

Министерство науки и высшего образования Российской Федерации  
Федеральное государственное автономное образовательное учреждение  
высшего образования  
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт цифрового развития  
Кафедра инфокоммуникаций

**«Работа со словарями в языке Python»**

**ОТЧЕТ**  
**по лабораторной работе №9**  
**дисциплины**  
**«Основы программной инженерии»**

Выполнил:

Борсуков Владислав Олегович  
2 курс, группа ПИЖ-б-о-21-1,  
09.03.04 «Программная  
инженерия», направленность  
(профиль) «Разработка и  
сопровождение программного  
обеспечения», очная форма  
обучения

---

(подпись)

Проверил:

---

(подпись)

Отчет защищен с оценкой \_\_\_\_\_ Дата защиты \_\_\_\_\_

Ставрополь, 2022 г.

### Проработка примера из лабораторной работы:

```
1  ▶  #!/usr/bin/env python3
2      # -*- coding: utf-8 -*-
3
4      import sys
5      from datetime import date
6
7      ▶  if __name__ == '__main__':
8          # Список работников.
9          workers = []
10
11         # Организовать бесконечный цикл запроса команд.
12         while True:
13             # Запросить команду из терминала.
14             command = input(">>> ").lower()
15
16             # Выполнить действие в соответствие с командой.
17             if command == 'exit':
18                 break
19
20             elif command == 'add':
21                 # Запросить данные о работнике.
22                 name = input("Фамилия и инициалы? ")
23                 post = input("Должность? ")
24                 year = int(input("Год поступления? "))
25
26                 # Создать словарь.
27                 worker = {
28                     'name': name,
29                     'post': post,
30                     'year': year,
31                 }
32
33                 # Добавить словарь в список.
34                 workers.append(worker)
35                 # Отсортировать список в случае необходимости.
36                 if len(workers) > 1:
37                     workers.sort(key=lambda item: item.get('name', ''))
38
```

Рисунок 1.1 – Код примера

```

39 elif command == 'list':
40     # Заголовок таблицы.
41     line = '+-{}-+-{}-+-{}-+-{}-+'.format(
42         '-' * 4,
43         '-' * 30,
44         '-' * 20,
45         '-' * 8
46     )
47     print(line)
48     print(
49         '| {:^4} | {:^30} | {:^20} | {:^8} |'.format(
50             "№",
51             "Ф.И.О.",
52             "Должность",
53             "Год"
54         )
55     )
56     print(line)
57
58     # Вывести данные о всех сотрудниках.
59     for idx, worker in enumerate(workers, 1):
60         print(
61             '| {:>4} | {:<30} | {:<20} | {:>8} |'.format(
62                 idx,
63                 worker.get('name', ''),
64                 worker.get('post', ''),
65                 worker.get('year', 0)
66             )
67         )
68     print(line)
69
70 elif command.startswith('select'):
71     # Получить текущую дату.
72     today = date.today()
73
74     # Разбить команду на части для выделения номера года.
75     parts = command.split(' ', maxsplit=1)
76     # Получить требуемый стаж.
77     period = int(parts[1])
78

```

Рисунок 1.2 – Код примера

```

79      # Инициализировать счетчик.
80      count = 0
81      # Проверить сведения работников из списка.
82      for worker in workers:
83          if today.year - worker.get('year', today.year) >= period:
84              count += 1
85              print(
86                  '{:>4}: {}'.format(count, worker.get('name', ''))
87              )
88
89      # Если счетчик равен 0, то работники не найдены.
90      if count == 0:
91          print("Работники с заданным стажем не найдены.")
92
93      elif command == 'help':
94          # Вывести справку о работе с программой.
95          print("Список команд:\n")
96          print("add - добавить работника;")
97          print("list - вывести список работников;")
98          print("select <стаж> - запросить работников со стажем;")
99          print("help - отобразить справку;")
100         print("exit - завершить работу с программой.")
101
102     else:
103         print(f"Неизвестная команда {command}", file=sys.stderr)

