

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 №2.6
дисциплины «Основы кроссплатформенного программирования»

Выполнила:
Мурашко Анастасия Юрьевна
1 курс, группа ИТС-б-о-22-1,
11.03.02 «Инфокоммуникационные
технологии и системы связи»,
направленность (профиль)
«Инфокоммуникационные системы и
сети», очная форма обучения

(подпись)

Руководитель практики:
Воронкин Р. А., доцент кафедры
инфокоммуникаций

(подпись)

Отчет защищен с оценкой _____ Дата защиты _____

Ставрополь, 2023 г.

Тема: Работа со словарями в языке Python

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

Ход работы:

Вариант №10

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', ''))
```

Рисунок 1. Задание 1

```

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

```

Рисунок 2. Задание 1

```

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':

```

Рисунок 3. Задание 1

```
71 # Проверить сведения работников из списка.
72 for worker in workers:
73     if today.year - worker.get('year', today.year) >= period:
74         count += 1
75         print(
76             '{:>4}: {}'.format(count, worker.get('name', ''))
77         )
78 # Если счетчик равен 0, то работники не найдены.
79 if count == 0:
80     print("Работники с заданным стажем не найдены.")
81 elif command == 'help':
82     # Вывести справку о работе с программой.
83     print("Список команд:\n")
84     print("add - добавить работника;")
85     print("list - вывести список работников;")
86     print("select <стаж> - запросить работников со стажем;")
87     print("help - отобразить справку;")
88     print("exit - завершить работу с программой.")
89 else:
90     print(f"Неизвестная команда {command}", file=sys.stderr)

if __name__ == '__main__': while True: else
```

main x

```
>>> 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? Зубарев В.В
Должность? Хореограф
Год поступления? 2000
>>> list
```

No	Ф.И.О.	Должность	Год
1	Зубарев В.В	Хореограф	2000

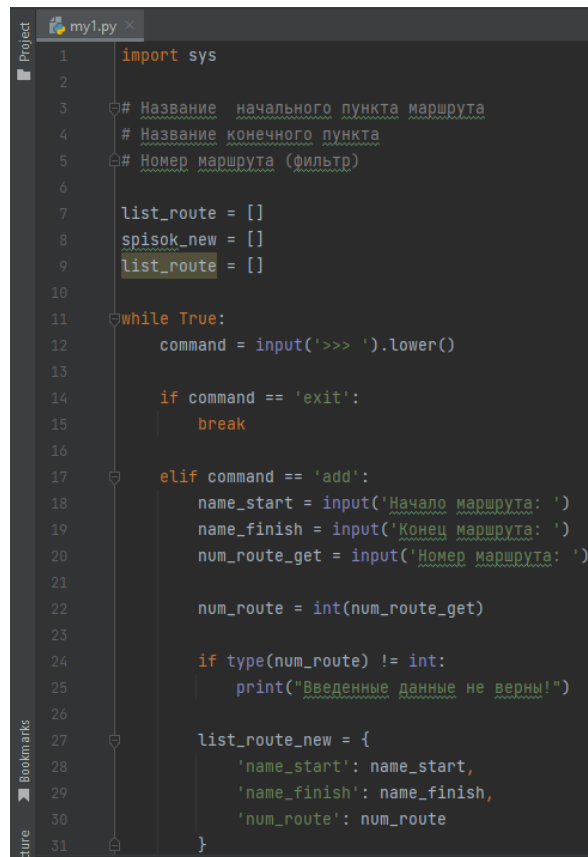
Рисунок 4. Задание 1

Индивидуальные задания

Задание 1

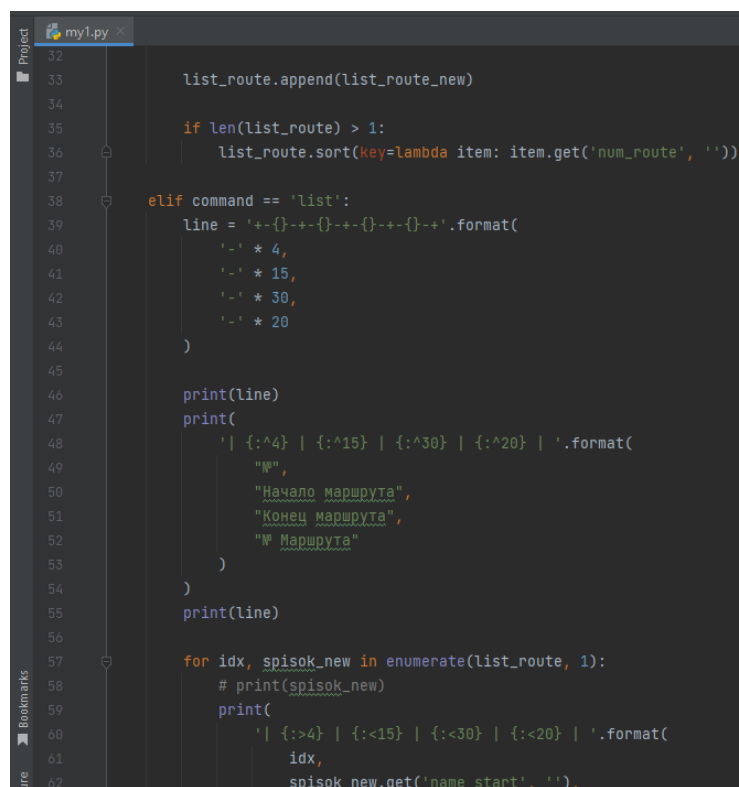
Использовать словарь, содержащий следующие ключи: название начального пункта маршрута; название конечного пункта маршрута; номер маршрута. Написать программу, выполняющую следующие действия: ввод с клавиатуры данных в список, состоящий из словарей заданной структуры; записи должны быть упорядочены по номерам маршрутов; вывод на экран информации о маршрутах, которые начинаются или оканчиваются в пункте,

название которого введено с клавиатуры; если таких маршрутов нет, выдать на дисплей соответствующее сообщение.



