

Министерство науки и высшего образования Российской Федерации  
Федеральное государственное автономное образовательное учреждение  
высшего образования  
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт цифрового развития  
Кафедра инфокоммуникаций

**ОТЧЕТ**  
**по лабораторной работе №2\_7**  
**дисциплины**  
**«Основы программной инженерии»**

Выполнил:  
Разворотников Денис Сергеевич  
2 курс, группа ПИЖ-б-о-21-1,  
09.03.04 «Программная инженерия»,  
направленность (профиль) «Разработка  
и сопровождение программного  
обеспечения», очная форма обучения

---

(подпись)

Проверил:

---

(подпись)

Отчет защищен с оценкой \_\_\_\_\_ Дата защиты \_\_\_\_\_

Ставрополь 2022

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

1. Был создан репозиторий в Github в который были добавлены правила gitignore для работы IDE PyCharm, была выбрана лицензия MIT, сам репозиторий был клонирован на локальный сервер и был организован в соответствии с моделью ветвления git-flow.

DenisRazvorotnikov / lab2\_7 ✓

Great repository names are short and memorable. Need inspiration? How about **bookish-octo-bassoon**?

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 ▾

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

**Create repository**

Рисунок 1 – Создание репозитория

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
if __name__ == "__main__":
    # Определим универсальное множество
    u = set("abcdefghijklmnopqrstuvwxyz")
    a = {"b", "c", "h", "o"}
    b = {"d", "f", "g", "o", "v", "y"}
    c = {"d", "e", "j", "k"}
    d = {"a", "b", "f", "g"}

    x = (a.intersection(b)).union(c)
    print(f"x = {x}")

    # Найдем дополнения множеств
    bn = u.difference(b)
    cn = u.difference(c)

    y = (a.difference(d)).union(cn.difference(bn))
    print(f"y = {y}")
```

```
/usr/local/bin/python3 /Users/denisrazvorotnikov
x = {'d', 'j', 'k', 'e', 'o'}
y = {'f', 'y', 'c', 'h', 'o', 'g', 'v'}

Process finished with exit code 0
```

Рисунок 2 – Результат работы программы

### 3. Выполнил задания.

1) Решите задачу: подсчитайте количество гласных в строке, введенной с клавиатуры с использованием множеств.

#### Код программы:

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

if __name__ == " main ":
    # Все согл
    a = set("bcdfghilkmnpqrstvwxyz")
    mark = set(".?,:!;'\\" " ")

    # Ввод строки
    x = set(input("Введите строку: ").lower())

    # Находим все гласные в строке
    gl = x.difference(a)
    gl = gl.difference(mark)

    count = len(gl)

    print(f"Все гласные буквы из введенной строки: = {gl}")
    print(f"Кол-во гласных букв: = {count}")
```

```
Введите строку: qwertyasd
Все гласные буквы из введенной строки: = {'e', 'y', 'a'}
Кол-во гласных букв: = 3

Process finished with exit code 0
```

Рисунок 3 – Результат работы программы

2) Решите задачу: определите общие символы в двух строках, введенных с клавиатуры.

Код программы:

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
if __name__ == "main ":
    # Ввод строки
    x1 = set(input("Введите первую строку: ").lower())
    x2 = set(input("Введите вторую строку: ").lower())

    mark = set(".? , ! ; ' ` \" ")

    g1 = x1.intersection(x2)
    g1 = g1.difference(mark)

    count = len(g1)

    print(f"Общие символы: = {g1}")
    print(f"Кол-во общих символов: = {count}")
```

```
Введите первую строку: hello! Nice day
Введите вторую строку: Hello. Good
Общие символы: = {'l', 'o', 'd', 'h', 'e'}
Кол-во общих символов: = 5

Process finished with exit code 0
```

Рисунок 4 – Результат работы программы

## Индивидуальное задание

### Вариант 24

$$A = \{a, b, d, I, x\}; \quad B = \{d, e, h, i, n, u\}; \quad C = \{e, f, m, n\}; \quad D = \{a, c, h, k, r, s, w, x\};$$
$$X = (A/C) \cap \bar{B}; \quad Y = (\bar{A} \cap D) \cup (C/B).$$

$$A / C = \{'i', 'b', 'x', 'a', 'd'\}$$

$$X = (A / C) \cap B = \{'x', 'a', 'b'\}$$

$$\text{He}(A) = \{'o', 'm', 's', 'y', 'g', 'c', 'k', 'q', 'j', 'r', 'n', 't', 'w', 'v', 'u', 'z', 'h', 'l', 'e', 'p', 'f'\}$$

$$\text{He}(B) = \{'o', 's', 'm', 'y', 'c', 'w', 't', 'z', 'x', 'a', 'p', 'g', 'k', 'q', 'j', 'r', 'b', 'v', 'l', 'f'\}$$

$$\text{He}(C) = \{'o', 's', 'y', 'g', 'c', 'k', 'q', 'j', 'r', 't', 'i', 'w', 'b', 'v', 'u', 'z', 'h', 'l', 'x', 'a', 'p', 'd'\}$$

$$\text{He}(A) \cap D = \{'k', 'r', 'c', 'h', 'w', 's'\}$$

$$C / B = \{'m', 'f'\}$$

$$Y = (\text{He}(A) \cap D) \cup (C / B) = \{'h', 'w', 'm', 'c', 'k', 'r', 's', 'f'\}$$

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

if __name__ == "__main__":
    # Universe
    u = set("abcdefghijklmnopqrstuvwxyz")

    # Set Data
    a = {"a", "b", "d", "i", "x"}
    b = {"d", "e", "h", "i", "n", "u"}
    c = {"e", "f", "m", "n"}
    d = {"a", "c", "h", "k", "r", "s", "w", "x"}

    # Inverses for a b and c
    ne_a = u.difference(a)
    ne_b = u.difference(b)
    ne_c = u.difference(c)

    # Definition X
    x = ne_b.intersection(a.difference(c))
    print(f'X = {x}')

    # Definition Y
    y = (ne_a.intersection(d)).union(c.difference(b))
    print(f'Y = {y}')
```

```
/usr/local/bin/python3 /usr/s/sergey/azvotchnikov  
X = {'x', 'b', 'a'}  
Y = {'c', 'f', 'k', 's', 'h', 'w', 'r', 'm'}  
|  
Process finished with exit code 0
```