```

Рисунок 1.3 – Код примера

```

C:\Users\Borsukov\Desktop\LR9\PyCharm\venv\Scripts\python.exe C:\Users\Borsu
>>> 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? Воронов П.Г.
Должность? Заведующий кафедры
Год поступления? 1980
>>> add
Фамилия и инициалы? Адзиев И.И.
Должность? Преподаватель
Год поступления? 2018
>>> list
+-----+-----+-----+-----+
| № |          Ф.И.О.          |      Должность      |      Год      |
+-----+-----+-----+-----+
|  1 | Адзиев И.И.              | Преподаватель       |      2018     |
|  2 | Воронов П.Г.              | Заведующий кафедры  |      1980     |
+-----+-----+-----+-----+
>>> select 20
      1: Воронов П.Г.
>>> git
>>> Неизвестная команда git
exit|

```

Рисунок 1.4 – Результат работы примера

**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.

```
1  ▶  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  import sys
5
6  ▶  if __name__ == '__main__':
7
8      # Source Dictionary
9      school = {
10         '1a': 26,
11         '1б': 28,
12         '2a': 29,
13         '2б': 23,
14         '6a': 19,
15         '7a': 22,
16         '9a': 20,
17         '10a': 18,
18     }
19
20     while True:
21         command = input(">>> ").lower()
22
23         if command == "exit":
24             break
25
26         elif command == "change":
27             while True:
28                 k = input("Введите номер класса: ")
29                 n = input("Введите количество учащихся: ")
30                 if k in school:
31                     school[k] = n
32                     break
33                 else:
34                     print("Такого класса нет")
35             print(school)
36
37         elif command == "add":
38             while True:
39                 k = input("Введите номер нового класса: ")
```

Рисунок 2.1 – Код программы задания №1

```

40         n = input("Введите количество учащихся: ")
41         if k in school:
42             print("Этот номер класса занят")
43         else:
44             school[k] = n
45         break
46     print(school)
47
48     elif command == "delete":
49         while True:
50             k = input("Введите номер расформированного класса: ")
51             if k in school:
52                 school.pop(k)
53                 break
54             else:
55                 print("Такого класса нет")
56         print(school)
57
58     elif command == "count":
59         s = 0
60         for i in school:
61             s += int(school.get(i))
62         print(s)
63
64     else:
65         print(f"Неизвестная команда {command}", file=sys.stderr)

