МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное
образовательное учреждение высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Кафедра инфокоммуникаций

Отчет по лабораторной работе № 8

Работа со словарями в языке Python

по дисциплине «Технологии программирования и алгоритмизации»

Выполнил студент группы ИВТ-б-о-20-1
Пушкин Н.С. « »_____20__г.
Подпись студента_____
Работа защищена« »_____20__г.

Проверил Воронкин Р.А._____
(подпись)

Ставрополь 2021

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

Ход работы:

1. Изучив теоретическую часть в методических указаниях, приступил к выполнению общих заданий.

2. Первое общее задание:

```
}
for key, value in school.items():
    print("Класс: ", key, " - количество учащихся: ", value)
a = input('Введите название именяемого класса: ')
a1 = int(input('Количество учеников изменяемого класса: '))
b = input("Введите новый класс: ")
b1 = int(input("Введите сколько новых учеников в классе: "))
c = input('Введите класс, который был расформирован: ')
del school[c]
school.update({b: b1})
summ = 0
for key, value in list(school.items()):
    if a == key:
        value = a1
    s = int(value)
    summ = s + summ
    print("Класс: ", key, " - количество учащихся: ", value)
print('Общее количество учащихся в школе: ', summ)
```

Рисунок 8.1 – Код первого общего задания

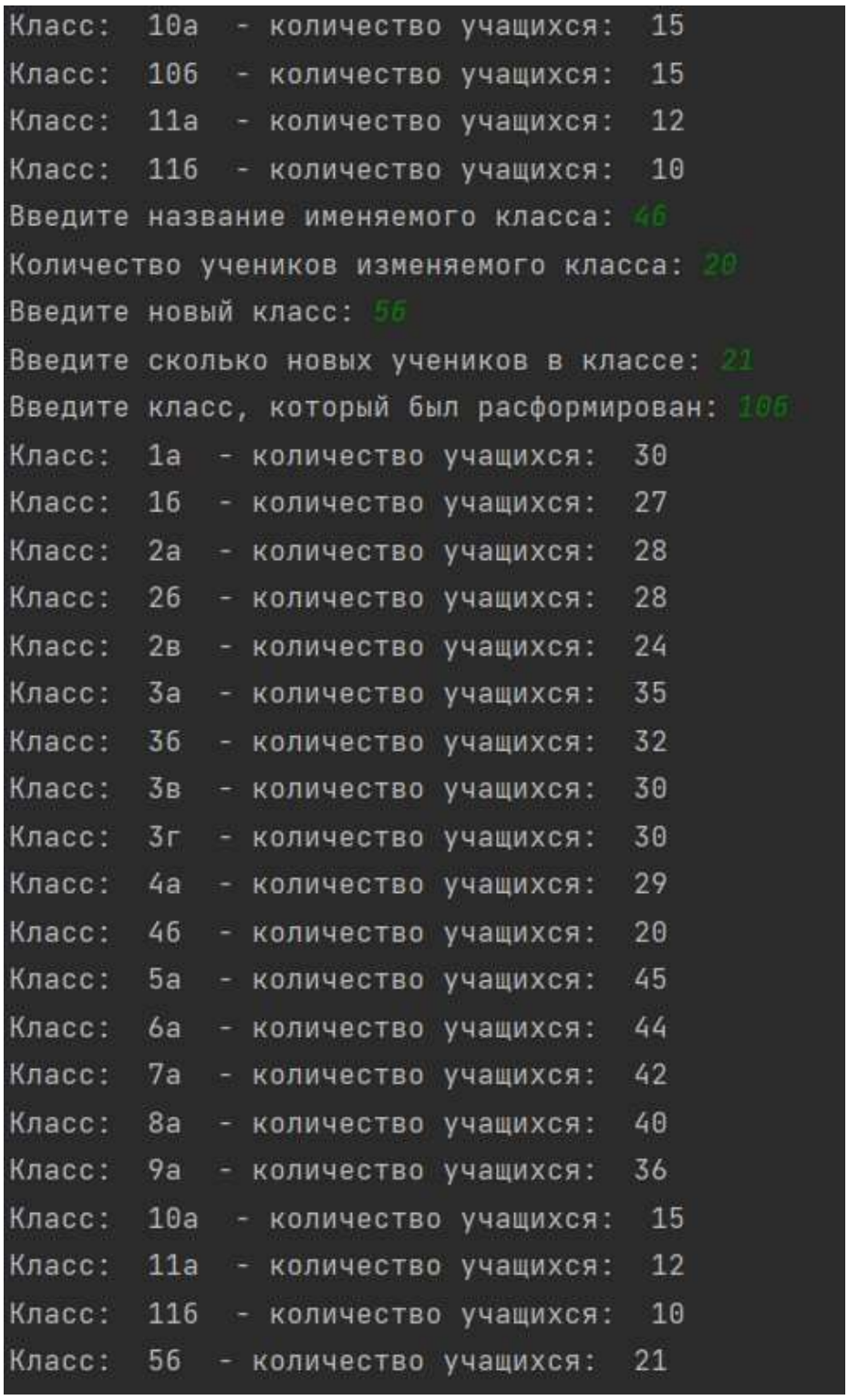


Рисунок 8.2 – Результат выполнения кода первого задания

3. После выполнения первого общего задания, приступил ко второму:

```
words = {
    1: 'one',
    2: 'two',
    3: 'three',
    4: 'four'
}
dict_items = words.items()
new_words = dict(zip(words.values(), words.keys()))
print(new_words)
```

Рисунок 8.3 – Код второго общего задания

```
C:\Users\zligo\PycharmProjects\pythonProject\ven
{'one': 1, 'two': 2, 'three': 3, 'four': 4}
```

Рисунок 8.4 – Результат выполнения второго задания

