

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

Кафедра инфокоммуникаций

Отчет по лабораторной работе №1

Работа со словарями в языке Python

По дисциплине «Технологии программирования и алгоритмизация»

Выполнил студент группы ИВТ-б-о-20-1

Галяс Д. И. « » _____ 20__ г.

Подпись студента _____

Работа защищена « » _____ 20__ г.

Проверил Воронкин Р. А. _____

(подпись)

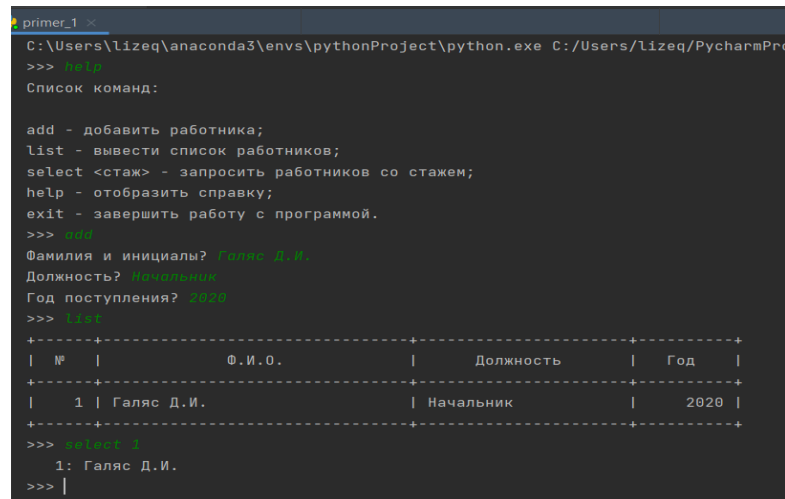
Ставрополь 2021

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

Ход работы:

Ссылка на репозиторий: <https://github.com/DIMITRY-GALYAS1/Rabota8.git>

1. Создал новый репозиторий на github, после клонировал его и создал в папке репозитория новый проект PyCharm.
2. Проработал примеры из лабораторной работы.

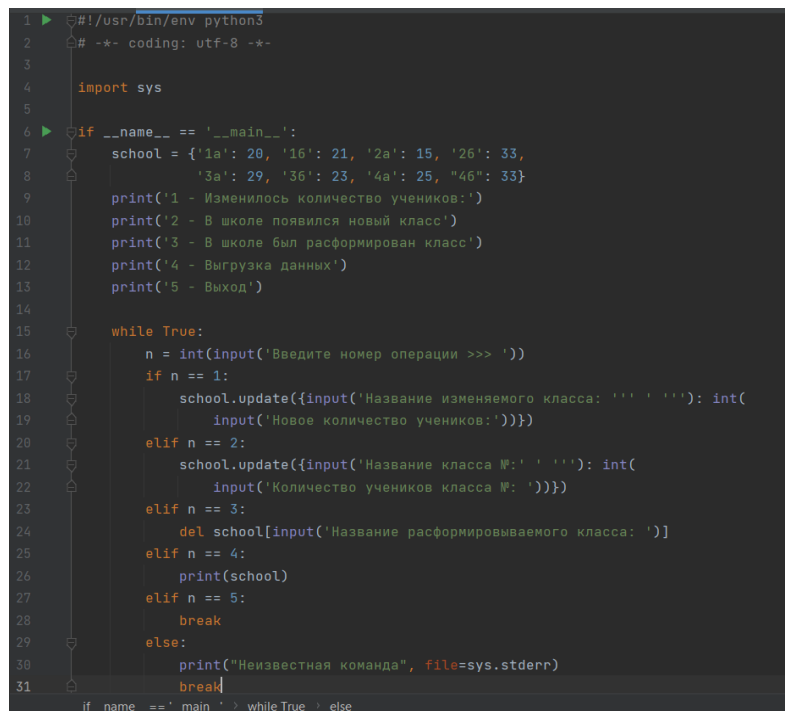


```
primer_1 >
C:\Users\lizeq\anaconda3\envs\pythonProject\python.exe C:/Users/Lizeq/PycharmPro
>>> help
Список команд:

add - добавить работника;
list - вывести список работников;
select <стаж> - запросить работников со стажем;
help - отобразить справку;
exit - завершить работу с программой.
>>> add
Фамилия и инициалы? Галяс Д.И.
Должность? Начальник
Год поступления? 2020
>>> list
+-----+-----+-----+-----+
| № |          Ф.И.О.          |      Должность      |      Год      |
+-----+-----+-----+-----+
|  1 | Галяс Д.И.              | Начальник           | 2020          |
+-----+-----+-----+-----+
>>> select 1
1: Галяс Д.И.
>>> |
```

