

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ ФГАОУ ВО
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт цифрового развития
Кафедра инфокоммуникаций

Отчет по лабораторной работе №6
По дисциплине основы кроссплатформенного программирования
«Работа со словарями в языке Python»

Выполнила:

студентк группы ИТС-б-о-21-1

Аллаёров Жамшид Хасан угли

(подпись)

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

инфокоммуникаций

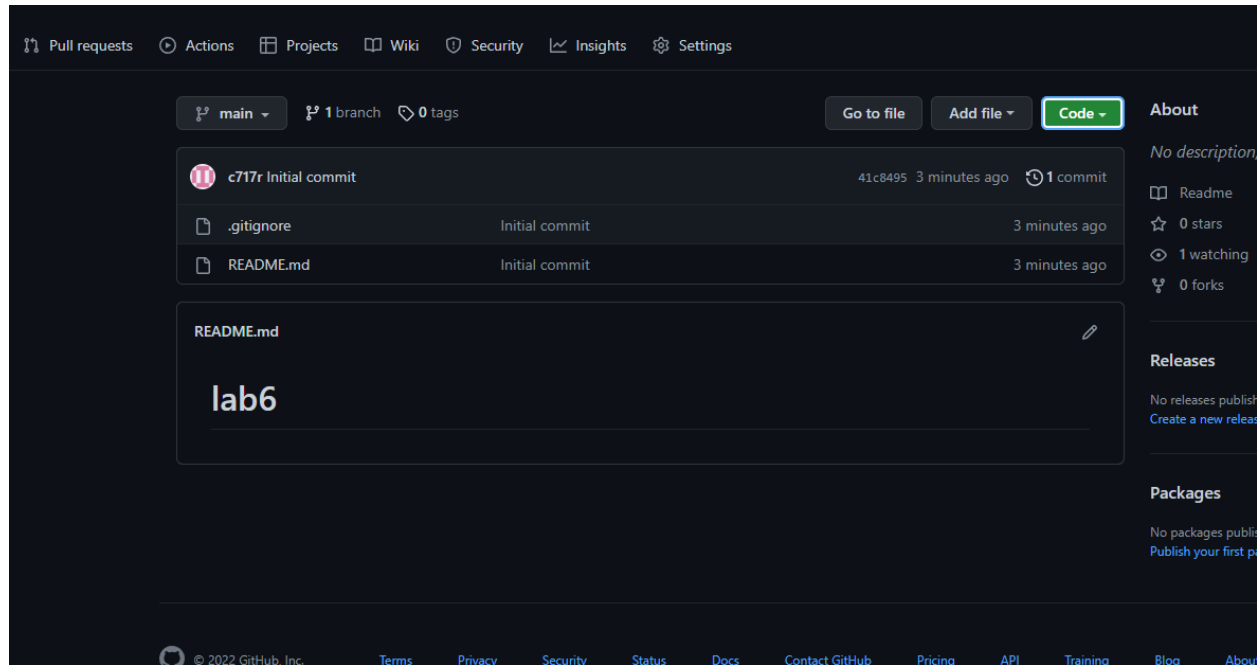
Воронкин Р. А.

Работа защищена с оценкой:

(подпись)

Ставрополь, 2022

Создать общедоступный репозиторий на GitHub, в котором будет использована лицензия MIT и язык программирования Python.



Решите задачу: создайте словарь, связав его с переменной `school`, и заполните 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.

The screenshot shows the Visual Studio Code interface with a file explorer on the left containing `.gitignore`, `ex1.py`, and `README.md`. The main editor displays the `ex1.py` file with the following Python code:

```
1 school = {"1a": 23, "1b": 22, "2a": 21, "2b": 20, "3a": 19, "3b": 20, "4a": 18, "4b":  
2         "6a": 20, "6b": 15, "7a": 18, "7b": 25, "8a": 25, "8b": 26, "9a": 29, "9b":  
3         "11a": 24, "11b": 23} # Словарь классов с количеством учащихся  
4  
5 school["2a"] = 19 # Во 2а классе изменилось количество учащихся  
6  
7 school["1b"] = 15 # В школе появился 1в класс  
8  
9 del school["10b"] # Класс 10б был удален  
10  
11 print(f"Всего учеников в школе: {sum(school.values())}")
```

The bottom panel shows the terminal output in Windows PowerShell:

```
Windows PowerShell  
(C) Корпорация Майкрософт (Microsoft Corporation). Все права защищены.  
  
Попробуйте новую кроссплатформенную оболочку PowerShell (https://aka.ms/pscore6)  
  
PS C:\Users\vis1on\lab6> & C:/Users/vis1on/AppData/Local/Programs/Python/Python310/python.exe c:/Use  
s/vis1on/lab6/ex1.py  
Всего учеников в школе: 466  
PS C:\Users\vis1on\lab6>
```

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.

