

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ
ФЕДЕРАЦИИ

Федеральное государственное автономное
образовательное учреждение высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Кафедра инфокоммуникаций

Основы кроссплатформенного программирования

Отчет по лабораторной работе №7

«Работа со словарями в языке Python»

Выполнил студент группы ИТС-б-о-21-1

Балхиев Хусейн Зафарович

« » _____ 20__ г.

Подпись студента _____

Проверил: Доцент, к.т.н, доцент кафедры
инфокоммуникаций

Воронкин А. В.

Работа защищена с оценкой: _____

(подпись)

Ставрополь, 2022

Лабораторная работа 7

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.

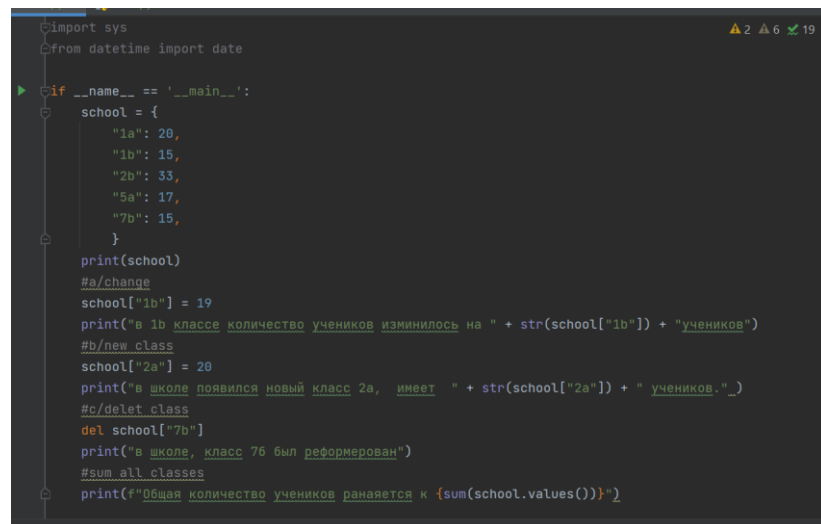
Ход работы:

Создадим общедоступный репозиторий

Индивидуальные задания:

Задание 1.

9. 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.



```
import sys
from datetime import date

if __name__ == '__main__':
    school = {
        "1a": 20,
        "1b": 15,
        "2b": 33,
        "5a": 17,
        "7b": 15,
    }
    print(school)
    #a/change
    school["1b"] = 19
    print("в 1б классе количество учеников изменилось на " + str(school["1b"]) + " учеников")
    #b/new class
    school["2a"] = 20
    print("в школе появился новый класс 2а, имеет " + str(school["2a"]) + " учеников.")
    #c/delete class
    del school["7b"]
    print("в школе, класс 7б был реформирован")
    #sum all classes
    print("общая количество учеников равняется к {sum(school.values())}")
```

Рис 1. Окно вывода Задания «Школа».

Задание 2.

Код программы:

```
#!/usr/bin/env
python3
```

```
# -*- coding: utf-8 -*-
```

```
import sys
from datetime import date
if __name__ == '__main__':
    students = []
    count = 0
    while True:
        command = input("Что вы хотите? ").lower()

        if command == "exit":
            break

        elif command == "add":
            name = input("Фамилия и инициалы? ")
            number = input("Номер группы? ")
            z = int(input("Оценка? "))
            z_1 = str(input("Успеваемость? "))

            student = {
                "name": name,
                "number": number,
                "z": z,
                "z_1": z_1
            }

            students.append(student)
            if len(students) > 1:
                students.sort(key=lambda item: item.get("name", ""))
                for idx, worker in enumerate(students, 1):
                    line = "| {:^3} | {:^25} | {:^15} | {:^10} | {:^15}
|".format(
                                idx,
                                worker.get("name", ""),
                                worker.get("number", ""),
                                worker.get("z", 0),
                                worker.get("z_1", 0)
                            )
                    print(line)
            elif command == "list":
                line = "+-{}-+-{}-+-{}-+-{}-+-{}-+ ".format(
                    "-" * 3,
                    "-" * 25,
                    "-" * 15,
                    "-" * 10,
                    "-" * 15,
                )
```

```

        "-" * 15
    )
    print(line)
    print(
        "| {:^3} | {:^25} | {:^15} | {:^10} | {:^15} |".format(
            "№",
            "Ф.И.О.",
            "Номер группы",
            "Оценка",
            "Успеваемость",
        )
    )
    print(line)
    for idx, worker in enumerate(students, 1):
        print(
            "| {:^3} | {:^25} | {:^15} | {:^10} | {:^15}
|".format(

                idx,
                worker.get("name", ""),
                worker.get("number", ""),
                worker.get("z", 0),
                worker.get("z_1", 0)
            )

        )
        print(line)

elif command == "select":
    count == 0
    mark = student.get("z", 0)
    for student in students:
        if 4 == mark or 5 == mark:
            count -= 1
            print(
                "{:>5} {}".format(" ", student.get("name", "")),
                "{:>1} {}".format("группа №",
student.get("number", ""))
            )
        if count == 0:
            print("таких студентов нет")
elif command == "help":
    print("список команд:\n")
    print("add - добавить студента;")
    print("list - список студентов;")
    print("select - вывести список студентов имеющих оценку 4 и
5")

    print("help - тобразить справку;")
    print("exit - завершить работу с программой.")

```

```
else:  
    print(f"Неизвестная команда {command}", file=sys.stderr)
```

```
список команд:  
  
add - добавить студента;  
list - список студентов;  
select - вывести список студентов имеющих оценку 4 и 5  
help - тобразить справку;  
exit - завершить работу с программой.  
Что вы хотите? 
```

Рис 3. Список команд.

```
Что вы хотите? add  
фамилия и инициалы? b yobb  
Номер группы? 1  
Оценка? 5  
Успеваемость? [jhjifz  
| 1 | b yobb | 1 | 5 | [jhjifz |  
Что вы хотите? 
```

Рис 3. Окно вывода Задания 2.

Ответы на вопросы:

1. Что такое словари в языке Python?

Ответ: 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?

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

3. Какие методы обхода словарей Вам известны?

Ответ: 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 де 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?

Ответ: 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?

Ответ: 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?

Ответ: 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.

Ответ: 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?

Ответ: *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: приобрёл навыки по работе со словарями при написании программ с помощью языка программирования Python версии 3.x.