Лабораторная работа 2.6 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.*

Ход работы

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. Если вспомнить такую структуру как список, то доступ к его элементам осуществляется по индексу, который представляет собой целое неотрицательное число, причем мы сами, непосредственно, не участвуем в его создании (индекса). В словаре аналогом индекса является ключ, при этом ответственность за его формирование ложится на программиста.

В языке программирования Python словари (тип `dict`) представляют собой еще одну разновидность структур данных наряду со списками и кортежами. *Словарь - это **изменяемый** (как список) **неупорядоченный** (в отличие от строк, списков и кортежей) 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.

В других языках структуры, схожие со словарями, называются по-другому. Например, в Java подобный тип данных называется отображением.

Чтобы представление о словаре стало более понятным, проведем аналогию с обычным словарем, например, англо-русским. На каждое английское слово в таком словаре есть русское слово-перевод: cat – кошка, dog – собака, table – стол и т. д. Если англо-русский словарь описать с помощью Python, то английские слова можно сделать ключами, а русские – их значениями:

```
{'cat': 'кошка', 'dog': 'собака',
'bird': 'птица', 'mouse': 'мышь'}
```

Обратите внимание на фигурные скобки, именно с их помощью определяется словарь. Синтаксис словаря на Питоне описывается такой схемой:

Часто при выводе словаря последовательность пар "ключ: значение" не совпадает с тем, как было введено:

```
>>> a = {'cat': 'кошка', 'dog': 'собака',
... 'bird': 'птица', 'mouse': 'мышь'}
>>> a
{'dog': 'собака', 'cat': 'кошка',
'bird': 'птица', 'mouse': 'мышь'}
```

Поскольку в словаре не важен порядок пар, то интерпретатор выводит их так, как ему удобно. Тогда как получить доступ к определенному элементу, если индексация не возможна в принципе? В словаре доступ к значениям осуществляется по ключам, которые заключаются в квадратные скобки (по аналогии с индексами списков):

```
>>> a['cat']
'кошка'
>>> a['bird']
'птица'
```

Словари, как и списки, являются изменяемым типом данных: позволительно изменять, добавлять и удалять элементы (пары "ключ: значение"). Изначально словарь можно создать пустым (например, `d = {}`) и потом заполнить его элементами. Добавление и изменение имеет одинаковый синтаксис: `словарь[ключ] = значение`. Ключ может быть как уже существующим (тогда происходит изменение значения), так и новым (происходит добавление элемента словаря). Удаление элемента осуществляется с помощью встроенной оператора `del` языка Python.

```
>>> a['elephant'] = 'бегемот' # добавляем
>>> a['table'] = 'стол' # добавляем
>>> a
{'dog': 'собака', 'cat': 'кошка', 'mouse': 'мышь',
'bird': 'птица', 'table': 'стол',
'elephant': 'бегемот'}
>>> a['elephant'] = 'слон' # изменяем
>>> del a['table'] # удаляем
>>> a
{'dog': 'собака', 'cat': 'кошка', 'mouse': 'мышь',
'bird': 'птица', 'elephant': 'слон'}
```

В словаре не может быть двух элементов с одинаковыми ключами. Однако могут быть одинаковые значения у разных ключей.

Ключом может быть любой неизменяемый тип данных. Значением – любой тип данных. Значения словарей вполне могут быть структурами, например, другими словарями или списками.

```
>>> nums = {1: 'one', 2: 'two', 3: 'three'}
>>> person = {'name': 'Tom', 1: [30, 15, 16],
... 2: 2.34, ('ab', 100): 'no'}
```

Перебор элементов словаря в цикле `for`

Элементы словаря перебираются в цикле `for` также, как элементы других сложных объектов. Однако "по-умолчанию" извлекаются только ключи:

```
>>> nums
{1: 'one', 2: 'two', 3: 'three'}
>>> for i in nums:
...     print(i)
...
1
2
3
```

Но по ключам всегда можно получить значения:

```
>>> for i in nums:
...     print(nums[i])
...
one
two
three
```

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

```
>>> n = nums.items()
>>> n
dict_items([(1, 'one'), (2, 'two'), (3, 'three')])
```