EXPLORER

LAB6

.gitignore

ex1.py

ex2.py

README.md

Get Started

ex1.py

ex2.py

ex2.py > ...

```
1  a_dict = {'color': 'blue', 'fruit': 'apple', 'pet': 'dog'}
2  d_items = a_dict.items()
3
4  for item in a_dict.items():
5      print(item)
6
```

PROBLEMS

OUTPUT

DEBUG CONSOLE

TERMINAL

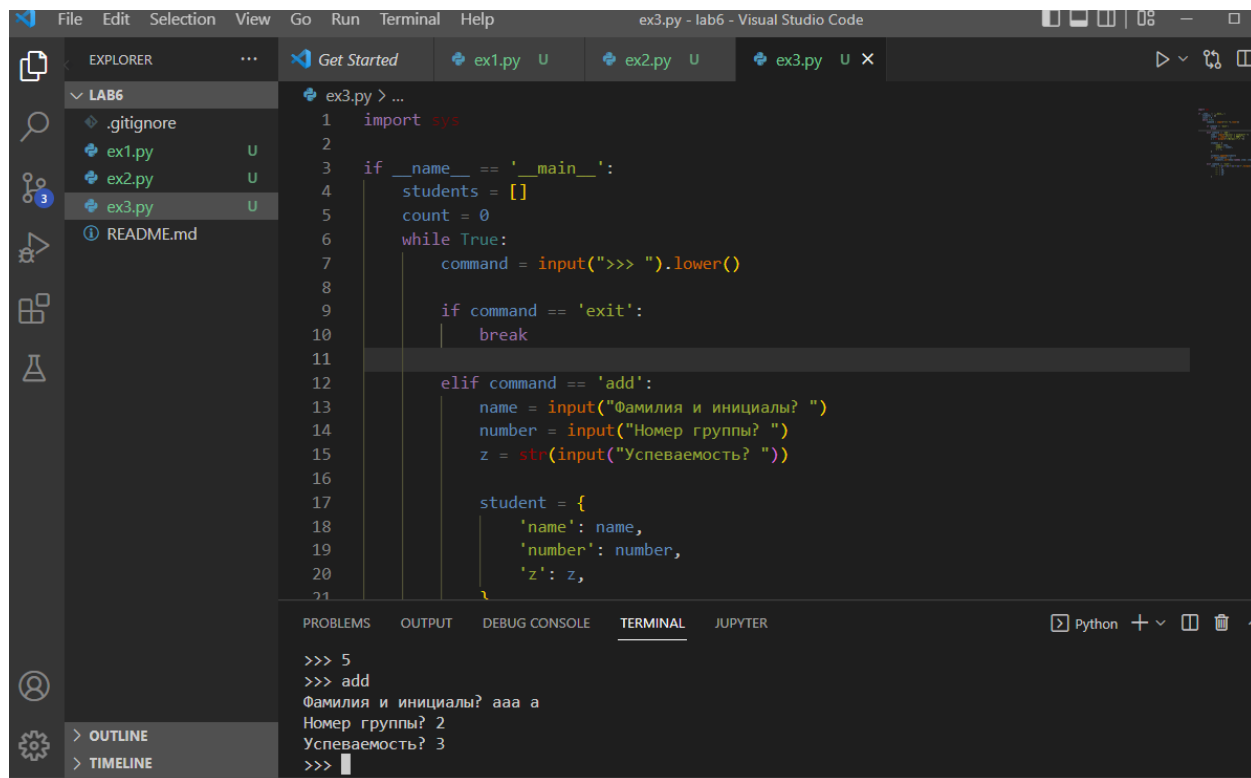
JUPYTER

```
PS C:\Users\vision\lab6> & C:/Users/vis1on/AppData/Local/Programs/Python/Pyt
s/vis1on/lab6/ex2.py
('color', 'blue')
('fruit', 'apple')
('pet', 'dog')
PS C:\Users\vision\lab6>
```

OUTLINE

TIMELINE

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 алфавиту; вывод на дисплей фамилий и номеров групп для всех студентов, имеющих хотя бы одну оценку 2; если таких студентов нет, вывести соответствующее сообщение.