4. После выполнения общих заданий приступил к выполнению индивидуального.

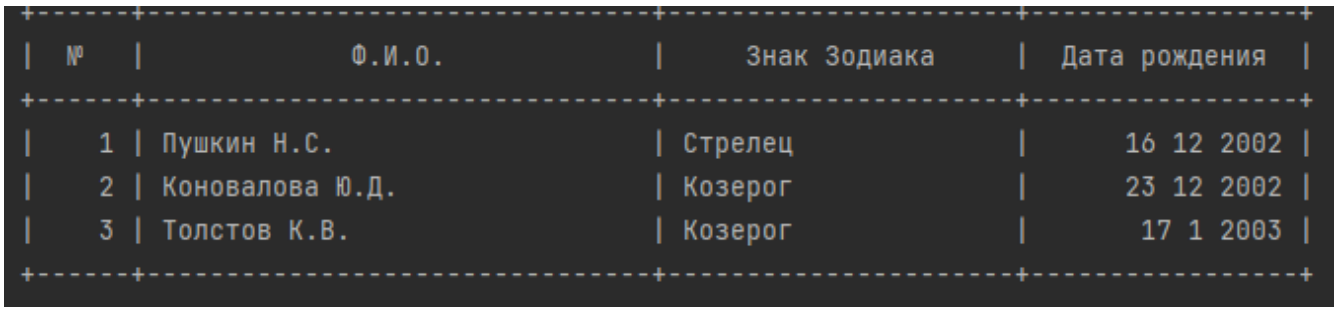
Вариант №16

```
        if command == 'exit':
            break

        elif command == 'add':
            # Запросить данные о работнике.
            name = input("Фамилия и инициалы? ")
            zodiac = input("Знак Зодиака? ")
            year = list(map(int, input("Дата рождения? ").split()))
            # Создать словарь.
            worker = {
                'name': name,
                'zodiac': zodiac,
                'year': year,
            }
            # Добавить словарь в список.
            workers.append(worker)
            # Отсортировать список в случае необходимости.
            if len(workers) > 1:
                workers.sort(key=lambda x: x.get('year'))

        elif command == 'list':
            # Заголовок таблицы.
            line = '+-{}-+-{}-+-{}-+-{}-+'.format(
                '-' * 4,
                '-' * 30,
                '-' * 20,
                '-' * 15
            )
            print(line)
            print(
                '| {:^4} | {:^30} | {:^20} | {:^15} |'.format(
                    "№",
                    "Ф.И.О.",
                    "Знак Зодиака",
                    "Дата рождения"
                )
            )
```

Рисунок 8.5 – Код индивидуального задания

```
add
Фамилия и инициалы? Пушкин Н.С.
Знак Зодиака? тСтрелец
Дата рождения? 1о 12
>>> add
Фамилия и инициалы? Коноваловa Ю.Д
Знак Зодиака? Козерог
Дата рождения? 23 12
>>> add
Фамилия и инициалы? Толстов К.В.
Знак Зодиака? Козерог
Дата рождения? 17 1
```

Рисунок 8.6 – Добавление пользователем данных в словарь



№	Ф.И.О.	Знак Зодиака	Дата рождения
1	Пушкин Н.С.	Стрелец	16 12 2002
2	Коновалова Ю.Д.	Козерог	23 12 2002
3	Толстов К.В.	Козерог	17 1 2003

Рисунок 8.7 – Вывод данных с сортировкой Ф.И.О.