```

Рисунок 2.2 – Код программы задания №1

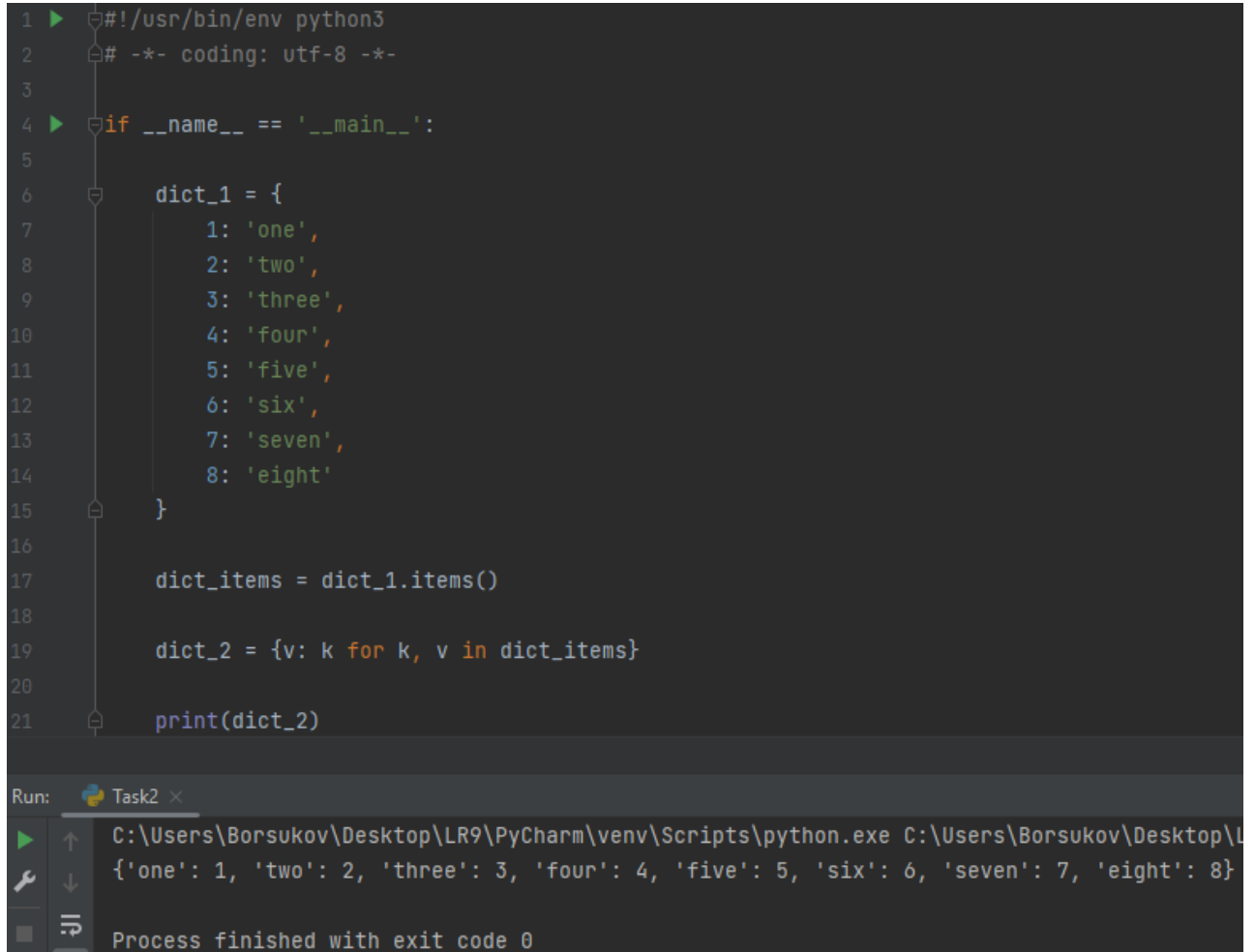
```

C:\Users\Borsukov\Desktop\LR9\PyCharm\venv\Scripts\python.exe C:\Users\Borsukov\Desktop\LR9\PyCharm
>>> change
Введите номер класса: 10a
Введите количество учащихся: 15
{'1a': 26, '16': 28, '2a': 29, '26': 23, '6a': 19, '7a': 22, '9a': 20, '10a': '15'}
>>> add
Введите номер нового класса: 11a
Введите количество учащихся: 13
{'1a': 26, '16': 28, '2a': 29, '26': 23, '6a': 19, '7a': 22, '9a': 20, '10a': '15', '11a': '13'}
>>> delete
Введите номер расформированного класса: 9a
{'1a': 26, '16': 28, '2a': 29, '26': 23, '6a': 19, '7a': 22, '10a': '15', '11a': '13'}
>>> git
>>> Неизвестная команда git
exit

```

Рисунок 2.3 – Результат работы кода задания №1

**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.



```
1 ▶ #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4 ▶ if __name__ == '__main__':
5
6     dict_1 = {
7         1: 'one',
8         2: 'two',
9         3: 'three',
10        4: 'four',
11        5: 'five',
12        6: 'six',
13        7: 'seven',
14        8: 'eight'
15    }
16
17    dict_items = dict_1.items()
18
19    dict_2 = {v: k for k, v in dict_items}
20
21    print(dict_2)
```

