МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное
образовательное учреждение высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Кафедра инфокоммуникаций

Отчет по лабораторной работе №2.6

Работа со словарями в языке Python

по дисциплине «Технологии программирования и алгоритмизация»

Выполнил
студент группы ИВТ-б-о-20-1
Дыбов Д.В. « »_____20__г.
Работа защищена « »_____20__г.
Проверил Воронкин Р.А. _____
(подпись)

Ставрополь 2021

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

Ход работы

Ссылка на репозиторий:

1. Создал новый репозиторий для лабораторной работы №2.6;
2. Клонировал созданный репозиторий на компьютер;
3. Создал новый PyCharm проект в папке репозитория;
4. Проработал пример:

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

import ...

if __name__ == '__main__':
    workers = []

    while True:
        command = input(">>> ").lower()

        if command == 'exit':
            break

        elif command == 'add':
            name = input("Фамилия и инициалы? ")
            post = input("Должность? ")
            year = int(input("Год поступления? "))

            worker = {
                'name': name,
                'post': post,
                'year': year,
            }

            workers.append(worker)
            if len(workers) > 1:
                workers.sort(key=lambda item: item.get('name', ''))

        elif command == 'list':
```

Рисунок 1 – Разобранный пример

5. Проверил пример на работоспособность;

```
>>> add
Фамилия и инициалы? Иванов И.И.
Должность? Стажёр
Год поступления? 2020
>>> add
Фамилия и инициалы? Петров А.С.
Должность? Директор
Год поступления? 2000
>>> list
+------+----------------------------------+----------------------+----------+
|  №   |             Ф.И.О.               |      Должность       |   Год    |
+------+----------------------------------+----------------------+----------+
|    1 | Иванов И.И.                      | Стажёр               |     2020 |
|    2 | Петров А.С.                      | Директор             |     2000 |
+------+----------------------------------+----------------------+----------+
>>> exit

Process finished with exit code 0
```

Рисунок 2 – Результат выполнения примера

6. Написал код для решения первой задачи;

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

from datetime import date
import sys

if __name__ == '__main__':
    school = {'1а': 33, '1б': 29, '2а': 15, '2б': 19, '3а': 35, '3б': 15, '4а': 22, '4б': 32}
    for i in range(1):
        school.update({input('Название изменяемого класса: '): int(
            input('Количество учеников изменяемого класса: '))})
        school.update({input('Название нового класса: '): int(
            input('Количество учеников нового класса: '))})
        del school[input('Название расформировываемого класса: ')]
        sc = sum(school[item] for item in school)
        print('Начальная школа:', school)
        print('Количество учеников в начальной школе:', sc)
```

Рисунок 3 – Написанный код

7. Проверил код на работоспособность и получил результат:

```
"F:\Основы Кроссплатформенного программирования\laba8\Scripts\python.exe" "C:/Program Files/Git/laba8/Zada4a1.py"
Название изменяемого класса: 2в
Количество учеников изменяемого класса: 30
Название нового класса: 4в
Количество учеников нового класса: 25
Название расформировываемого класса: 2б
Начальная школа: {'1а': 33, '1б': 29, '2а': 30, '3а': 35, '3б': 15, '4а': 22, '4б': 32, '4в': 25}
Количество учеников в начальной школе: 221
```

Рисунок 4 – Результат выполнения первой задачи

8. Решил вторую задачу;

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

from datetime import date
import sys

if __name__ == '__main__':
    Origin = {1: 'a', 2: 'b', 3: 'c'}
    LOL = Origin.items()
    Revers = {v: k for k, v in LOL}
    print(Revers)
```

Рисунок 5 – Решение второй задачи

9. Проверил код на работоспособность и получил результат:

```
"F:\Основы Кроссплатформенного програ
{'a': 1, 'b': 2, 'c': 3}

Process finished with exit code 0
```

Рисунок 6 – Результат выполнения второй задачи

10. Выполнил индивидуальное задание, согласно своему варианту;

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

from datetime import date
import sys

if __name__ == '__main__':
    Airplane = []

    while True:

        command = input(">>> ").lower()

        if command == 'exit':
            break

        elif command == 'add':
            path = input("Название пункта назначения рейса ")
            number = input("Номер рейса ")
            model = float(input("Тип самолёта "))

            Airplanes = {
                'path': path,
                'number': number,
                'model': model,
            }

            Airplane.append(Airplanes)
```

Рисунок 7 – Решение индивидуального задания

9. Проверил код на работоспособность и получил результат:

```
"F:\Основы Кроссплатформенного программирования\laba8\Scripts\python.exe" "C:/Program Files/Git/laba2.6/Individual.py"
>>> add
Название пункта назначения рейса Москва
Номер рейса CM-1245
Тип самолёта 2908
>>> add
Название пункта назначения рейса Санкт-Петербург
Номер рейса CC-327
Тип самолёта 09182
>>> add
Название пункта назначения рейса Екатеринбург
Номер рейса CE-12730
Тип самолёта 8172
>>> list
+----+--------------------------+--------------------+--------------------------+
| №  |     Пункт назначения     |     Номер рейса    |       Тип самолёта       |
+----+--------------------------+--------------------+--------------------------+
|  1 | Екатеринбург             | CE-12730           |                   8172.0 |
|  2 | Москва                   | CM-1245            |                   2908.0 |
|  3 | Санкт-Петербург          | CC-327             |                   9182.0 |
+----+--------------------------+--------------------+--------------------------+
```

Рисунок 8 – Результат выполнения индивидуального задания

8. С помощью сайта проверил первую задачу на наличие ошибок;
9. Результат выдал ошибку;