```
1 import sys
2
3 # Название начального пункта маршрута
4 # Название конечного пункта
5 # Номер маршрута (фильтр)
6
7 list_route = []
8 spisok_new = []
9 list_route = []
10
11 while True:
12     command = input('>>> ').lower()
13
14     if command == 'exit':
15         break
16
17     elif command == 'add':
18         name_start = input('Начало маршрута: ')
19         name_finish = input('Конец маршрута: ')
20         num_route_get = input('Номер маршрута: ')
21
22         num_route = int(num_route_get)
23
24         if type(num_route) != int:
25             print("Введенные данные не верны!")
26
27         list_route_new = {
28             'name_start': name_start,
29             'name_finish': name_finish,
30             'num_route': num_route
31         }
```

Рисунок 1. Задание 1 (1)



```
32
33 list_route.append(list_route_new)
34
35 if len(list_route) > 1:
36     list_route.sort(key=lambda item: item.get('num_route', ''))
37
38 elif command == 'list':
39     line = '+-{}-+-{}-+-{}-+-{}-+'.format(
40         '-' * 4,
41         '-' * 15,
42         '-' * 30,
43         '-' * 20
44     )
45
46     print(line)
47     print(
48         '| {:^4} | {:^15} | {:^30} | {:^20} |'.format(
49             "№",
50             "Начало маршрута",
51             "Конец маршрута",
52             "№ Маршрута"
53         )
54     )
55     print(line)
56
57 for idx, spisok_new in enumerate(list_route, 1):
58     # print(spisok_new)
59     print(
60         '| {:>4} | {:<15} | {:<30} | {:<20} |'.format(
61             idx,
62             spisok_new.get('name_start', ''),
```

Рисунок 2. Задание 1 (2)

```

63         spisok_new.get('name_finish', ''),
64         spisok_new.get('num_route', 0)
65     )
66 )
67
68 print(line)
69
70
71 elif command == 'route':
72     route_sear = input('Введите пункт маршрута: ')
73     search_route = []
74     for route_sear_itme in list_route:
75         if route_sear == route_sear_itme['name_start']:
76             search_route.append(route_sear_itme)
77         if route_sear == route_sear_itme['name_finish']:
78             search_route.append(route_sear_itme)
79
80     if len(search_route) > 0:
81         line_new = '++{}--{}--{}--{}--+'.format(
82             '-' * 4,
83             '-' * 15,
84             '-' * 30,
85             '-' * 20
86         )
87         print(line_new)
88
89         print(
90             '| {:^4} | {:^15} | {:^30} | {:^20} | '.format(
91                 "№",
92                 "Начало маршрута",
93                 "Конец маршрута"

```

Рисунок 3. Задание 1 (3)

