

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ**

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

Кафедра инфокоммуникаций

«Работа со словарями в языке Python»

**Отчет по лабораторной работе № 2.6
по дисциплине «Программирование на Python»**

Выполнил студент группы ИВТ-б-о-21-1

Сумин Никита Сергеевич.

« » _____ 2022г.

Подпись студента _____

Работа защищена « » _____ 20__ г.

Проверил Воронкин Р.А. _____
(подпись)

Ставрополь 2022

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

Порядок выполнения работы:

1. Создал общедоступный репозиторий на GitHub, в котором использована лицензия MIT и язык программирования Python.

Owner * Repository name *

BrandooDi / mywork2.6 ✓

Great repository names are short and memorable. Need inspiration? How about [urban-rotary-phone?](#)

Description (optional)

☒ Public
Anyone on the internet can see this repository. You choose who can commit.

☐ Private
You choose who can see and commit to this repository.

Initialize this repository with:
Skip this step if you're importing an existing repository.

☒ Add a README file
This is where you can write a long description for your project. [Learn more.](#)

Add .gitignore
Choose which files not to track from a list of templates. [Learn more.](#)

.gitignore template: Python ▼

Choose a license
A license tells others what they can and can't do with your code. [Learn more.](#)

License: MIT License ▼

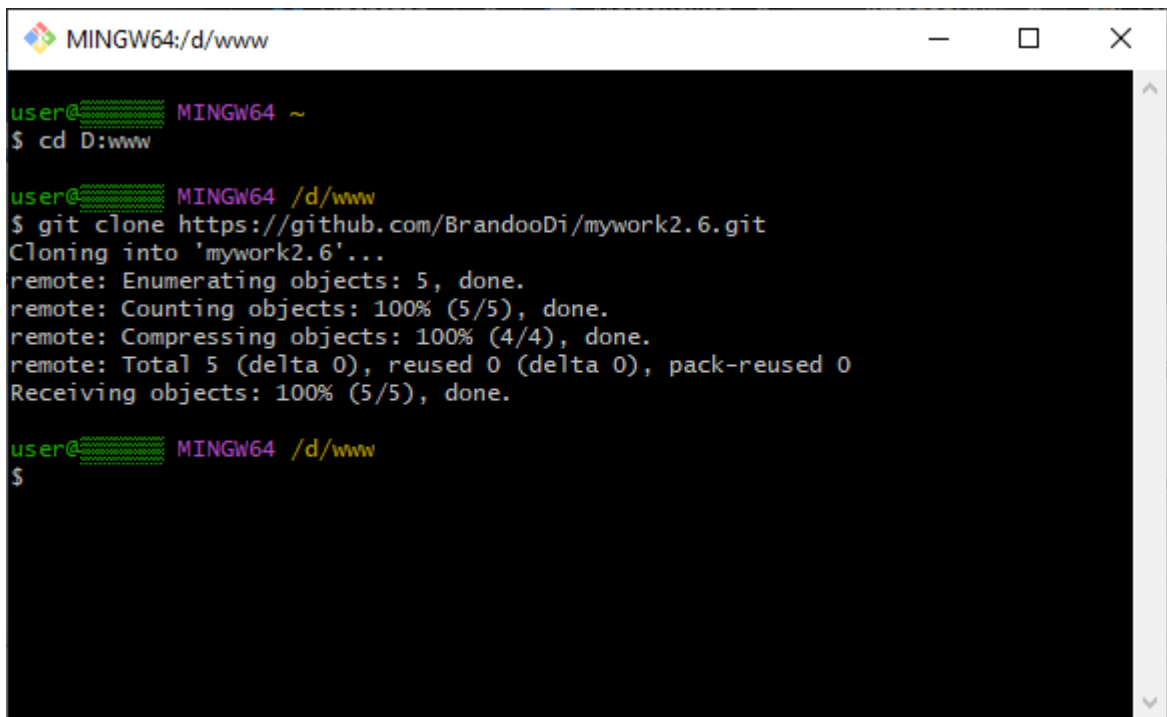
This will set `main` as the default branch. Change the default name in your [settings](#).

i You are creating a public repository in your personal account.

Create repository

Рисунок 1 - Создание репозитория

2. Выполните клонирование созданного репозитория.