Run: Task2 ×

C:\Users\Borsukov\Desktop\LR9\PyCharm\venv\Scripts\python.exe C:\Users\Borsukov\Desktop\LR9\PyCharm\venv\Scripts\python.exe C:\Users\Borsukov\Desktop\LR9\PyCharm\venv\Scripts\python.exe C:\Users\Borsukov\Desktop\LR9\PyCharm\venv\Scripts\python.exe { 'one': 1, 'two': 2, 'three': 3, 'four': 4, 'five': 5, 'six': 6, 'seven': 7, 'eight': 8 }

Process finished with exit code 0

Рисунок 3 – Код и результат работы программы задания №2



**Индивидуальное задание:** использовать словарь, содержащий следующие ключи: фамилия и инициалы; номер группы; успеваемость (список из пяти элементов). Написать программу, выполняющую следующие действия: ввод с клавиатуры данных в список, состоящий из словарей заданной структуры; записи должны быть упорядочены по возрастанию номера группы; вывод на дисплей фамилий и номеров групп для всех студентов, включенных в массив, если средний балл студента больше 4.0; если таких студентов нет, вывести соответствующее сообщение..

```
1  ▶ #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  import sys
5
6  ▶ if __name__ == '__main__':
7      # Список для всех студентов.
8      students = []
9
10     # Список для студентов со средним баллом больше 4
11     filter_students = []
12
13     # Вывод справки.
14     print("Список команд:")
15     print("add - добавить студента")
16     print("list - вывести список студентов")
17     print("filter list - список студентов со средним баллом больше 4")
18     print("exit - завершить работу с программой")
19
20
21     # Бесконечный цикл запроса команд.
22     while True:
23         # Запрос команды из терминала.
24         command = input(">>> ").lower()
25
26         # Выполнение действия в соответствии с командой.
27         if command == 'exit':
28             break
29
30         elif command == 'add':
31             # Запрос данных о студенте.
32             name = input("Фамилия и инициалы студента: ")
33             group = int(input("Номер группы: "))
34             marks = list(map(int, input("Пять оценок студента: ").split()))
35
36             if len(marks) != 5:
37                 print("Неверное количество оценок", file=sys.stderr)
38                 continue
```

Рисунок 4.1 – Код программы индивидуального задания

```

39
40     # Создание словаря.
41     student = {
42         'name': name,
43         'group': group,
44         'marks': marks,
45     }
46
47     # Добавление словаря в список, если средний балл больше 4.
48     if sum(marks) / 5 > 4:
49         filter_students.append(student)
50
51     # Добавление словаря в список со всеми студентами.
52     students.append(student)
53
54     # Сортировка списков по номеру группы.
55     if len(students) > 1:
56         students.sort(key=lambda item: item.get('group', ''))
57
58     if len(filter_students) > 1:
59         students.sort(key=lambda item: item.get('group', ''))
60
61     elif command == 'list':
62         if len(students) > 0:
63             # Заголовок таблицы.
64             line = '+--{}--{}--{}--+'.format(
65                 '-' * 4,
66                 '-' * 30,
67                 '-' * 14,
68             )
69             print(line)
70             print(
71                 '| {:^4} | {:^30} | {:^14} |'.format(
72                     "№",
73                     "Ф.И.О.",
74                     "Номер группы",
75                 )
76             )

```

Рисунок 4.2 – Код программы индивидуального задания

```

77         print(line)
78
79         # Вывод данных о всех студентах.
80         for idx, student in enumerate(students, 1):
81             print(
82                 '| {:>4} | {:<30} | {:<14} |'.format(
83                     idx,
84                     student.get('name', ''),
85                     student.get('group', ''),
86                 )
87             )
88         print(line)
89
90     else:
91         print("Список студентов пустой.")
92
93     elif command == "filter list":
94         if len(students) > 0:
95             # Заголовок таблицы.
96             line = '+-{}-+-{}-+-{}-+'.format(
97                 '-' * 4,
98                 '-' * 30,
99                 '-' * 14,
100             )
101             print(line)
102             print(
103                 '| {:^4} | {:^30} | {:^14} |'.format(
104                     "№",
105                     "Ф.И.О.",
106                     "Номер группы",
107                 )
108             )
109             print(line)
110
111             # Вывод данных о студентах со средним баллом больше 4.
112             for idx, student in enumerate(filter_students, 1):
113                 print(
114                     '| {:>4} | {:<30} | {:<14} |'.format(