Рисунок 5 – Результат работы программы

```

[denisrazvorotnikov@Air-Denis lab2_7 % git add .
[denisrazvorotnikov@Air-Denis lab2_7 % git commit -m "Add file"
[develop 1aae00e] Add file
10 files changed, 117 insertions(+)
create mode 100644 .idea/.gitignore
create mode 100644 .idea/inspectionProfiles/profiles_settings.xml
create mode 100644 .idea/lab2_7.iml
create mode 100644 .idea/misc.xml
create mode 100644 .idea/modules.xml
create mode 100644 .idea/vcs.xml
create mode 100644 1.py
create mode 100644 2.py
create mode 100644 3.py
create mode 100644 idz.py
denisrazvorotnikov@Air-Denis lab2_7 %

```

Рисунок 6 – Коммит изменений

```

Your branch is up to date with 'origin/main'.
[denisrazvorotnikov@Air-Denis lab2_7 % git merge develop
Updating 6a7c8d2..1aae00e
Fast-forward
 .idea/.gitignore           | 3 +++
 .idea/inspectionProfiles/profiles_settings.xml | 6 ++++++
 .idea/lab2_7.iml           | 8 ++++++++
 .idea/misc.xml             | 7 +++++++
 .idea/modules.xml          | 8 ++++++++
 .idea/vcs.xml              | 6 ++++++
 1.py                       | 19 ++++++++++++++++++++++
 2.py                       | 19 ++++++++++++++++++++++
 3.py                       | 16 ++++++++++++++++++++
 idz.py                     | 25 ++++++++++++++++++++++
10 files changed, 117 insertions(+)
create mode 100644 .idea/.gitignore
create mode 100644 .idea/inspectionProfiles/profiles_settings.xml
create mode 100644 .idea/lab2_7.iml
create mode 100644 .idea/misc.xml
create mode 100644 .idea/modules.xml
create mode 100644 .idea/vcs.xml
create mode 100644 1.py
create mode 100644 2.py
create mode 100644 3.py
create mode 100644 idz.py
denisrazvorotnikov@Air-Denis lab2_7 %

```

Рисунок 7 – Слияние веток main и develop



### Контрольные вопросы:

#### 1. Что такое словари в языке Python?

Словари в Python – это изменяемые отображения ссылок на объекты, доступные по ключу.

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

Функция `len()` возвращает длину (количество элементов) в объекте. Аргумент может быть последовательностью, такой как строка, байты, кортеж, список или диапазон или коллекцией (такой как словарь, множество или неизменяемое множество).

### 3. Какие методы обхода словарей Вам известны?

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)`

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

С помощью метода `.get()`

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

С помощью функции `dict.update()`

### 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. Предположим, что есть список имен и номером сотрудников, и их нужно объединить в массив кортежей. Для этого можно использовать функцию `zip()`. Вот пример программы, которая делает именно это:

```
employee_numbers = [2, 9, 18, 28]
employee_names = ["Дима", "Марина", "Андрей", "Никита"]
zipped_values = zip(employee_names, employee_numbers)
zipped_list = list(zipped_values)
print(zipped_list)
```

Функция `zip` возвращает следующее:

```
[('Дима', 2), ('Марина', 9), ('Андрей', 18), ('Никита', 28)]
```

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?

```
import datetime  
dt_now = datetime.datetime.now()  
print(dt_now)
```

Результат:

```
2022-09-11 15:43:32.249588
```

Получить текущую дату:

```
from datetime import date  
current_date = date.today()  
print(current_date)
```

Результат:

```
2022-09-11
```

Получить текущее время:

```
import datetime
```

```
current_date_time = datetime.datetime.now()  
current_time = current_date_time.time()  
print(current_time)
```

Результат:

15:51:05.627643