Вывод: в ходе выполнения лабораторной работы были приобретены навыки по работе со словарями и их методами на языке программирования Python.

Контрольные вопросы

1. Что такое словари в языке Python?

Словарь – это структура данных, которая предназначена для хранения произвольных объектов с доступом по ключу.

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

Да

3. Какие методы обхода словарей Вам известны?

1) For I in dict

2) for key in dict.keys()

3) for value in dict.value()

4) for key, value in dict.items()

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

Print(dict[‘Название ключа’])

Print(dict.get(‘Название ключа’))

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

Dict[‘Ключ’]= значение

6. Что такое словарь исключений?

Исключение в Python – это конструкция, используемая для сигнализации о важном событии, обычно об ошибке, которое происходит при выполнении программы. Исключение может привести к остановке программы, если она не будет должным образом «поймана» (т.е. обработана правильно). Если вы думаете, что ваша программа может вызвать исключение при выполнении.

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

Функция zip объединяет в кортежи элементы из последовательностей переданных в качестве аргументов. Функция прекращает выполнение, как только достигнут конец самого короткого списка.

```
a = [1,2,3]
b = "xyz"
c = (None, True)
res = list(zip(a, b, c))
print (res)
```

[(1, 'x', None), (2, 'y', True)]

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.

Класс datetime.date(year, month, day) - стандартная дата. Атрибуты: year, month, day. Неизменяемый объект.

Класс datetime.time(hour=0, minute=0, second=0, microsecond=0, tzinfo=None) - стандартное время, не зависит от даты. Атрибуты: hour, minute, second, microsecond, tzinfo.

Класс datetime.timedelta - разница между двумя моментами времени, с точностью до микросекунд.

Класс datetime.tzinfo - абстрактный базовый класс для информации о временной зоне (например, для учета часового пояса и / или летнего времени).

Класс datetime.datetime(year, month, day, hour=0, minute=0, second=0, microsecond=0, tzinfo=None) - комбинация даты и времени.

Обязательные аргументы:

- datetime.MINYEAR (1) ≤ year ≤ datetime.MAXYEAR (9999)
- 1 ≤ month ≤ 12
- 1 ≤ day ≤ количество дней в данном месяце и году
- Методы класса datetime:
- datetime.today() - объект datetime из текущей даты и времени. Работает также, как и datetime.now() со значением tz=None.
- datetime.fromtimestamp(timestamp) - дата из стандартного представления времени.
- datetime.fromordinal(ordinal) - дата из числа, представляющего собой количество дней, прошедших с 01.01.1970.

- datetime.now(tz=None) - объект datetime из текущей даты и времени.
- datetime.combine(date, time) - объект datetime из комбинации объектов date и time.
- datetime.strptime(date_string, format) - преобразует строку в datetime (так же, как и функция strptime из модуля time).
- datetime.strftime(format) - см. функцию strftime из модуля time.
- datetime.date() - объект даты (с отсечением времени).
- datetime.time() - объект времени (с отсечением даты).
- datetime.replace([year[, month[, day[, hour[, minute[, second[, microsecond[, tzinfo]]]]]]]]) - возвращает новый объект datetime с изменёнными атрибутами.
- datetime.timetuple() - возвращает struct_time из datetime.
- datetime.toordinal() - количество дней, прошедших с 01.01.1970.
- datetime.timestamp() - возвращает время в секундах с начала эпохи.
- datetime.weekday() - день недели в виде числа, понедельник - 0, воскресенье - 6.
- datetime.isoweekday() - день недели в виде числа, понедельник - 1, воскресенье - 7.
- datetime.isocalendar() - кортеж (год в формате ISO, ISO номер недели, ISO день недели).
- datetime.isoformat(sep='T') - красивая строка вида "YYYY-MM-DDTHH:MM:SS.mmmmmm" или, если microsecond == 0, "YYYY-MM-DDTHH:MM:SS"
- datetime.ctime() - см. ctime() из модуля time.