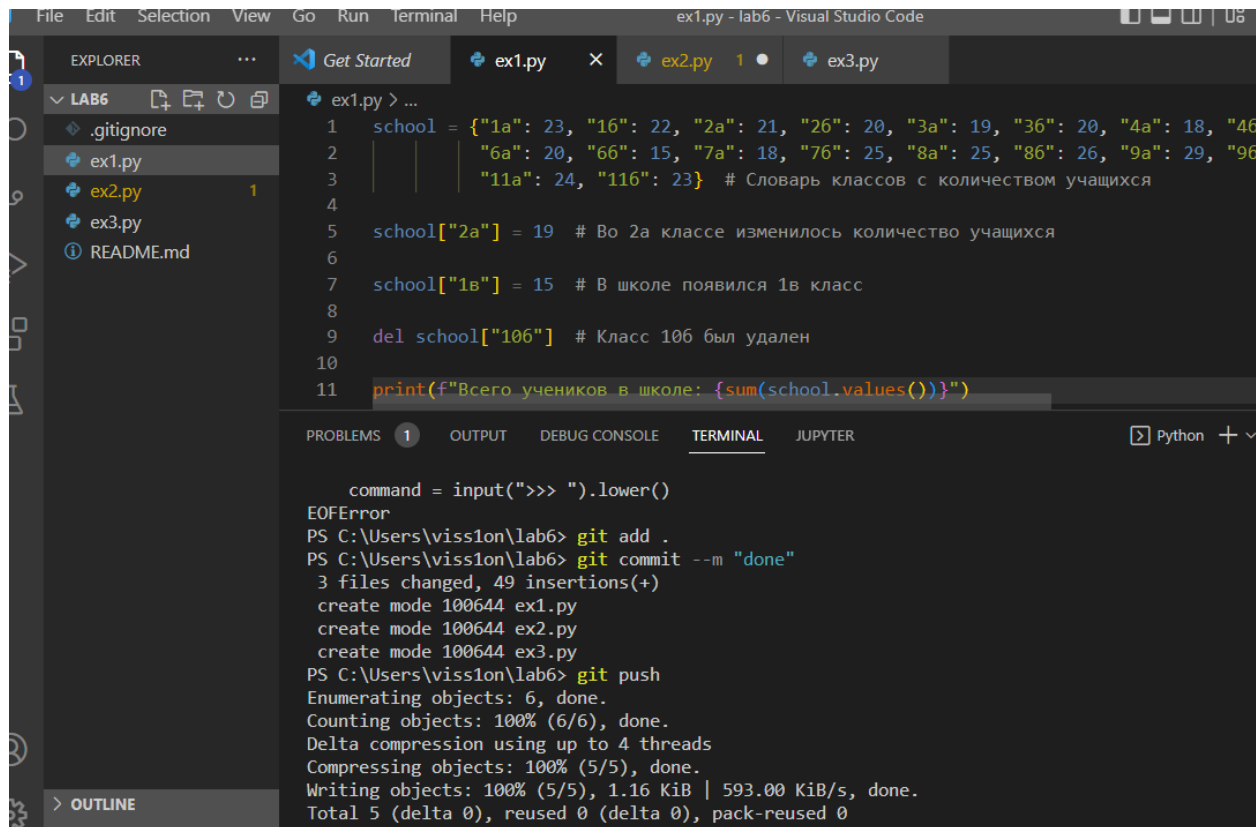


The screenshot shows the Visual Studio Code interface with a Python file named `ex3.py` open. The file is located in a workspace named `LAB6`. The code in `ex3.py` is as follows:

```
1 import sys
2
3 if __name__ == '__main__':
4     students = []
5     count = 0
6     while True:
7         command = input(">>> ").lower()
8
9         if command == 'exit':
10             break
11
12         elif command == 'add':
13             name = input("Фамилия и инициалы? ")
14             number = input("Номер группы? ")
15             z = str(input("Успеваемость? "))
16
17             student = {
18                 'name': name,
19                 'number': number,
20                 'z': z,
21             }
```

The terminal window at the bottom shows the execution of the script. The user has entered `5` at the prompt, then `add`. The program then prompts for the student's name, group number, and grade. The user has entered `aaa a` for the name, `2` for the group number, and `3` for the grade. The prompt `>>>` is shown at the end of the terminal output.

Отправьте сделанные изменения на сервер GitHub.



```
File Edit Selection View Go Run Terminal Help
ex1.py - lab6 - Visual Studio Code

EXPLORER
LAB6
  .gitignore
  ex1.py
  ex2.py
  ex3.py
  README.md

ex1.py > ...
1 school = {"1a": 23, "16": 22, "2a": 21, "26": 20, "3a": 19, "36": 20, "4a": 18, "46": 19, "5a": 17, "56": 18, "6a": 20, "66": 15, "7a": 18, "76": 25, "8a": 25, "86": 26, "9a": 29, "96": 28, "10a": 27, "106": 26, "11a": 24, "116": 23} # Словарь классов с количеством учащихся
2
3
4
5 school["2a"] = 19 # Во 2а классе изменилось количество учащихся
6
7 school["1в"] = 15 # В школе появился 1в класс
8
9 del school["106"] # Класс 106 был удален
10
11 print(f"Всего учеников в школе: {sum(school.values())}")

TERMINAL
command = input(">>> ").lower()
EOFError
PS C:\Users\vision\lab6> git add .
PS C:\Users\vision\lab6> git commit --m "done"
3 files changed, 49 insertions(+)
create mode 100644 ex1.py
create mode 100644 ex2.py
create mode 100644 ex3.py
PS C:\Users\vision\lab6> git push
Enumerating objects: 6, done.
Counting objects: 100% (6/6), done.
Delta compression using up to 4 threads
Compressing objects: 100% (5/5), done.
Writing objects: 100% (5/5), 1.16 KiB | 593.00 KiB/s, done.
Total 5 (delta 0), reused 0 (delta 0), pack-reused 0
```

