

Лабораторная работа 2.6

Тема: 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.

Порядок выполнения работы


1. Создаём аккаунт в GitHub. Затем создаём новый общедоступный репозиторий, в котором будет использована лицензия MIT и язык программирования Python.

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Required fields are marked with an asterisk (*).

Owner *

 RomanGorchakov ▾

Repository name *

/ Test

✓ Test is available.

Great repository names are short and memorable. Need inspiration? How about [automatic-octo-chainsaw](#) ?

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:

☐ Add a README file

This is where you can write a long description for your project. [Learn more about READMEs.](#)

Add .gitignore

.gitignore template: Python ▾

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

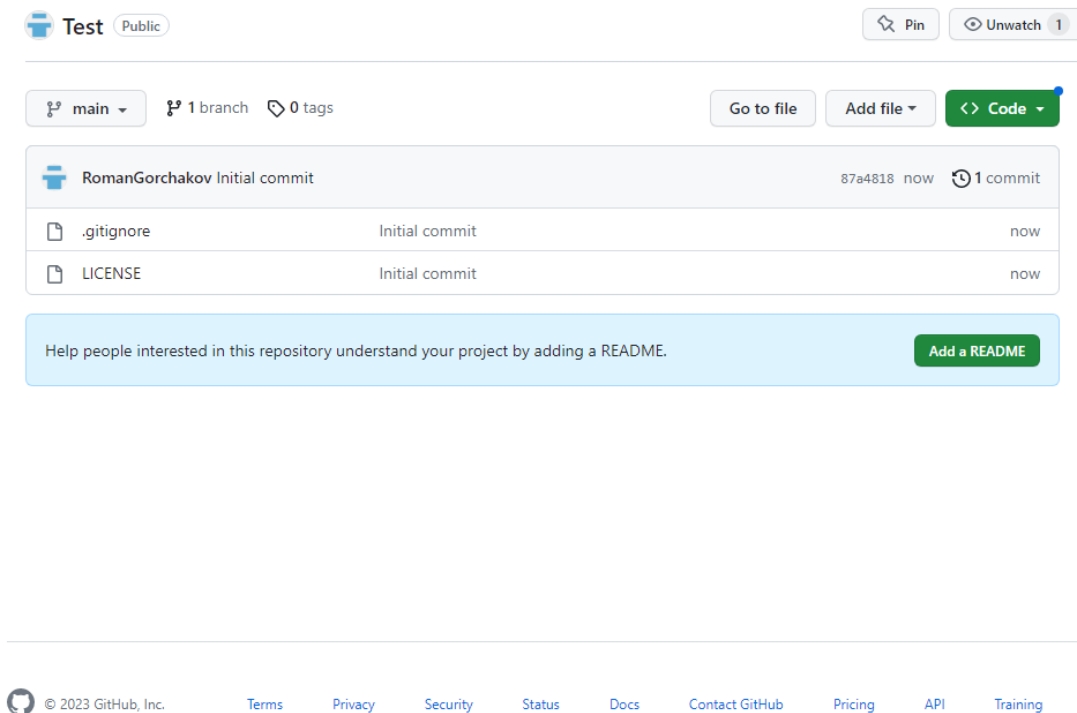
Choose a license

License: MIT License ▾


















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

 You are creating a public repository in your personal account.

Create repository



2. Теперь необходимо дополнить файл `.gitignore` с необходимыми правилами для языка программирования Python. Для этого переходим по ссылке «<https://github.com/github/gitignore>» и скачиваем оттуда файл «Python.gitignore».

	Phalcon.gitignore	Remove trailing asterisks in Phalcon rules	10 years ago
	PlayFramework.gitignore	Added /project/project to PlayFramework.gitignore	7 years ago
	Plone.gitignore	Covered by global vim template	10 years ago
	Prestashop.gitignore	Update for Prestashop 1.7 (#3261)	3 years ago
	Processing.gitignore	Ignore transpiled .java and .class files (#3016)	4 years ago
	PureScript.gitignore	Update PureScript adding .spago (#3278)	3 years ago
	Python.gitignore	Update Python.gitignore	last year
	Qooxdoo.gitignore	Add gitignore for qooxdoo apps	13 years ago
	Qt.gitignore	Remove trailing whitespace	2 years ago
	R.gitignore	Merge pull request #3792 from jl5000/patch-1	2 years ago
	README.md	Merge pull request #3854 from AnilSeervi/patch-1	2 years ago
	ROS.gitignore	Added ignore for files created by <code>catkin_make_isolated</code>	6 years ago
	Racket.gitignore	Update Racket.gitignore	2 years ago
	Rails.gitignore	Ignore Rails .env according recommendations	2 years ago
	Raku.gitignore	Changes the name of Perl 6 to Raku (#3312)	3 years ago
	RhodesRhomobile.gitignore	Add Rhodes mobile application framework gitignore	13 years ago
	Ruby.gitignore	Ruby: ignore RuboCop remote inherited config files (#3197)	4 years ago

```
1  # Byte-compiled / optimized / DLL files
2  __pycache__/
3  *.py[cod]
4  *$py.class
5
6  # C extensions
7  *.so
8
9  # Distribution / packaging
10 .Python
11 build/
12 develop-eggs/
13 dist/
14 downloads/
15 eggs/
16 .eggs/
17 lib/
18 lib64/
19 parts/
20 sdist/
21 var/
22 wheels/
23 share/python-wheels/
24 *.egg-info/
25 .installed.cfg
26 *.egg
27 MANIFEST
28
```

3. Теперь создаём файл «README.md», где вносим информацию о своей группе и ФИО. Сохраняем набранный текст через кнопку «Commit changes».



