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»

Выполнила студентка группы ИВТ-б-о-20-1

Хацукова А.И. « »_____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. Изучила теоретическую часть лабораторной работы и выполнила пример:

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

from datetime import date
import sys

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)
```

2. Далее приступила к выполнению первого общего задания:

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

if __name__ == '__main__':
    school = {
        '1a': 31,
        '1b': 30,
        '2c': 28,
        '5a': 24,
        '9c': 30,
        '11b': 17
    }
    for key, value in school.items():
        print("В классе ", key, " учатся ", value, "учеников")
    print('Меняем список....')
    school['11b'] = 12
    del school['1a']
    school['9c'] = '29'

    for key, value in school.items():
        print("Теперь в классе ", key, " учатся ", value, "студентов")

    s = 0
    for i in school:
        s = s + int(school[i])
    print("Теперь в школе учится:", s)
```

Рисунок 1 – Первое задание

```
Run:  zd1
C:\Users\Аделайда\AppData\Local\Programs\Python\Pyth
В классе  1a  учатся  31 учеников
В классе  1b  учатся  30 учеников
В классе  2c  учатся  28 учеников
В классе  5a  учатся  24 учеников
В классе  9c  учатся  30 учеников
В классе  11b  учатся  17 учеников
Меняем список....
Теперь в классе  1b  учатся  30 студентов
Теперь в классе  2c  учатся  28 студентов
Теперь в классе  5a  учатся  24 студентов
Теперь в классе  9c  учатся  29 студентов
Теперь в классе  11b  учатся  12 студентов
Теперь в школе учится: 123

Process finished with exit code 0
```

Рисунок 2 – Результат выполнения первого задания

3. Второе общее задание:

README.md | primer.py | zd1.py | zd2.py | ind.py

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

if __name__ == '__main__':
    bts = {
        1: 'taehyung',
        2: 'jungkook',
        3: 'jimin',
        4: 'namjoon',
        5: 'hoseok',
        6: 'jin',
        7: 'yoongi'
    }
    print(bts)
    dict_items = bts.items()
    new_bts = dict(zip(bts.values(), bts.keys()))
    print(new_bts)
```

Рисунок 3 – Второе общее задание

```
C:\Users\Аделайда\AppData\Local\Programs\Python\Python39\python.exe C:/Users/Аделайда/Documents/laba8/zd2.py
{1: 'taehyung', 2: 'jungkook', 3: 'jimin', 4: 'namjoon', 5: 'hoseok', 6: 'jin', 7: 'yoongi'}
{'taehyung': 1, 'jungkook': 2, 'jimin': 3, 'namjoon': 4, 'hoseok': 5, 'jin': 6, 'yoongi': 7}

Process finished with exit code 0
```

Рисунок 4 – Результат выполнения второго задания

4. После приступила к выполнению индивидуального задания по варианту (1 вариант)

5.

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

import sys

if __name__ == '__main__':
    # Список студентов.
    students = []
    # 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? ")
```

```
            num = input("Номер группы? ")
            print("Успеваемость?  ")
            mat = int(input())
            rus = int(input())
            hist = int(input())
            phis = int(input())
            bio = int(input())
            # Создать словарь.
            student = {
                'name': name,
                'num': num,
                'mat': mat,
                'rus': rus,
                'hist': hist,
                'phis': phis,
                'bio': bio
            }
            # Добавить словарь в список.
            students.append(student)
            # Отсортировать список в случае необходимости.
            if len(students) > 1:
                students.sort(key=lambda item: item.get('num', ''))

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

```
                    "русский",
                    "физика",
                    "история",
                    "биология"
                )
            )
            print(line)
            # Вывести данные о всех сотрудниках.
            for idx, student in enumerate(students, 1):
                print(
                    '| {:>4} | {:<30} | {:<20} | {:>20} | {:>20} | {:<20} |
{:<20} | {:>20} |'.format(
                        idx,
                        student.get('name', ''),
                        student.get('num', ''),
                        student.get('mat', ),
                        student.get('rus', ),
                        student.get('hist', ),
                        student.get('phis', ),
                        student.get('bio', )
                    )
                )
            print(line)

        elif command.startswith('select '):
            count = 0
            for student in students:
                averg = (student.get('mat', '') + student.get('rus', '')
                         + student.get('hist', '') + student.get('phis', '')
+ student.get('bio', '')) / 5
                if averg >= 4:
                    count += 1
                    print(
                            '{:>4}: {}'.format(count, student.get('name',
''), student.get('num', ''))
                        )
            # Если счетчик равен 0, то студенты не найдены.
            if count == 0:
                print("Ученики со средним баллом выше 4-х не найдены.")
        elif command == 'help':
            # Вывести справку о работе с программой.
            print("Список команд:\n")
            print("add - добавить студентов;")
```

```
            print("list - вывести список студентов;")
            print("select - вывести имена студентов со среднем баллом выше
4")
            print("help - отобразить справку;")
            print("exit - завершить работу с программой.")
        else:
            print(f"Неизвестная команда {command}", file=sys.stderr)
```

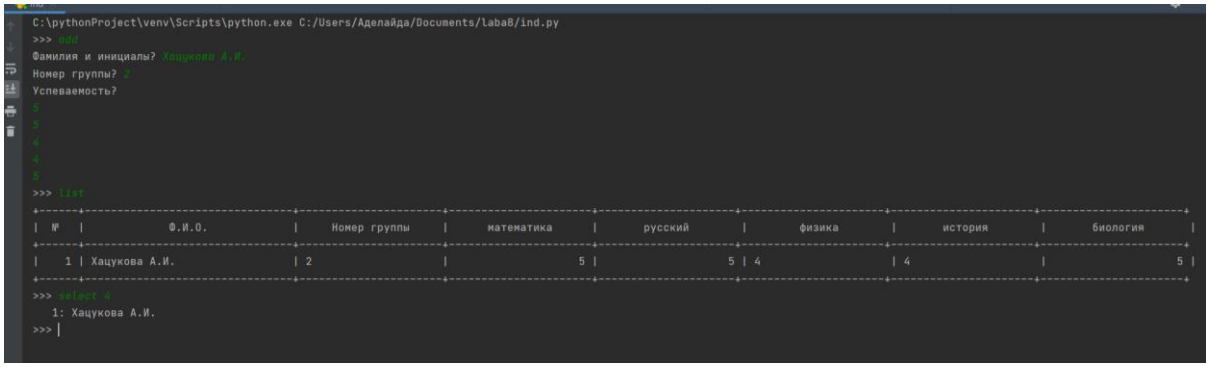

Рисунок 5 – Результат выполнения индивидуального задания

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.

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