В цикле for можно распаковывать кортежи, таким образом сразу извлекая как ключ, так и его значение:

```
>>> for key, value in nums.items():
...     print(key, 'is', value)
...
1 is one
2 is two
3 is three
```

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:

```
>>> v_nums = []
>>> for v in nums.values():
...     v_nums.append(v)
...
>>> v_nums
['one', 'two', 'three']
```

То же самое можно сделать с помощью списковых включений:

```
>>> v_nums = [v for v in nums.values]
```

Методы словаря

Кроме рассмотренных выше трех методов *items()*, *keys()* и *values()* словари обладают еще восемью. Это методы *clear()*, *copy()*, *fromkeys()*, *get()*, *pop()*, *popitem()*, *setdefault()*, *update()*.

Метод *clear()* удаляет все элементы словаря, но не удаляет сам словарь. В итоге остается пустой словарь:

```
>>> a
{'dog': 'собака', 'cat': 'кошка', 'mouse': 'мышь',
'bird': 'птица', 'elephant': 'слон'}
>>> a.clear()
>>> a
{}
```

Словарь – это изменяемый тип данных. Следовательно, как и список он передается в функцию по ссылке. Поэтому иногда, чтобы избежать нежелательного изменения глобального словаря его копируют. Это делают и с другими целями.

```
>>> nums2 = nums.copy()
>>> nums2[4] = 'four'
>>> nums
{1: 'one', 2: 'two', 3: 'three'}
>>> nums2
{1: 'one', 2: 'two', 3: 'three', 4: 'four'}
```

Метод *fromkeys()* позволяет создать словарь из списка, элементы которого становятся ключами. Применять метод можно как классу *dict*, так и к его объектам:

```
>>> a = [1, 2, 3]
>>> c = dict.fromkeys(a)
>>> c
{1: None, 2: None, 3: None}
>>> d = dict.fromkeys(a, 10)
>>> d
{1: 10, 2: 10, 3: 10}
>>> c
{1: None, 2: None, 3: None}
```

Метод *get()* позволяет получить элемент по его ключу:

```
>>> nums.get(1)
'one'
```

Равносильно `nums[1]`, если ключ присутствует в словаре. Если ключ отсутствует в словаре, то выражение `словарь[ключ]` приведет к возникновению исключительной ситуации, тогда как выражение `словарь.get(ключ, значение)` в этом случае вернет `значение`, по умолчанию `значение` равно *None*.

Метод *pop()* удаляет из словаря элемент по указанному ключу и возвращает значение удаленной пары. Метод *popitem()* не принимает аргументов, удаляет и возвращает произвольный элемент.

```
>>> nums.pop(1)
'one'
>>> nums
{2: 'two', 3: 'three'}
>>> nums.popitem()
(2, 'two')
>>> nums
{3: 'three'}
```

С помощью setdefault() можно добавить элемент в словарь:

```
>>> nums.setdefault(4, 'four')
'four'
>>> nums
{3: 'three', 4: 'four'}
```

Равносильно `nums[4] = 'four'`, если элемент с ключом 4 отсутствует в словаре. Если он уже есть, то `nums[4] = 'four'` перезапишет старое значение, *setdefault()* – нет.

С помощью *update()* можно добавить в словарь другой словарь:

```
>>> nums.update({6: 'six', 7: 'seven'})
>>> nums
{3: 'three', 4: 'four', 6: 'six', 7: 'seven'}
```

Также метод обновляет значения существующих ключей. Включает еще ряд особенностей.

