словарь изменения: в одном из классов изменилось количество учащихся, в школе появился новый класс, в школе был расформирован какой-либо класс. Также необходимо вычислить общее количество учащихся в школе.

Код программы решения данной задачи и результаты работы программы (рис. 4, 5).

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

# Необходимо решить задачу: создать словарь, связав его с переменной
# «school», наполнить его данными, которые отражали бы количество
# учащихся в разных классах. Далее внести в словарь изменения: в одном
# из классов изменилось количество учащихся, в школе появился новый
# класс, в школе был расформирован какой-либо класс. Также необходимо
# вычислить общее количество учащихся в школе.
```

```
import sys


if __name__ == '__main__':
    # Добавим некоторое количество классов в школе.
    # Создать словарь (ключ-класс, учащиеся-значение).
    school = {'1А': 23,
              '1Б': 24,
              '1Н': 25
              }

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

        # Выполнить действие в соответствие с командой.
        if command == 'exit':
            break

        # Добавление классов и учащихся.
        elif command == 'add':
            # Запросить данные о классе в школе.
            school_class = input("Введите номер класса и букву: ")
            number_students = int(input("Введите количесво учащихся: "))
            # Добавить в словарь.
            school[school_class] = number_students

        # Изменить учащихся в классе.
        elif command == 'change':
            # Запросить данные о классе в школе.
            school_class = input("Введите класс в котором хотите изменить
кол-во учащихся: ")
            number_students = int(input("Введите новое кол-во учащихся: "))
            # Изменить словарь.
            school[school_class] = number_students

        # Удалить класс.
        elif command == 'del':
            # Запросить данные о классе в школе.
            school_class = input("Введите класс, который хотите
расформировать: ")
            # Удалить ключ этого класса.
            del school[school_class]

        # Получить кол-во учащихся в школе.
        elif command == 'sum':
            print("Сумма всех учащихся в школе = ", sum(school.values()))

        # Вывести справку о работе с программой.
        elif command == 'help':
            print("Список команд:\n")
            print("add -    добавить новый класс;")
            print("help -   отобразить справку;")
            print("exit -   завершить работу с программой.")
            print("change - изменить кол-во учащихся в школе.")
            print("del -    удалить класс.")
            print("sum -    получить кол-во учащихся в школе.")

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

Рисунок 4 – Код программы первой задачи.