```

93         "Конец маршрута",
94         "№ Маршрута"
95     )
96 )
97 print(line_new)
98
99 for idx_new, spisok_new_new in enumerate(search_route, 1):
100     print(
101         '| {:>4} | {:<15} | {:<30} | {:<20} | '.format(
102             idx_new,
103             spisok_new_new.get('name_start', ''),
104             spisok_new_new.get('name_finish', ''),
105             spisok_new_new.get('num_route', '')
106         )
107     )
108
109     print(line_new)
110 else:
111     print('Таких маршрутов не найдено', file=sys.stderr)
112
113
114 elif command == 'help':
115     print('Список команд:\n')
116     print('add - добавить маршрут.')
117     print('list - вывести список маршрутов.')
118     print('route <Пункты маршрутов> - запросить Начало или конечные пункты маршрутов.')
119     print('help - Справочник.')
120     print('exit - Завершить работу программы.')
121 else:
122     print(f'Команда <{command}> не существует.', file=sys.stderr)
123     print('Введите <help> для просмотра доступных команд')

```

Рисунок 4. Задание 1 (4)

```
Run: my1 x
C:\Users\User\PycharmProjects\pythonProject4\venv\Scripts\python.exe C:/Users/User/Pycha
>>> add
Начало маршрута: Владимира Нургалиева
Конец маршрута: ж/к Олимпийский
Номер маршрута: 48
>>> list
+-----+-----+-----+-----+-----+
| № | Начало маршрута | Конец маршрута | № Маршрута |
+-----+-----+-----+-----+-----+
| 1 | Владимира Нургалиева | ж/к Олимпийский | 48 |
+-----+-----+-----+-----+-----+
>>> route
Введите пункт маршрута: Владимира Нургалиева
+-----+-----+-----+-----+-----+
| № | Начало маршрута | Конец маршрута | № Маршрута |
+-----+-----+-----+-----+-----+
| 1 | Владимира Нургалиева | ж/к Олимпийский | 48 |
+-----+-----+-----+-----+-----+
>>> help
Список команд:

add - добавить маршрут.
list - вывести список маршрутов.
route <Пункты маршрутов> - запросить Начало или конечные пункты маршрутов.
help - Справочник.
exit - Завершить работу программы.
>>> exit

Process finished with exit code 0
```

Рисунок 5. Задание 1 (5)

Ответы на контрольные вопросы:

1. Что такое словари в языке Python?

Словарь (dict) представляет собой структуру данных (которая ещё называется ассоциативный массив), предназначенную для хранения произвольных объектов с доступом по ключу.

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

Да может! Функция len() возвращает длину (количество элементов) в объекте.

3. Какие методы обхода словарей Вам известны?

У словаря как класса есть метод items(), который создает особую структуру, состоящую из кортежей. Каждый кортеж включает ключ и значение:

```
>>> n = nums.items()
>>> n
dict_items([(1, 'one'), (2, 'two'), (3, 'three')])
```

Методы словаря `keys()` и `values()` позволяют получить отдельно перечни ключей и значений. Так что если, например, надо перебрать только значения или только ключи, лучше воспользоваться одним из этих методов:

```
>>> v_nums = []
>>> for v in nums.values():
...     v_nums.append(v)
...
>>> v_nums
['one', 'two', 'three']
```

Так же существуют методы `clear()`, `copy()`, `fromkeys()`, `get()`, `pop()`, `popitem()`, `setdefault()`, `update()`.

Метод `clear()` удаляет все элементы словаря, но не удаляет сам словарь. В итоге остается пустой Словарь. Метод `fromkeys()` позволяет создать словарь из списка, элементы которого становятся ключами. Применять метод можно как классу `dict`, так и к его объектам. Метод `get()` позволяет получить элемент по его ключу. Метод `pop()` удаляет из словаря элемент по указанному ключу и возвращает значение удаленной пары. Метод `popitem()` не принимает аргументов, удаляет и возвращает произвольный элемент. С помощью `setdefault()` можно добавить элемент в словарь. С помощью `update()` можно добавить в словарь другой словарь.

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

Операция `dict[key]` вернет элемент словаря `dict` с ключом `key`. Операция вызывает исключение `KeyError`, если ключ `key` отсутствует в словаре.

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

Операция `d[key] = value` добавит в словарь `dict` новый элемент - пару ключ-значение.

Если в словаре существует ключ `key` то эта операция присвоит ключу `key` новое значение `value`.

6. Что такое словарь включений?

Словарь включений аналогичен списковым включениям, за исключением того, что он создаёт объект словаря вместо списка. Как и в случае со списком, мы можем использовать условный оператор внутри словаря включения, чтобы получить только элементы словаря, удовлетворяющие заданному критерию.

7. Самостоятельно изучите возможности функции `zip()` приведите примеры ее использования.

Функция `zip()` создает итератор кортежей, который объединяет элементы каждой из переданных последовательностей `*iterables`.

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:

- `date` — хранит дату
- `time` — хранит время
- `datetime` — хранит дату и время

Вывод: В ходе выполнения лабораторной работы приобретены навыки по работе со словарями при написании программ с помощью языка программирования Python версии 3.x.