```

Рисунок 4.3 – Код программы индивидуального задания

```
115         idx,  
116         student.get('name', ''),  
117         student.get('group', ''),  
118     )  
119     )  
120     print(line)  
121  
122     else:  
123         print("Нет студентов со средним баллом больше 4")  
124  
125     else:  
126         print(f"Неизвестная команда {command}", file=sys.stderr)
```

Рисунок 4.4 – Код программы индивидуального задания

```

C:\Users\Borsukov\Desktop\LR9\PyCharm\venv\Scripts\python.exe C:\
Список команд:
add - добавить студента
list - вывести список студентов
filter list - список студентов со средним баллом больше 4
exit - завершить работу с программой
>>> add
Фамилия и инициалы студента: Иванов И.И.
Номер группы: 10
Пять оценок студента: 5 5 5 5 4
>>> add
Фамилия и инициалы студента: Петров П.И.
Номер группы: 21
Пять оценок студента: 5 4 4 4 4
>>> add
Фамилия и инициалы студента: Козлов К.К.
Номер группы: 3
Пять оценок студента: 3 3 3 3 3
>>> list
+-----+-----+-----+-----+
| № |           Ф.И.О.           | Номер группы |
+-----+-----+-----+-----+
|  1 | Козлов К.К.                | 3            |
|  2 | Иванов И.И.                | 10           |
|  3 | Петров П.И.                | 21           |
+-----+-----+-----+-----+
>>> filter list
+-----+-----+-----+-----+
| № |           Ф.И.О.           | Номер группы |
+-----+-----+-----+-----+
|  1 | Иванов И.И.                | 10           |
|  2 | Петров П.И.                | 21           |
+-----+-----+-----+-----+
>>> command
>>> Неизвестная команда command
exit

Process finished with exit code 0

```

Рисунок 4.5 – Результат работы программы индивидуального задания

## 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. Данные в словаре хранятся в формате ключ – значение.

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

Функция len() широко используется для определения размера объектов в Python. В нашем случае передача объекта словаря этой функции вернет размер словаря, то есть количество пар ключ-значение, присутствующих в словаре.

### 3. Какие методы обхода словарей Вам известны?

Элементы словаря перебираются в цикле for также, как элементы других сложных объектов. Однако "по-умолчанию" извлекаются только ключи.

С другой стороны у словаря как класса есть метод items(), который создает особую структуру, состоящую из кортежей. Каждый кортеж включает ключ и значение.

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.

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

```
>>> for i in nums:
...     print(nums[i])
...
one
two
three
```

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

С помощью метода `setdefault()`, при непосредственном обращении к ключу словарю.

## 6. Что такое словарь включений?

Словарь включений аналогичен списковым включениям, за исключением того, что он создаёт объект словаря вместо списка.

Основной пример:

```
>>> {x: x * x for x in (1, 2, 3, 4)}  
{1: 1, 2: 4, 3: 9, 4: 16}
```

## 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. У функции `zip()` множество сценариев применения. Например, она пригодится, если нужно создать набор словарей из двух массивов, каждый из которых содержит имя и номер сотрудника. Функция `zip()` принимает итерируемый объект, например, список, кортеж, множество или словарь в качестве аргумента. Затем она генерирует список кортежей, которые содержат элементы из каждого объекта, переданного в функцию. Предположим, что есть список имен и номером сотрудников, и их нужно объединить в массив кортежей. Для этого можно использовать функцию `zip()`.

## 8. Самостоятельно изучите возможности модуля `datetime`. Каким функционалом по работе с датой и временем обладает этот модуль?

Модуль `datetime` предоставляет классы для обработки времени и даты разными способами. Поддерживается и стандартный способ представления времени, однако больший упор сделан на простоту манипулирования датой, временем и их частями.