Python code

```
6
7  if __name__ == '__main__':
8      school = {'1а': 33, '1б': 29, '2а': 15, '2б': 19, '3а': 35, '3б': 15, '4а': 22, '4б': 32}
9      for i in range(1):
10         school.update({input(f'Название изменяемого класса : '): int(
11             input(f'Количество учеников изменяемого класса : '))})
12         school.update({input(f'Название нового класса  : '): int(
13             input(f'Количество учеников нового класса  : '))})
14         del school[input(f'Название расформировываемого класса: ')]
15         sc = sum(school[item] for item in school)
16         print(school)
17         print(sc)
18
```

```
Syntax errors detected :

Line 10:
school.update({input(f'\u041d\u0430\u0437\u0432\u0430\u043d\u0438\u0435
\u0438\u0437\u043c\u0435\u043d\u044f\u0435\u043c\u043e\u0433\u043e \u043a\u043b\
int(
^
SyntaxError: invalid syntax
```

Рисунок 9 – Проверка кода на наличие ошибок

10. Исправил ошибку и проверил код снова;
11. Результат показал, что ошибок в написании кода нет:

Python code

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

from datetime import date
import sys

if __name__ == '__main__':
    school = {'1а': 33, '1б': 29, '2а': 15, '2б': 19, '3а': 35, '3б': 15, '4а': 22, '4б': 32}
    for i in range(1):
        school.update({input('Название изменяемого класса : '): int(
            input('Количество учеников изменяемого класса : '))})
        school.update({input('Название нового класса  : '): int(
            input('Количество учеников нового класса  : '))})
```

No syntax errors detected :)

Рисунок 10 – Повторная проверка первой задачи на наличие ошибок

12. Проверил вторую задачу на наличие ошибок;
13. Результат показал, что код написан правильно:

Python code

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

from datetime import date
import sys

if __name__ == '__main__':
    Origin = {1: 'a', 2: 'b', 3: 'c'}
    LOL = Origin.items()
    Revers = {v: k for k, v in LOL}
    print(Revers)

```

No syntax errors detected :)

Рисунок 11 – Проверка второй задачи на наличие ошибок

8. Проверил индивидуальное задание на наличие ошибок;
9. Результат показал, что код написан правильно:

Python code

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

from datetime import date
import sys

if __name__ == '__main__':
    Airplane = []

    while True:

        command = input(">>> ").lower()
```

No syntax errors detected :)

Рисунок 12 – Проверка индивидуального задания на наличие ошибок

10. Сохранил все изменения в репозитории;

```
The file will have its original line endings in your working directory

C:\Program Files\Git\laba2.6>git commit -m "Add python files"
[main f7c0f1e] Add python files
 9 files changed, 238 insertions(+)
 create mode 100644 .idea/.gitignore
 create mode 100644 .idea/inspectionProfiles/profiles_settings.xml
 create mode 100644 .idea/laba8.iml
 create mode 100644 .idea/misc.xml
 create mode 100644 .idea/modules.xml
 create mode 100644 Individual.py
 create mode 100644 Primer1.py
 create mode 100644 Zada4a1.py
 create mode 100644 Zada4a2.py

C:\Program Files\Git\laba2.6>git push
Enumerating objects: 14, done.
Counting objects: 100% (14/14), done.
Delta compression using up to 8 threads
Compressing objects: 100% (12/12), done.
Writing objects: 100% (13/13), 3.57 KiB | 1.78 MiB/s, done.
Total 13 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), done.
To https://github.com/dibovdmitry/laba2.6.git
   29634f1..f7c0f1e  main -> main

C:\Program Files\Git\laba2.6>
```

Рисунок 13 – Сохранение изменений

Контрольные вопросы

1. Что такое словари в языке Python?

Словари в python – это структура данных, предназначенная для хранения произвольных объектов с доступом по ключу.

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

Функция len() может быть использована при работе со словарями. len() возвращает целое число, представляющее количество пар key:value в словаре.

3. Какие методы обхода словарей Вам известны?

Clear – удаляет все элементы словаря, оставляя сам словарь.

Copy – копирует словарь.

Fromkeys – позволяет создать словарь из списка, элементы которого становятся ключами.

Get – позволяет получить элементы по его ключу.

Pop – удаляет из словаря элемент по указанному ключу и возвращает произвольный элемент.

Popitem – не принимает аргументов, удаляет и возвращает произвольный элемент.

Setdefault – позволяет добавить элемент в словарь.

Update – позволяет добавить словарь в другой словарь.

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

Чтобы получить значения из словаря по ключу необходимо использовать метод keys.

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

Чтобы установить значение в словаре по ключу необходимо написать имя словаря, а за ним в квадратных скобках ключ.

6. Что такое словарь исключений?

Словарь исключений – это словарь, который содержит конструкции, используемые для сигнализации о важном событии, которое происходит при выполнении программы.

7. Самостоятельно изучите возможности функции *zip()* приведите примеры ее использования.

Функция zip() принимает итерируемый объект, например, список, кортеж, множество или словарь в качестве аргумента.

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

Этот модуль позволяет управлять датами и временем, представляя их в таком виде, в котором пользователи смогут их понимать.

С помощью модуля datetime можно:

– получить текущие дату и время;

– получить текущее время;

– получить текущую дату;

– создать объекты даты и времени;

– получить разницу для двух дат;

– получить прошлые и будущие даты;

– производить арифметические операции с датами.

– работать с часовыми поясами;

– конвертировать часовые пояса.

Вывод: в ходе выполнения лабораторной работы были приобретены навыки по работе со словарями при написании программ с помощью языка программирования Python версии 3.x.