Словарь включений

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

Основной пример:

```
>>> {x: x * x for x in (1, 2, 3, 4)}
{1: 1, 2: 4, 3: 9, 4: 16}
```

это просто еще один способ написания:

```
>>> dict((x, x * x) for x in (1, 2, 3, 4))
{1: 1, 2: 4, 3: 9, 4: 16}
```

Как и в случае со списком, мы можем использовать условный оператор внутри словаря включения, чтобы получить только элементы словаря, удовлетворяющие заданному критерию.

```
>>> {name: len(name) for name in ('Stack', 'Overflow', 'Exchange') if len(name)
> 6}
{'Exchange': 8, 'Overflow': 8}
```

Или переписать с помощью генераторного выражения.

```
>>> dict((name, len(name)) for name in ('Stack', 'Overflow', 'Exchange') if
len(name) > 6)
{'Exchange': 8, 'Overflow': 8}
```

Начиная со словаря и используя словарь в качестве фильтра пары ключ-значение

```
>>> initial_dict = {'x': 1, 'y': 2}
>>> {key: value for key, value in initial_dict.items() if key == 'x'}
{'x': 1}
```

Переключение ключа и значения словаря (инвертировать словарь)

```
>>> my_dict = {1: 'a', 2: 'b', 3: 'c'}
```

если вы хотели поменять местами ключи и значения, вы можете использовать несколько подходов в зависимости от вашего стиля кодирования:

```
>>> swapped = {v: k for k, v in my_dict.items()}
>>> swapped = dict((v, k) for k, v in my_dict.items())
>>> swapped = dict(zip(my_dict.values(), my_dict))
>>> swapped = dict(zip(my_dict.values(), my_dict.keys()))
>>> swapped = dict(map(reversed, my_dict.items()))

>>> print(swapped)
{a: 1, b: 2, c: 3}
```

Объединение словарей

Объедините словари и при необходимости переопределите старые значения с помощью вложенного словаря включений.

```
>>> dict1 = {'w': 1, 'x': 1}
>>> dict2 = {'x': 2, 'y': 2, 'z': 2}
>>> {k: v for d in [dict1, dict2] for k, v in d.items()}
{'w': 1, 'x': 2, 'y': 2, 'z': 2}
```

Пример 1. *Использовать словарь, содержащий следующие ключи: фамилия и инициалы работника; название занимаемой должности; год поступления на работу. 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.*

Решение: Определим следующие ключи для словарей:

- *name* - фамилия и инициалы работника;
- *post* - название занимаемой должности;
- *year* - год поступления.

Введем следующие команды для работы со списком словарей в интерактивном режиме:

- *add* - запросить информацию о сотруднике с клавиатуры и добавить в список, поддерживая список в отсортированном состоянии;
- *list* - вывести на экран содержимое списка словарей;
- *select* - вывести на дисплей фамилий работников, чей стаж работы в организации превышает заданное значение, при этом это значение должно быть аргументом команды *select* и отделено от нее пробелом;
- *help* - вывести на дисплей список команд с описанием;
- *exit* - завершить работу программы.

Работа с программой осуществляется следующим образом. Вначале с помощью команды *add* добавляется некоторое число работников. Контроль ввода осуществляется при помощи команды *list*. После чего можно выбирать работников с помощью команды *select*, например, для выбора всех работников, у которых стаж работы более 10 лет необходимо ввести команду:

```
>>> select 10
```

Напишем программу для решения поставленной задачи.

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

import sys
from datetime import date


if __name__ == '__main__':
    # Список работников.
    workers = []

    # Организовать бесконечный цикл запроса команд.
    while True:
        # Запросить команду из терминала.
        command = input(">>> ").lower()

        # Выполнить действие в соответствие с командой.
        if command == 'exit':
            break

        elif command == 'add':
            # Запросить данные о работнике.
            name = input("Фамилия и инициалы? ")
            post = input("Должность? ")
            year = int(input("Год поступления? "))

            # Создать словарь.
            worker = {
                'name': name,
                'post': post,
                'year': year,
            }

            # Добавить словарь в список.
            workers.append(worker)
            # Отсортировать список в случае необходимости.
            if len(workers) > 1:
                workers.sort(key=lambda item: item.get('name', ''))

        elif command == 'list':
            # Заголовок таблицы.
            line = '+-{}-+-{}-+-{}-+-{}-+'.format(
                '-' * 4,
                '-' * 30,
                '-' * 20,
                '-' * 8
            )
            print(line)
            print(
                '| {:^4} | {:^30} | {:^20} | {:^8} |'.format(
                    "№",
                    "Ф.И.О.",
```

```
                "Должность",
                "Год"
            )
        )
        print(line)