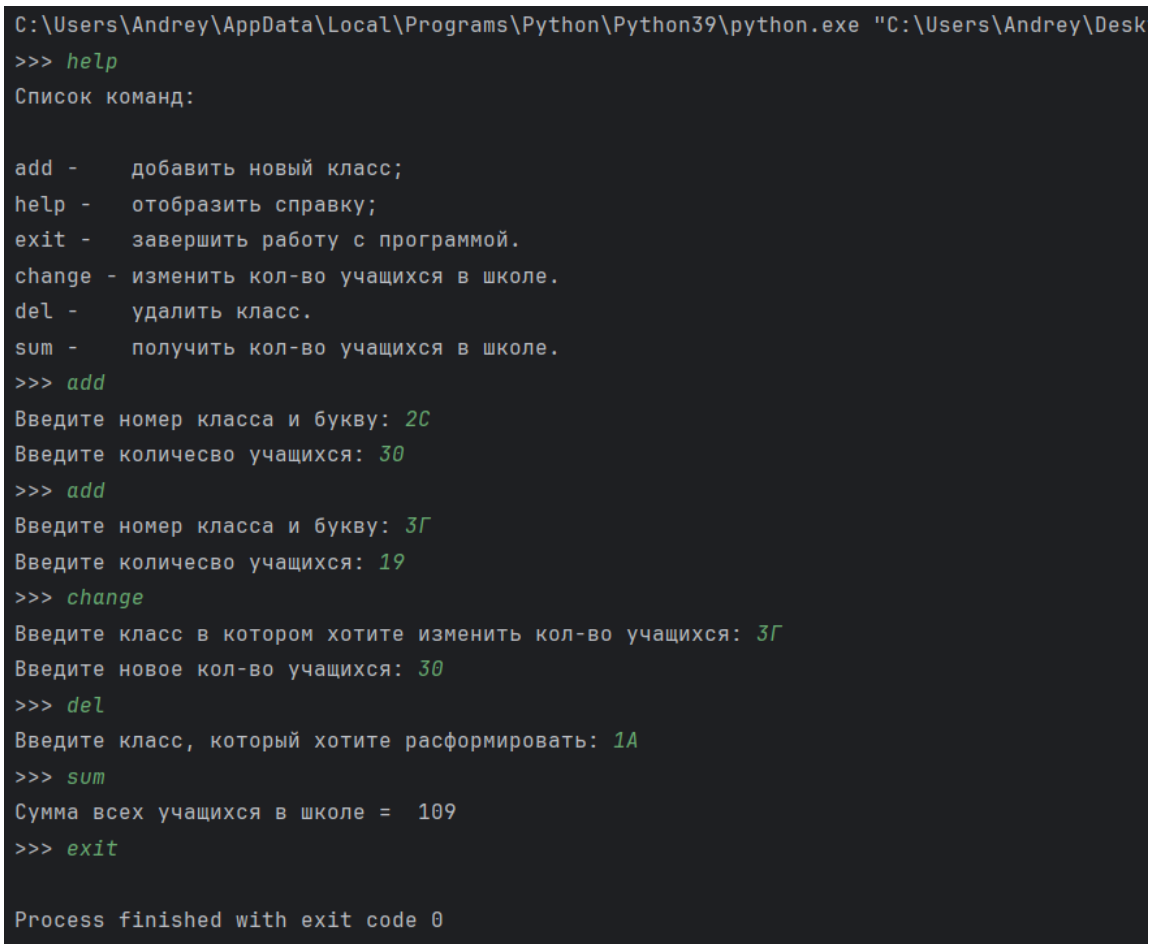

Рисунок 5 – Результаты работы программы первой задачи.

Необходимо решить задачу: необходимо создать словарь, в котором ключами являются числа а значениями строки. Необходимо применить к такому словарю метод «items()», с помощью полученного объекта «dict_items» создать словарь, «обратный» исходному (ключи – это строки, а значения это - числа).

Код программы решения задачи №2 и результаты работы программы (рис. 6, 7).

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

# Необходимо решить задачу: необходимо создать словарь, в котором
# ключами являются числа а значениями строки. Необходимо применить
# к такому словарю метод «items()», с помощью полученного объекта
# «dict_items» создать словарь, «обратный» исходному (ключи - это
# строки, а значения это - числа).

if __name__ == '__main__':
    # Первый словарь.
    first_dict = {1: 'one', 2: 'two', 3: 'three', 4: 'four', 5: 'five',
                  6: 'six', 7: 'seven', 8: 'eight', 9: 'nine', 10: 'ten'}
    print("Первый словарь, в котором ключи - это числа, а значения - строки:", first_dict)

    # Создадим второй словарь, обратный исходному.
    second_dict = {}
    for j, i in first_dict.items():
        second_dict[i] = j
    print("Второй словарь, обратный исходному:", second_dict)
```

Рисунок 6 – Код программы второй задачи.

```
C:\Users\Andrey\AppData\Local\Programs\Python\Python39\python.exe "C:\Users\Andrey\Desktop\пайтон\9 лаба\Py
Первый словарь, в котором ключи - это числа, а значения - строки: {1: 'one', 2: 'two', 3: 'three', 4: 'four
Второй словарь, обратный исходному: {'one': 1, 'two': 2, 'three': 3, 'four': 4, 'five': 5, 'six': 6, 'seven
```

Рисунок 7 – Результаты работы программы второй задачи.

Выполнение индивидуального задания:

Необходимо 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 (Вариант 26 (7)).

Код программы решения индивидуального задания и результаты работы программы (рис. 8, 9).

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

# Необходимо 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.

import sys


if __name__ == '__main__':
    # Список пунктов (станций).
    stations = []

    # 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 пункте.
            name = input("Название пункта: ")
            # Создать словарь.
            station = {'name': name}
            print("Добавить поезд? n/y")
            cucle = input()
            if cucle == 'n':
                station['train'] = 'Поездов нет'
                station['time'] = ' '
            else:
                train = input("Номер поезда: ")
                dep_time = input("Время отправления поезда: ")
                # Добавить в словарь поезд и время.
                station['train'] = train
                station['time'] = dep_time

            # Добавить словарь в список.
            stations.append(station)

            # Отсортировать список в случае необходимости по времени поезда.
            if len(stations) > 1:
                stations.sort(key=lambda item: item.get('time', ''))

        elif command.startswith('info '):
            # Инициализировать счетчик.
            count = 0

            # Разбить команду на части для выделения названия пункта.
            name_station = command.split(' ', maxsplit=1)
            name_station = name_station[1]
            for station in stations:
                if station.get('name') == name_station:
                    count += 1
                    print("Номер поезда пункта: ", station.get('train'),
                          "Время отправления: ", station.get('time'))
```

```
            # Если счетчик равен 0, то станции не найдены.
            if count == 0:
                print("Станции не найдены.")

        elif command == 'help':
            # Вывести справку о работе с программой.
            print("Список команд:\n")
            print("add - добавить станцию;")
            print("info <станция> - запросить информацию о станции;")
            print("help - отобразить справку;")
            print("exit - завершить работу с программой.")

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

Рисунок 8 – Код программы индивидуального задания.