4. После этого нужно организовать репозиторий в соответствии с моделью ветвления Git-flow. Для этого В окне «Codespace» выбираем опцию «Create codespace on main», где введём команды: «git branch develop» и «git push -u origin develop» для создания ветки разработки; «git branch feature_branch» для создания ветки функций; «git branch release/1.0.0» для создания ветки релиза; «git checkout main» и «git branch hotfix» для создания веток hotfix.

```
@RomanGorchakov → /workspaces/Py6 (main) $ git checkout -b develop
Switched to a new branch 'develop'
● @RomanGorchakov → /workspaces/Py6 (develop) $ git branch feature_branch
● @RomanGorchakov → /workspaces/Py6 (develop) $ git branch release/1.0.0
● @RomanGorchakov → /workspaces/Py6 (develop) $ git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
● @RomanGorchakov → /workspaces/Py6 (main) $ git branch hotfix
● @RomanGorchakov → /workspaces/Py6 (main) $ git checkout develop
Switched to branch 'develop'
○ @RomanGorchakov → /workspaces/Py6 (develop) $
```

5. Создаём файл «example.py», в котором нужно использовать словарь, содержащий следующие ключи: фамилия и инициалы работника; название

занимаемой должности; год поступления на работу, и написать программу, выполняющую следующие действия:

- ввод с клавиатуры данных в список, состоящий из заданных словарей;
- записи должны быть размещены по алфавиту;
- вывод на дисплей фамилий работников, чей стаж работы в организации превышает значение, введенное с клавиатуры;
- если таких работников нет, вывести на дисплей соответствующее сообщение.

```
>>> add
фамилия и инициалы? Svetlana Vladimirovna
Должность? Police
Год поступления? 2004
>>> add
фамилия и инициалы? Dmitry Vladimirovich
Должность? Police
Год поступления? 1990
>>> list
+-----+-----+-----+-----+
| № | Ф.И.О. | Должность | Год |
+-----+-----+-----+-----+
| 1 | Dmitry Vladimirovich | Police | 1990 |
| 2 | Svetlana Vladimirovna | Police | 2004 |
+-----+-----+-----+-----+
>>> help
Список команд:

add - добавить работника;
list - вывести список работников;
select <стаж> - запросить работников со стажем;
help - отобразить справку;
exit - завершить работу с программой.
>>> exit

...Program finished with exit code 0
Press ENTER to exit console.
```

6. Создаём файл «school.py», в котором нужно создать словарь, связав его с переменной school, и наполнить данными, которые бы отражали количество учащихся в разных классах (1а, 1б, 2б, 6а, 7в и т. п.). Внести 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. Вычислить общее количество учащихся в школе.

```

Количество классов: 27
1a
1b
2a
2b
2c
3a
3b
3c
4a
4b
4c
5a
5b
5c
6a
6b
6c
7a
7b
7c
8a
8b
8c
9a
9b
9c
10
{'1a': 22, '1b': 27, '2a': 18, '2b': 28, '2c': 27, '3a': 15, '3b': 27,
 '3c': 19, '4a': 17, '4b': 16, '4c': 28, '5a': 22, '5b': 19, '5c': 28,
 '6a': 15, '6b': 15, '6c': 25, '7a': 23, '7b': 30, '7c': 30, '8a': 23,
 '8b': 25, '8c': 18, '9a': 27, '9b': 20, '9c': 29, '10': 25}
Название класса, в котором изменилось количество учеников: 6b
21
Название класса, который появился в школе: 11
21
Название класса, который был расформирован: 2c
Всего учеников в школе: 618

...Program finished with exit code 0
Press ENTER to exit console.

```

7. Создаём файл «reverse.py», в котором нужно создать словарь, где ключами являются числа, а значениями – строки. Применить к нему метод `items()` и с помощью полученного объекта `dict_items` создать новый словарь, «обратный» исходному.

```
{0: '3gtAmY4572V', 1: 'la1TWjMz', 2: '7dCbz9tv8laR9',
3: 'rT9HsSt6', 4: 'XAS4oDJdGr3', 5: 'enmQ5uhVqGH1I7O',
6: 'q4LReFhOKf', 7: '106UL79jVv', 8: 'zjOuwX'}
{'3gtAmY4572V': 0, 'la1TWjMz': 1, '7dCbz9tv8laR9': 2,
'rT9HsSt6': 3, 'XAS4oDJdGr3': 4, 'enmQ5uhVqGH1I7O': 5,
'q4LReFhOKf': 6, '106UL79jVv': 7, 'zjOuwX': 8}

...Program finished with exit code 0
Press ENTER to exit console.
```