        # Вывести данные о всех сотрудниках.
        for idx, worker in enumerate(workers, 1):
            print(
                '| {:>4} | {:<30} | {:<20} | {:>8} |'.format(
                    idx,
                    worker.get('name', ''),
                    worker.get('post', ''),
                    worker.get('year', 0)
                )
            )

        print(line)

    elif command.startswith('select '):
        # Получить текущую дату.
        today = date.today()

        # Разбить команду на части для выделения номера года.
        parts = command.split(' ', maxsplit=1)
        # Получить требуемый стаж.
        period = int(parts[1])

        # Инициализировать счетчик.
        count = 0
        # Проверить сведения работников из списка.
        for worker in workers:
            if today.year - worker.get('year', today.year) >= period:
                count += 1
                print(
                    '{:>4}: {}'.format(count, worker.get('name', ''))
                )

        # Если счетчик равен 0, то работники не найдены.
        if count == 0:
            print("Работники с заданным стажем не найдены.")

    elif command == 'help':
        # Вывести справку о работе с программой.
        print("Список команд:\n")
        print("add - добавить работника;")
        print("list - вывести список работников;")
        print("select <стаж> - запросить работников со стажем;")
        print("help - отобразить справку;")
        print("exit - завершить работу с программой.")

    else:
        print(f"Неизвестная команда {command}", file=sys.stderr)
```

Необходимо обратить внимание на строку `workers.sort(key=lambda item: item.get('name', ''))`, в которой осуществляется сортировка списка по заданному ключу. Для этого метод *sort()* списка использует параметр `key`, значение которого равно лямбда-выражению (с лямбда-выражениями мы познакомимся при изучении функций), позволяющему выбрать из словаря фамилию и инициалы работника.

Для получения текущего номера года использованы возможности встроенного модуля *datetime*. Для этого вначале запрашивается текущая дата посредством вызова метода *today()* класса *date*. После чего у полученного объекта `today` производится чтение атрибута *year*, который и будет содержать номер текущего года.

Также необходимо обратить внимание на использование метода `format` для вывода данных в виде таблицы и нумерованного списка.

Аппаратура и материалы

1. Компьютерный класс общего назначения с конфигурацией ПК не хуже рекомендованной для ОС Windows 10 с подключением к глобальной сети Интернет.
2. Операционная система Windows 10.
3. Система контроля версий Git.
4. Браузер для доступа к web-сервису GitHub, рекомендован к использованию Google Chrome.
5. Дистрибутив языка программирования Python, включающий набор популярных библиотек Anaconda.
6. Интегрированная среда разработки PyCharm Community Edition.

Указания по технике безопасности

При работе на ЭВМ без разрешения руководителя занятия запрещается:

- подавать (снимать) напряжение на ПЭВМ и электрические розетки с распределительного щита;
- включать и выключать блоки питания ПЭВМ и мониторы;
- извлекать ПЭВМ из защитного кожуха;
- устранять неисправности, возникшие в ходе выполнения лабораторной работы.

Методика и порядок выполнения работы

1. Изучить теоретический материал работы.
2. Создать общедоступный репозиторий на GitHub, в котором будет использована лицензия MIT и язык программирования Python.
3. Выполните клонирование созданного репозитория.
4. Дополните файл `.gitignore` необходимыми правилами для работы с IDE PyCharm.
5. Организуйте свой репозиторий в соответствие с моделью ветвления git-flow.
6. Создайте проект PyCharm в папке репозитория.
7. Проработайте пример лабораторной работы. Создайте для него отдельный модуль языка Python. Зафиксируйте изменения в репозитории.
8. Приведите в отчете скриншоты результатов выполнения примера при различных исходных данных вводимых с клавиатуры.
9. 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.

10. Зафиксируйте сделанные изменения в репозитории.
11. 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.
12. Зафиксируйте сделанные изменения в репозитории.
13. Приведите в отчете скриншоты работы программ и UML-диаграммы деятельности решения индивидуального задания.
14. Зафиксируйте сделанные изменения в репозитории.
15. Добавьте отчет по лабораторной работе в *формате PDF* в папку *doc* репозитория. Зафиксируйте изменения.