Что такое словари в языке Python?

Словари в Python - неупорядоченные коллекции произвольных объектов с доступом по ключу. Их иногда ещё называют ассоциативными массивами или хеш-таблицами.

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

Вы можете легко проверить, все ли ключи на месте, сравнив количество строк в файле с количеством ключей в словаре.

```
# Считаем количество строк в файле print("Number of lines: " + str(len(lines))) # Считаем ключи словаря, их должно быть столько же, сколько строк в файле print("Number of dictionary keys: " + str(len(reviews.keys())))
```

3. Какие методы обхода словарей Вам известны?

Объявим словарь с отношением различных валют к российскому рублю, который нам по какой-то причине нужно обойти:

```
currencies = {"rub": 1, "usd": 69.78, "eur": 78.28}
```

Самый очевидный вариант обхода словаря — это попытаться напрямую запустить цикл `for` по объекту словаря, так же как мы делаем это со списками, кортежами, строками и любыми другими итерируемыми объектами.

```
for something in currencies:
```

```
    print(something)
```

Словарь и правда поддерживает протокол итераций, но словарь не так прост, как другие объекты, которые мы упомянули выше. Словарь состоит из нескольких частей, ведь словарь — это отношение между ключами и значениями. Получается, что теоретически цикл по словарю может получать либо ключи, либо значения, либо пары (ключ, значение). Попробуйте угадать, что же именно выведет код выше?

А выведет он следующее:

```
rub
```

```
usd
```

```
eur
```

То есть обход словаря в цикле будет возвращать только ключи этого словаря.

Пожалуй, задать такое поведение по умолчанию — это очень логичное решение со стороны разработчиков Python. Было бы намного внезапнее, если бы цикл по словарю получал значения. Вариант с кортежами (ключ, значение) в качестве поведения по умолчанию мне кажется не таким уж плохим, но имеем то, что имеем.

Есть куча задач, в которых нужно обойти лишь ключи словаря, и это отличное решение для таких задач. У этого способа есть один крупный недостаток: нужно знать как работают словари. По коду совершенно неясно, что будет обходиться в цикле — ключи, значения или пары, а читатель может либо этого не знать, либо забыть, и в итоге неправильно интерпретировать код. Поэтому во избежание неоднозначности даже для обхода ключей словаря я рекомендую использовать следующий способ.

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

Для получения значения конкретного ключа используются квадратные скобки []. Предположим, что в нашем словаре есть пара 'марафон': 26.

```
# берём значение с ключом "марафон"  
dictionary['марафон']
```

Опять же, вы получите ошибку, если попытаетесь получить значение по несуществующему ключу. Для избежания подобных ошибок существуют методы, о которых мы сейчас поговорим.

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

Методы словарей

clear() - очищает словарь.

copy() - возвращает копию словаря.

get(key[, default]) - возвращает значение ключа, но если его нет, не бросает исключение, а возвращает default (по умолчанию None).

items() - возвращает пары (ключ, значение).

keys() - возвращает ключи в словаре.

6. Что такое словарь включений?

Списковые включения в Python являются краткими синтаксическими конструкциями. Их можно использовать для создания списков из других списков, применяя функции к каждому элементу в списке. В этом разделе объясняется и демонстрируется использование этих выражений.

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.

В Python есть несколько встроенных функций, которые позволяют перебирать данные. Одна из них — `zip`. Функция `zip()` в Python создает итератор, который объединяет элементы из нескольких источников данных.

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.

В этом материале разберемся с основами этой функции, познакомимся с отличиями в Python 2 и Python 3, а также узнаем, в каких ситуациях `zip()` может быть полезна.

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

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. Так, он состоит из объектов следующих типов:

- `date` — хранит дату
- `time` — хранит время
- `datetime` — хранит дату и время

Как получить текущие дату и время?

С помощью модуля Python это сделать очень просто. Сначала нужно импортировать класс *datetime* из модуля *datetime*, после чего создать объект *datetime*. Модуль предоставляет метод `now()`, который возвращает текущие дату и время с учетом локальных настроек.