Рисунок 1. Пример 1

3. Решил общие задания.



```
1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4 import sys
5
6 if __name__ == '__main__':
7     school = {'1a': 20, '1b': 21, '2a': 15, '2b': 33,
8              '3a': 29, '3b': 23, '4a': 25, '4b': 33}
9     print('1 - Изменилось количество учеников:')
10    print('2 - В школе появился новый класс')
11    print('3 - В школе был расформирован класс')
12    print('4 - Выгрузка данных')
13    print('5 - Выход')
14
15    while True:
16        n = int(input('Введите номер операции >>> '))
17        if n == 1:
18            school.update({input('Название изменяемого класса: '): int(
19                input('Новое количество учеников:'))})
20        elif n == 2:
21            school.update({input('Название класса №: '): int(
22                input('Количество учеников класса №:'))})
23        elif n == 3:
24            del school[input('Название расформировываемого класса: ')]
25        elif n == 4:
26            print(school)
27        elif n == 5:
28            break
29        else:
30            print("Неизвестная команда", file=sys.stderr)
31            break
32
33 if __name__ == '__main__': while True: else
```

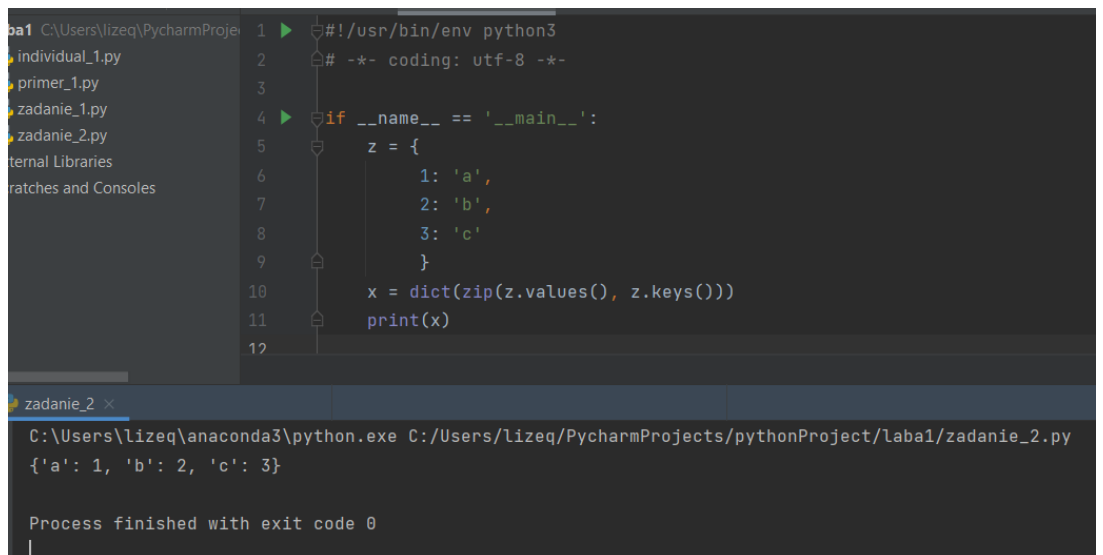
Рисунок 2. Код задания 1

```

C:\Users\lizeq\anaconda3\envs\pythonProject\python.exe C:/Users/lizeq/PycharmProjects/pythonProject/lab1/zadanie_1.py
1 - Изменилось количество учеников:
2 - В школе появился новый класс
3 - В школе был расформирован класс
4 - Выгрузка данных
5 - Выход
Введите номер операции >>> 1
Название изменяемого класса: 1a
Новое количество учеников: 27
Введите номер операции >>> 2
Название класса №: 11a
Количество учеников класса №: 22
Введите номер операции >>> 3
Название расформировываемого класса: 4a
Введите номер операции >>> 4
{'1a': 27, '16': 21, '2a': 15, '26': 33, '3a': 29, '36': 23, '46': 33, '11a': 22}
Введите номер операции >>>

```

Рисунок 3. Работа кода задания 1



```

C:\Users\lizeq\anaconda3\python.exe C:/Users/lizeq/PycharmProjects/pythonProject/lab1/zadanie_2.py
{'a': 1, 'b': 2, 'c': 3}

Process finished with exit code 0