```
>>> help
Список команд:

add - добавить станцию;
info <станция> - запросить информацию о станции;
help - отобразить справку;
exit - завершить работу с программой.
>>> add
Название пункта: первозельск
Добавить поезд? n/y
y
Номер поезда: 001
Время отправления поезда: 10:00
>>> add
Название пункта: второвинск
Добавить поезд? n/y
y
Номер поезда: 002
Время отправления поезда: 11:00
>>> add
Название пункта: третьинск
Добавить поезд? n/y
n
>>> info первозельск
Номер поезда пункта:  001 Время отправления:  10:00
>>> info третьинск
Номер поезда пункта:  Поездов нет Время отправления:
>>> exit
```

Рисунок 9 – Результаты работы программы индивидуального задания.

UML-диаграмма деятельности индивидуального задания (рис. 10).

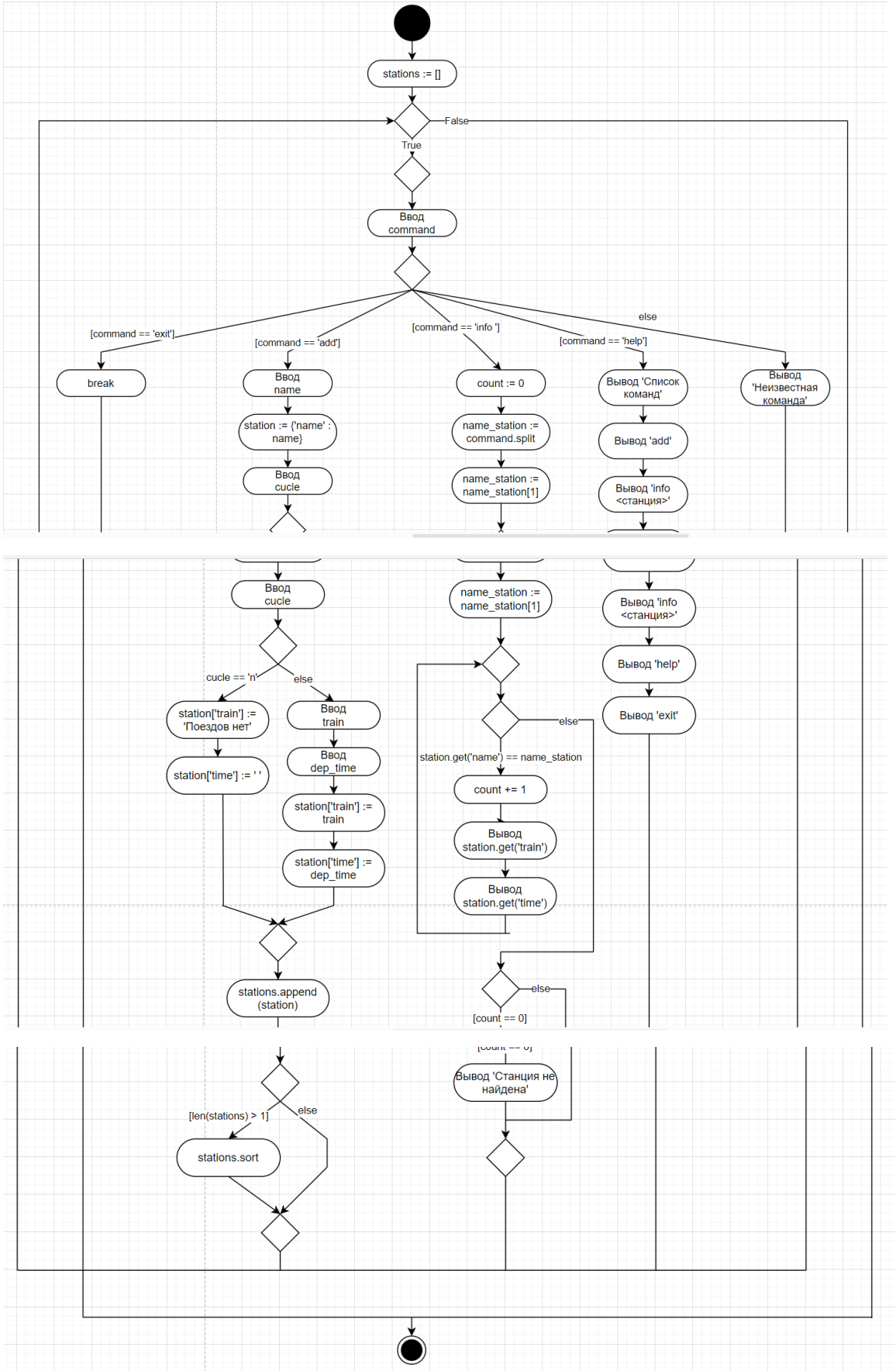

Рисунок 10 - UML-диаграмма деятельности индивидуального задания.

Ответы на контрольные вопросы:

1. Словарь представляет собой структуру данных (которая ещё называется ассоциативный массив), предназначенную для хранения произвольных объектов с доступом по ключу. Данные в словаре хранятся в формате ключ – значение. В списке доступ к элементам осуществляется по индексу, который представляет собой целое неотрицательное число. В словаре аналогом индекса является ключ, при этом ответственность за его формирование ложится на программиста. В языке программирования Python словари (тип «dict») представляют собой еще одну разновидность структур данных наряду со списками и кортежами. Словарь – это изменяемый (как список) неупорядоченный (в отличие от строк, списков и кортежей) набор элементов "ключ: значение". ("Неупорядоченный" – значит, что последовательность расположения пар не важна. Язык программирования ее не учитывает, в следствие чего обращение к элементам по индексам невозможно).

2. Функция «len()» может быть использована при работе со словарями, она выводит количество ключей в словаре.

3. Для обхода словаря можно воспользоваться циклом «for» с оператором «in», а также с методами «keys()» (Все ключи), «items()» (Превращает пары ключ-значение в отдельные списки в кортеже) или «values()» (Все значения).