8. Создаём файл «individual.py», в котором нужно использовать словарь, содержащий следующие ключи: название пункта назначения рейса; номер рейса; тип самолета. Написать программу, выполняющую следующие действия: ввод с клавиатуры данных в список, состоящий из словарей заданной структуры; записи должны быть упорядочены по возрастанию номера рейса; вывод на экран номеров рейсов и типов самолетов, вылетающих в пункт назначения, название которого совпало с названием, введенным с клавиатуры; если таких рейсов нет, выдать на дисплей соответствующее сообщение.

```
>>> add
Название пункта назначения рейса: stavropol
>>> add
Название пункта назначения рейса: krasnodarsk
>>> add
Название пункта назначения рейса: moscow
>>> add
Название пункта назначения рейса: st. petersburg
>>> add
Название пункта назначения рейса: kazan
>>> add
Название пункта назначения рейса: volgograd
>>> list
+-----+-----+-----+-----+
| No | Пункт назначения | Номер рейса | Тип самолёта |
+-----+-----+-----+-----+
| 1 | kazan | 9080 | 88 |
| 2 | krasnodarsk | 1485 | 20 |
| 3 | moscow | 1583 | 54 |
| 4 | st. petersburg | 7377 | 91 |
| 5 | stavropol | 5312 | 27 |
| 6 | volgograd | 2754 | 57 |
+-----+-----+-----+-----+
>>> select stavropol
1: stavropol
Номер рейса: 5312
Тип самолёта: 27
>>> exit

...Program finished with exit code 0
Press ENTER to exit console.
```

9. Выполняем коммит файлов в репозиторий Git в ветку разработки, сливаем её с веткой main и отправляем изменения на сервер GitHub.

```
@RomanGorchakov → /workspaces/Py6 (develop) $ git add .
@RomanGorchakov → /workspaces/Py6 (develop) $ git commit -m "Python files"
[develop 5d60ec0] Python files
4 files changed, 259 insertions(+)
create mode 100644 example.py
create mode 100644 individual.py
create mode 100644 reverse.py
create mode 100644 school.py
@RomanGorchakov → /workspaces/Py6 (develop) $ git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
@RomanGorchakov → /workspaces/Py6 (main) $ git merge develop
Updating ced3f74..5d60ec0
Fast-forward
 example.py | 111 +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
 individual.py | 99 +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
 reverse.py | 23 +++++++++++++++++++++++++++++++++++++
 school.py | 26 +++++++++++++++++++++
4 files changed, 259 insertions(+)
create mode 100644 example.py
create mode 100644 individual.py
create mode 100644 reverse.py
create mode 100644 school.py
@RomanGorchakov → /workspaces/Py6 (main) $ git push -u
Enumerating objects: 7, done.
Counting objects: 100% (7/7), done.
Delta compression using up to 2 threads
Compressing objects: 100% (6/6), done.
Writing objects: 100% (6/6), 3.26 KiB | 3.26 MiB/s, done.
Total 6 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/RomanGorchakov/Py6
ced3f74..5d60ec0 main -> main
branch 'main' set up to track 'origin/main'.
@RomanGorchakov → /workspaces/Py6 (main) $
```

main

1 Branch 0 Tags

Go to file

Add file

Code

RomanGorchakov Python files 5d60ec0 · 6 minutes ago 4 Commits

.gitignore	Create .gitignore	4 days ago
LICENSE	Create LICENSE	4 days ago
README.md	Create README.md	4 days ago
example.py	Python files	6 minutes ago
individual.py	Python files	6 minutes ago
reverse.py	Python files	6 minutes ago
school.py	Python files	6 minutes ago

README MIT license

Информация обо мне

Имя: Роман.

Фамилия: Горчаков.

Отчество: Владимирович.

Название группы: ПИЖ-6-о-22-1.

Контрольные вопросы

1. Что такое словари в языке Python?

Словарь представляет собой структуру данных (которая ещё называется ассоциативный массив), предназначенную для хранения произвольных объектов с доступом по ключу. Данные в словаре хранятся в формате ключ – значение.

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

Да, может.

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.

С другой стороны, у словаря как класса есть метод `items()`, который создает особую структуру, состоящую из кортежей. Каждый кортеж включает ключ и 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?

Метод `get()` позволяет получить элемент по его ключу. Метод `pop()` удаляет из словаря элемент по указанному ключу и возвращает значение удаленной пары.

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

Метод `fromkeys()` позволяет создать словарь из списка, элементы которого становятся ключами. Применять метод можно как классу `dict`, так и к его объектам. С помощью `setdefault()` можно добавить элемент в словарь.

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.

В Python функция `zip` позволяет пройти одновременно по нескольким итерируемым объектам.

```
a = [10, 20, 30, 40]
b = ['a', 'b', 'c', 'd', 'e']
for i, j in zip(a, b):
    print(i, j)
```

Здесь выражение `zip(a, b)` создает объект-итератор, из которого при каждом обороте цикла извлекается кортеж, состоящий из двух элементов. Первый берется из списка `a`, второй – из `b`.

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.