```

Рисунок 4. Задание 2

4. Выполнил индивидуальное задание.

```

C:\Users\lizeq\anaconda3\envs\pythonProject\python.exe "C:/Users/lizeq/PycharmProjects/pythonProject/lab1/individual_1 novoe.py"
>>> nov
Фамилия и инициалы? Косимов К.С.
Номер группы? 2
Успеваемость? 3 4 5 5 5
>>> nov
Фамилия и инициалы? Козлов Д.М.
Номер группы? 1
Успеваемость? 2 3 5 5 5
>>> list
+-----+-----+-----+-----+
| № |          Ф.И.О.          | Номер группы | Успеваемость |
+-----+-----+-----+-----+
|  1 | Козлов Д.М.              | 1           | 2 3 5 5 5   |
|  2 | Косимов К.С.             | 2           | 3 4 5 5 5   |
+-----+-----+-----+-----+
>>> select
* Козлов Д.М. группа № 1
>>> exit
Process finished with exit code 0

```

Рисунок 5. Индивидуальное задание

Контрольные вопросы:

1. Что такое словари в языке Python?

Словарь (dict) 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. Словарь - это изменяемый неупорядоченный набор элементов "ключ: значение". "Неупорядоченный" – значит, что последовательность расположения пар не важна. Язык программирования ее не учитывает, в следствие чего обращение к элементам по индексам невозможно.

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

Функция len() может быть использована для получения количества пар ключ:значение. Она возвращает целое число, представляющее количество пар key:value в словаре.

3. Какие методы обхода словарей Вам известны?

Для обхода словарей существует несколько методов. Методы словаря keys() и values() позволяют получить отдельно перечни ключей и значений. Метод .items(), возвращает представление словаря, содержащее кортежи из двух элементов, вида (ключ, значение).

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

Операция dict[key] вернет элемент словаря dict с ключом key. Операция вызывает исключение KeyError, если ключ key отсутствует в словаре.

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

Операция dict[key] = value добавит в словарь новый элемент - пару ключ-значение. Если в словаре существует ключ key то эта операция присвоит ключу key новое значение value.

6. Что такое словарь включений?

Словарь включений аналогичен списковым включениям, за исключением того, что он создаёт объект словаря вместо списка. Как и в случае со списком, мы можем использовать условный оператор внутри словаря включения, чтобы получить только элементы словаря, удовлетворяющие заданному критерию.

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

Функция zip() позволяет создать словарь путем объединения списков ключей и значений.

Создание словаря из списков ключей и значений

```
Numbers = dict(zip([1, 2, 3], ['One', 'Two', 'Three']))
```

```
print(Numbers) # {1: 'One', 2: 'Two', 3: 'Three'}
```

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.

Обязательные аргументы:

- `datetime.MINYEAR (1) ≤ year ≤ datetime.MAXYEAR (9999)`

- `1 ≤ month ≤ 12`

- `1 ≤ day ≤ количество дней в данном месяце и году`

- Методы класса `datetime`:

- `datetime.today()` - объект `datetime` из текущей даты и времени. Работает также, как и `datetime.now()` со значением `tz=None`.

- `datetime.fromtimestamp(timestamp)` - дата из стандартного представления времени.

- `datetime.fromordinal(ordinal)` - дата из числа, представляющего собой количество дней, прошедших с 01.01.1970.

- `datetime.now(tz=None)` - объект `datetime` из текущей даты и времени.
- `datetime.combine(date, time)` - объект `datetime` из комбинации объектов `date` и `time`.
- `datetime.strptime(date_string, format)` - преобразует строку в `datetime` (так же, как и функция `strptime` из модуля `time`).
- `datetime.strftime(format)` - см. функцию `strftime` из модуля `time`.
- `datetime.date()` - объект даты (с отсечением времени).
- `datetime.time()` - объект времени (с отсечением даты).
- `datetime.replace([year[, month[, day[, hour[, minute[, second[, microsecond[, tzinfo]]]]]]])` - возвращает новый объект `datetime` с изменёнными атрибутами.
- `datetime.timetuple()` - возвращает `struct_time` из `datetime`.
- `datetime.toordinal()` - количество дней, прошедших с 01.01.1970.
- `datetime.timestamp()` - возвращает время в секундах с начала эпохи.
- `datetime.weekday()` - день недели в виде числа, понедельник - 0, воскресенье - 6.
- `datetime.isoweekday()` - день недели в виде числа, понедельник - 1, воскресенье - 7.
- `datetime.isocalendar()` - кортеж (год в формате ISO, ISO номер недели, ISO день недели).
- `datetime.isoformat(sep='T')` - красивая строка вида "YYYY-MM-DDTHH:MM:SS.mmmmmm" или, если `microsecond == 0`, "YYYY-MM-DDTHH:MM:SS"
- `datetime.ctime()` - см. `ctime()` из модуля `time`.

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