4. Объект словаря обладает функцией «get()», которой можно пользоваться для доступа к элементам словаря. Ее нужно добавлять к словарю через точку и затем передавать название ключа как аргумент функции.

5. Чтобы в словарь добавить значение ключу, можно: map = {1: 'one', 2: 'two', 3: 'three'}; nums2[4] = 'four'; nums.setdefault(4, 'four').

6. Словарное включение работает по принципу: {ключ: значение for элемент in итерируемый_объект}. Словарное включение аналогично списковым включениям, то есть сокращает длину кода, позволяя заполнить словарь всего одной строкой кода.

7. Функция «zip()» в Python позволяет объединять элементы из нескольких итерируемых объектов (списков, строк) в кортежи. Каждый кортеж содержит элементы на соответствующих позициях из каждого переданного итерируемого объекта. Если объекты имеют разную длину, то «zip()» остановится, когда закончится итерация на самом коротком объекте. Например: first = [2, 9, 18, 28]; second = ["Smth1", "Hey!", "Good day", "2htmS"]; zip(first, second) ==> [(2, "Smth1"),(9, "Hey!"),(18, "Good day"),(28, "2htmS")].

8. Модуль позволяет управлять датами и временем, представляя их в таком виде, в котором пользователи смогут их понимать. «datetime» включает различные компоненты (Можно переставлять местами вывод месяца, года и дня, изменять часовой пояс, указывать 12-ти или 24-чсовой формат и пр.). Так, он состоит из объектов следующих типов: «date» — хранит дату, «time» — хранит время, «datetime» — хранит дату и время.

Вывод: в ходе лабораторной работы была подробнее изучена работа со словарями в языке программирования Python. Словарь – это изменяемый (элементы можно добавлять или удалять) неупорядоченный тип данных на ряду со списками и кортежами. Синтаксис словаря представляет собой связку ключей и их значений, ключи должны быть уникальными, значения могут повторяться. Были изучены способы перебора элементов в словаре, методы преобразования словарей в другие типы данных и обратно, способы извлечения значений ключей в словаре. Были изучены основные методы словаря для его очистки, копирования, редактирования и т.д. Были изучены и рассмотрены словарные включения, аналогичные списковым включениям. Был изучен подробнее модуль «datetime».