16. Выполните слияние ветки для разработки с веткой *main*/*master*.
17. Отправьте сделанные изменения на сервер GitHub.
18. Отправьте адрес репозитория GitHub на электронный адрес преподавателя.

Индивидуальное задание

Составить программу с использованием списков и словарей для решения задачи. Номер варианта определяется по согласованию с преподавателем.

1. Использовать словарь, содержащий следующие ключи: фамилия и инициалы; номер группы; успеваемость (список из пяти элементов). Написать программу, выполняющую следующие действия: ввод с клавиатуры данных в список, состоящий из словарей заданной структуры; записи должны быть упорядочены по возрастанию номера группы; вывод на дисплей фамилий и номеров групп для всех студентов, включенных в массив, если средний балл студента больше 4.0; если таких студентов нет, вывести соответствующее сообщение.
2. Использовать словарь, содержащий следующие ключи: фамилия и инициалы; номер группы; успеваемость (список из пяти элементов). Написать программу, выполняющую следующие действия: ввод с клавиатуры данных в список, состоящий из словарей заданной структуры; записи должны быть упорядочены по возрастанию среднего балла; вывод на дисплей фамилий и номеров групп для всех студентов, имеющих оценки 4 и 5; если таких студентов нет, вывести соответствующее сообщение.
3. Использовать словарь, содержащий следующие ключи: фамилия и инициалы; номер группы; успеваемость (список из пяти элементов). Написать программу, выполняющую следующие действия: ввод с клавиатуры данных в список, состоящий из словарей заданной структуры; записи должны быть упорядочены по алфавиту; вывод на дисплей фамилий и номеров групп для всех студентов, имеющих хотя бы одну оценку 2; если таких студентов нет, вывести соответствующее сообщение.
4. 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.
5. 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 размещены в алфавитном порядке по названиям пунктов назначения; вывод на экран пунктов назначения и номеров рейсов, обслуживаемых самолетом, тип которого введен с клавиатуры; если таких рейсов нет, выдать на дисплей соответствующее сообщение.
6. Использовать словарь, содержащий следующие ключи: название пункта назначения; номер поезда; время отправления. Написать программу, выполняющую следующие действия: ввод

с клавиатуры данных в список, состоящий из словарей заданной структуры; записи должны быть размещены в алфавитном порядке по названиям пунктов назначения; вывод на экран информации о поездах, отправляющихся после введенного с клавиатуры времени; если таких поездов нет, выдать на дисплей соответствующее сообщение.

7. Использовать словарь, содержащий следующие ключи: название пункта назначения; номер поезда; время отправления. Написать программу, выполняющую следующие действия: ввод с клавиатуры данных в список, состоящий из словарей заданной структуры; записи должны быть упорядочены по времени отправления поезда; вывод на экран информации о поездах, направляющихся в пункт, название которого введено с клавиатуры; если таких поездов нет, выдать на дисплей соответствующее сообщение.
8. Использовать словарь, содержащий следующие ключи: название пункта назначения; номер поезда; время отправления. Написать программу, выполняющую следующие действия: ввод с клавиатуры данных в список, состоящий из словарей заданной структуры; записи должны быть упорядочены по номерам поездов; вывод на экран информации о поезде, номер которого введен с клавиатуры; если таких поездов нет, выдать на дисплей соответствующее сообщение.
9. Использовать словарь, содержащий следующие ключи: название начального пункта маршрута; название конечного пункта маршрута; номер маршрута. Написать программу, выполняющую следующие действия: ввод с клавиатуры данных в список, состоящий из словарей заданной структуры; записи должны быть упорядочены по номерам маршрутов; вывод на экран информации о маршруте, номер которого введен с клавиатуры; если таких маршрутов нет, выдать на дисплей соответствующее сообщение.
10. Использовать словарь, содержащий следующие ключи: название начального пункта маршрута; название конечного пункта маршрута; номер маршрута. Написать программу, выполняющую следующие действия: ввод с клавиатуры данных в список, состоящий из словарей заданной структуры; записи должны быть упорядочены по номерам маршрутов; вывод на экран информации о маршрутах, которые начинаются или оканчиваются в пункте, название которого введено с клавиатуры; если таких маршрутов нет, выдать на дисплей соответствующее сообщение.
