

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 №9**  
**дисциплины «Программирование на Python»**

Выполнил:

Лейс Алексей Вячеславович  
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»

---

(подпись)

Руководитель практики: кандидат тех.  
наук доцент кафедры  
инфокоммуникаций: Воронкин Р.А

---

(подпись)

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

Ставрополь, 2023 г.

## 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.

### Порядок выполнения работы:

Пример 1:

```
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4
5  import re
6  import sys
7  from datetime import date
8
9
10 if __name__ == '__main__':
11     # Список работников.
12     workers = []
13
14     # Организовать бесконечный цикл запроса команд.
15     while True:
16         # Запросить команду из терминала.
17         command = input(">>> ").lower()
18
19         # Выполнить действие в соответствие с командой.
20         match command:
21             case 'exit':
22                 break
23
24             case 'add':
25                 # Запросить данные о работнике.
26                 name = input("Фамилия и инициалы? ")
27                 post = input("Должность? ")
28                 year = int(input("Год поступления? "))
29
30                 # Создать словарь.
31                 worker = {
32                     'name': name,
33                     'post': post,
34                     'year': year,
35                 }
36
37                 # Добавить словарь в список.
38                 workers.append(worker)
39                 # Отсортировать список в случае необходимости.
40                 if len(workers) > 1:
41                     workers.sort(key=lambda item: item.get('name', ''))
42
43             case 'list':
44                 # Заголовок таблицы.
45                 line = '+-{}-+-{}-+-{}-+-{}-+'.format(
46                     '-' * 4,
47                     '-' * 30,
48                     '-' * 20,
49                     '-' * 8
```

```

50     )
51     print(line)
52     print(
53         '| {:^4} | {:^30} | {:^20} | {:^8} |'.format(
54             "№",
55             "Ф.И.О.",
56             "Должность",
57             "Год"
58         )
59     )
60     print(line)
61
62     # Вывести данные о всех сотрудниках.
63     for idx, worker in enumerate(workers, 1):
64         print(
65             '| {:>4} | {:<30} | {:<20} | {:>8} |'.format(
66                 idx,
67                 worker.get('name', ''),
68                 worker.get('post', ''),
69                 worker.get('year', 0)
70             )
71         )
72
73     print(line)
74
75     case _ if (m := re.match(r'select (\d+)', command)):
76         # Получить текущую дату.
77         today = date.today()
78
79         # Получить требуемый стаж.
80         period = int(m.group(1))
81
82         # Инициализировать счетчик.
83         count = 0
84         # Проверить сведения работников из списка.
85         for worker in workers:
86             if today.year - worker.get('year', today.year) >= period:
87                 count += 1
88             print(
89                 '{:>4}: {}'.format(count, worker.get('name', ''))
90             )
91
92         # Если счетчик равен 0, то работники не найдены.
93         if count == 0:
94             print("Работники с заданным стажем не найдены.")
95
96     case 'help':
97         # Вывести справку о работе с программой.
98         print("Список команд:\n")
99
100         print("add - добавить работника;")
101         print("list - вывести список работников;")
102         print("select <стаж> - запросить работников с заданным стажем;")
103         print("help - отобразить справку;")
104         print("exit - завершить работу с программой.")
105
106     case _:
107         print(f"Неизвестная команда {command}", file=sys.stderr)

```

### Индивидуальное задание:

Использовать словарь, содержащий следующие ключи: расчетный счет плательщика; расчетный счет получателя; перечисляемая сумма в руб. Написать программу, выполняющую следующие действия: ввод с клавиатуры данных в список, состоящий из словарей заданной структуры; записи должны быть размещены в алфавитном порядке по расчетным счетам плательщиков; вывод на экран информации о сумме, снятой с расчетного счета плательщика, введенного с клавиатуры; если такого расчетного счета нет, выдать на дисплей соответствующее сообщение.

```
1 # Задача 2: Использовать словарь для учета данных о переводах между счетами.
2
3 # Ввод данных с клавиатуры в список словарей
4 transactions_list = []
5 while True:
6     payer_account = input("Введите расчетный счет плательщика (или 'конец' для завершения): ")
7
8     if payer_account.lower() == 'конец':
9         break
10
11     recipient_account = input("Введите расчетный счет получателя: ")
12     transfer_amount = float(input("Введите перечисляемую сумму в рублях: "))
13
14     transaction_data = {
15         'payer_account': payer_account,
16         'recipient_account': recipient_account,
17         'transfer_amount': transfer_amount
18     }
19
20     transactions_list.append(transaction_data)
21
22 # Сортировка списка по расчетным счетам плательщиков
23 transactions_list.sort(key=lambda x: x['payer_account'])
24
25 # Вывод на экран информации о сумме, снятой с расчетного счета плательщика
26 search_account = input("Введите расчетный счет плательщика для поиска суммы: ")
27 found = False
28
29 for transaction in transactions_list:
30     if transaction['payer_account'] == search_account:
31         print(f"Сумма, снятая с расчетного счета {search_account}: {transaction['transfer_amount']} руб.")
32         found = True
33         break
34
35 if not found:
36     print(f"Расчетный счет {search_account} не найден в данных.")
37
```

input

Введите расчетный счет плательщика для поиска суммы: 12

Сумма, снятая с расчетного счета 12: 700.0 руб.

Ответы на вопросы:

1. **Что такое словари в языке Python?**
  - Словари в Python - это изменяемые коллекции, представляющие собой набор пар "ключ-значение". Они используются для хранения данных, к которым можно обращаться по ключу.
2. **Может ли функция len() быть использована при работе со словарями?**
  - Да, функция len() может быть использована для определения количества элементов (пар "ключ-значение") в словаре.
3. **Какие методы обхода словарей Вам известны?**
  - Для обхода словарей можно использовать циклы, такие как **for key in my\_dict:**. Также есть методы, такие как **keys()**, **values()**, и **items()**, которые возвращают представления ключей, значений и пар "ключ-значение" соответственно.

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

- Значения из словаря можно получить по ключу с использованием квадратных скобок, например: `value = my_dict[key]`.

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

- Значение в словаре по ключу можно установить с использованием квадратных скобок, например: `my_dict[key] = value`.

6. **Что такое словарь включений?**

- Словарь включений (или словарное включение) - это компактный способ создания словаря в одной строке с использованием синтаксиса, например: `my_dict = {key: value for key, value in iterable}`.

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

- Функция `zip()` используется для объединения элементов из нескольких итерируемых объектов. Пример: `list_of_keys = ['a', 'b', 'c'], list_of_values = [1, 2, 3], my_dict = dict(zip(list_of_keys, list_of_values))`.

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

- Модуль `datetime` предоставляет классы для работы с датой и временем. Он включает функционал для создания, форматирования и арифметики с датами и временем, а также поддерживает различные календарные системы. Примеры включают `datetime.datetime` для представления даты и времени, и методы для работы с ними, такие как `strftime()`, `strptime()`, и многое другое.

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