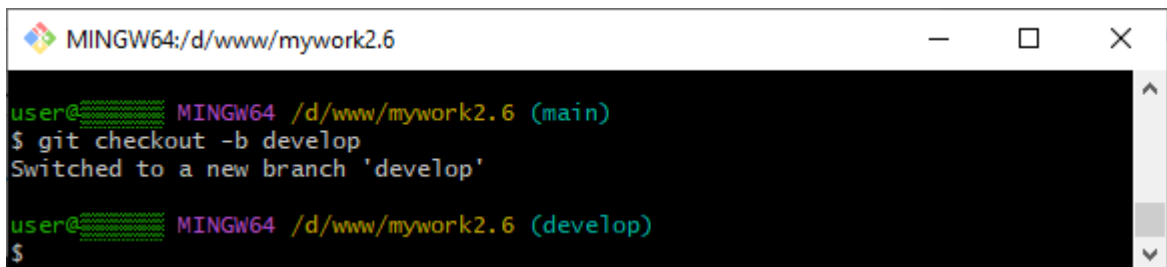
```
MINGW64:/d/www
user@MINGW64 ~
$ cd D:www

user@MINGW64 /d/www
$ git clone https://github.com/BrandooDi/mywork2.6.git
Cloning into 'mywork2.6'...
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 5 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (5/5), done.

user@MINGW64 /d/www
$
```

Рисунок 2 - Клонирование репозитория

3. Организуйте свой репозиторий в соответствии с моделью ветвления git-flow.



```
MINGW64:/d/www/mywork2.6
user@MINGW64 /d/www/mywork2.6 (main)
$ git checkout -b develop
Switched to a new branch 'develop'

user@MINGW64 /d/www/mywork2.6 (develop)
$
```

Рисунок 3 - Ветвление по модели git-flow 4.

Создайте проект PyCharm в папке репозитория.





	Project	29.10.2022 16:07	Папка с файлами	
	.gitignore	29.10.2022 15:59	Текстовый докум...	2 КБ
	LICENSE	29.10.2022 15:59	Файл	2 КБ
	README.md	29.10.2022 15:59	Файл "MD"	1 КБ

Рисунок 4 - Проект PyCharm

5. Проработайте пример лабораторной работы. Создайте для него отдельный модуль языка Python. Зафиксируйте изменения в репозитории.

```

D:\www\mywork2.6\Project\venv\Scripts\python.exe D:/www/mywork2.6/Project/main.py
>>> add
Фамилия и инициалы? Петров А.В
Должность? Менеджер
Год поступления? 2016
>>> add
Фамилия и инициалы? Иванов Н.К
Должность? Директор
Год поступления? 2020
>>> list
+-----+-----+-----+-----+
| No |          Ф.И.О.          |      Должность      |      Год      |
+-----+-----+-----+-----+
|  1 | Иванов Н.К              |      Директор       |      2020     |
|  2 | Петров А.В              |      Менеджер       |      2016     |
+-----+-----+-----+-----+
>>> |

```

Рисунок 5 - Результат выполнения примера

6. 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.

```

D:\www\mywork2.6\Project\venv\Scripts\python.exe D:/www/mywork2.6/Project/venv/zd1.py
{'1a': 19, '1б': 23, '2а': 20, '3в': 25, '6а': 17, '7б': 21, '9а': 18, '9в': 20}
Кол-во обучающихся: 161

Process finished with exit code 0

```

Рисунок 6 - Результат выполнения задачи 1

7. Решите задачу: создайте словарь, где ключами являются числа, а значениями – строки. Примените к нему метод items(), с с помощью полученного объекта dict_items создайте новый словарь, "обратный" исходному, т. е. ключами являются строки, а значениями – числа.

```
D:\www\mywork2.6\Project\venv\Scripts\python.exe D:/www/mywork2.6/Project/venv/zd2.py
{1: 'one', 2: 'two', 3: 'three', 4: 'four', 5: 'five'}
{'one': 1, 'two': 2, 'three': 3, 'four': 4, 'five': 5}

Process finished with exit code 0
```

Рисунок 7 - Результат выполнения задачи 2

8. Выполните 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..

```
Расчётный счёт плательщика: 0
Перечисляемая сумма в руб: 53452
>>> 1111
```

No	Расчётный счет получателя	Расчётный счёт плательщика	Перечисляемая сумма в руб
1	А	Б	2525
2	В	Г	543543
3	Ю	Я	53452

```
>>> |
```

Рисунок 8 - Результат выполнения индивидуального задания

Контрольные вопросы

1. Что такое словари в языке Python?

Словарь - это изменяемый (как список) неупорядоченный (в отличие от строк, списков и кортежей) набор элементов "ключ: значение".

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

Да, функция len() определяет длину словаря.

3. Какие методы обхода словарей Вам известны?

Элементы словаря перебираются в цикле `for` также, как элементы других сложных объектов. Однако "по-умолчанию" извлекаются только ключи. Но по ключам всегда можно получить значения.

С другой стороны у словаря как класса есть метод `items()`, который создает особую структуру, состоящую из кортежей. Каждый кортеж включает ключ и значение. В цикле `for` можно распаковывать кортежи, таким образом сразу извлекая как ключ, так и его значение.

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.

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

Обращением по ключу в квадратных скобках (`dict[key]`) или при помощи метода `get()`.

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

Обращением по ключу в квадратных скобках (`dict[key]`) или при помощи метода `setdefault()`, если элемент с указанным ключом отсутствует в списке.

6. Что такое словарь включений?

Словарь включений аналогичен списковым включениям, за исключением того, что он создаёт объект словаря вместо списка.

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.

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.

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