11. Использовать словарь, содержащий следующие ключи: фамилия, имя; номер телефона; дата рождения (список из трех чисел). Написать программу, выполняющую следующие действия: ввод с клавиатуры данных в список, состоящий из словарей заданной структуры; записи должны быть упорядочены по датам рождения; вывод на экран информации о человеке, номер телефона которого введен с клавиатуры; если такого нет, выдать на дисплей соответствующее сообщение.
12. Использовать словарь, содержащий следующие ключи: фамилия, имя; номер телефона; дата рождения (список из трех чисел). Написать программу, выполняющую следующие действия: ввод с клавиатуры данных в список, состоящий из словарей заданной структуры; записи должны быть размещены по алфавиту; вывод на экран информации о людях, чьи дни рождения приходятся на месяц, значение которого введено с клавиатуры; если таких нет, выдать на дисплей соответствующее сообщение.
13. Использовать словарь, содержащий следующие ключи: фамилия, имя; номер телефона; дата рождения (список из трех чисел). Написать программу, выполняющую следующие действия: ввод с клавиатуры данных в список, состоящий из словарей заданной структуры; записи должны быть упорядочены по трем первым цифрам номера телефона; вывод на экран информации о человеке, чья фамилия введена с клавиатуры; если такого нет, выдать на дисплей соответствующее сообщение.
14. Использовать словарь, содержащий следующие ключи: фамилия, имя; знак Зодиака; дата рождения (список из трех чисел). Написать программу, выполняющую следующие действия: ввод с клавиатуры данных в список, состоящий из словарей заданной структуры; записи должны быть упорядочены по датам рождения; вывод на экран информации о человеке, чья

фамилия введена с клавиатуры; если такого нет, выдать на дисплей соответствующее сообщение.

15. Использовать словарь, содержащий следующие ключи: фамилия, имя; знак Зодиака; дата рождения (список из трех чисел). Написать программу, выполняющую следующие действия: ввод с клавиатуры данных в список, состоящий из словарей заданной структуры; записи должны быть упорядочены по датам рождения; вывод на экран информации о людях, родившихся под знаком, название которого введено с клавиатуры; если таких нет, выдать на дисплей соответствующее сообщение.
16. Использовать словарь, содержащий следующие ключи: фамилия, имя; знак Зодиака; дата рождения (массив из трех чисел). Написать программу, выполняющую следующие действия: ввод с клавиатуры данных в список, состоящий из словарей заданной структуры; записи должны быть упорядочены по знакам Зодиака; вывод на экран информации о людях, родившихся в месяц, значение которого введено с клавиатуры; если таких нет, выдать на дисплей соответствующее сообщение.
17. Использовать словарь, содержащий следующие ключи: название товара; название магазина, в котором продается товар; стоимость товара 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 названиям
    товаров; вывод на экран информации о товаре, название которого введено с клавиатуры; если таких товаров нет, выдать на дисплей соответствующее сообщение.
18. Использовать словарь, содержащий следующие ключи: название товара; название магазина, в котором продается товар; стоимость товара 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 названиям магазинов; вывод на экран информации о товарах, продающихся в магазине, название которого введено с клавиатуры; если такого магазина нет, выдать на дисплей соответствующее сообщение.
19. 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.

Содержание отчета и его форма

Отчет по лабораторной работе оформляется электронно в формате PDF, должен содержать ответы на контрольные вопросы, ссылку на репозиторий с которым выполнялась работа, скриншоты IDE PyCharm, скриншоты результатов работы программ.

Вопросы для защиты работы

1. Что такое словари в языке Python?
2. Может ли функция *len()* быть использована при работе со словарями?
3. Какие методы обхода словарей Вам известны?
4. Какими способами можно получить значения из словаря по ключу?
5. Какими способами можно установить значение в словаре по ключу?
6. Что такое словарь включений?
7. Самостоятельно изучите возможности функции *zip()* приведите примеры ее использования.

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