Отчет по лабораторной работе № 3
по курсу
"Разработка интернет приложений"

" Python-классы "

Бодунов А.Г., ИУ5-53

Москва, МГТУ - 2016 год

Задание

Вход:
username или vk_id пользователя

Выход:
Гистограмма распределения возрастов друзей пользователя, поступившего на вход

Пример:
Вход:
reigning

Выход:
19 #
20 ##
21 ##
22 ####################
23 ################
24 ####
25 #
28 #
29 #
30 #
37 #
38 ##
45 #

Указания

За основу возьмите базовый класс:
https://gist.github.com/Abashinos/024c1dcaf92f1ff733c63a07e447ab51

Для реализации методов ВК наследуйтесь от этого базового класса. Создайте один класс для получения id пользователя из username и один для получения и обработки списка друзей. В классах-наследниках необходимо реализовать методы:

- get_params - если есть get параметры (необязательно).
- get_json - если нужно передать post данные (необязательно).
- get_headers - если нужно передать дополнительные заголовки (необязательно).
- response_handler - обработчик ответа. В случае успешного ответа необходим, чтобы преобразовать результат запроса. В случае ошибочного ответа необходим, чтобы сформировать исключение.
- _get_data - внутренний метод для отправки http запросов к VK API.

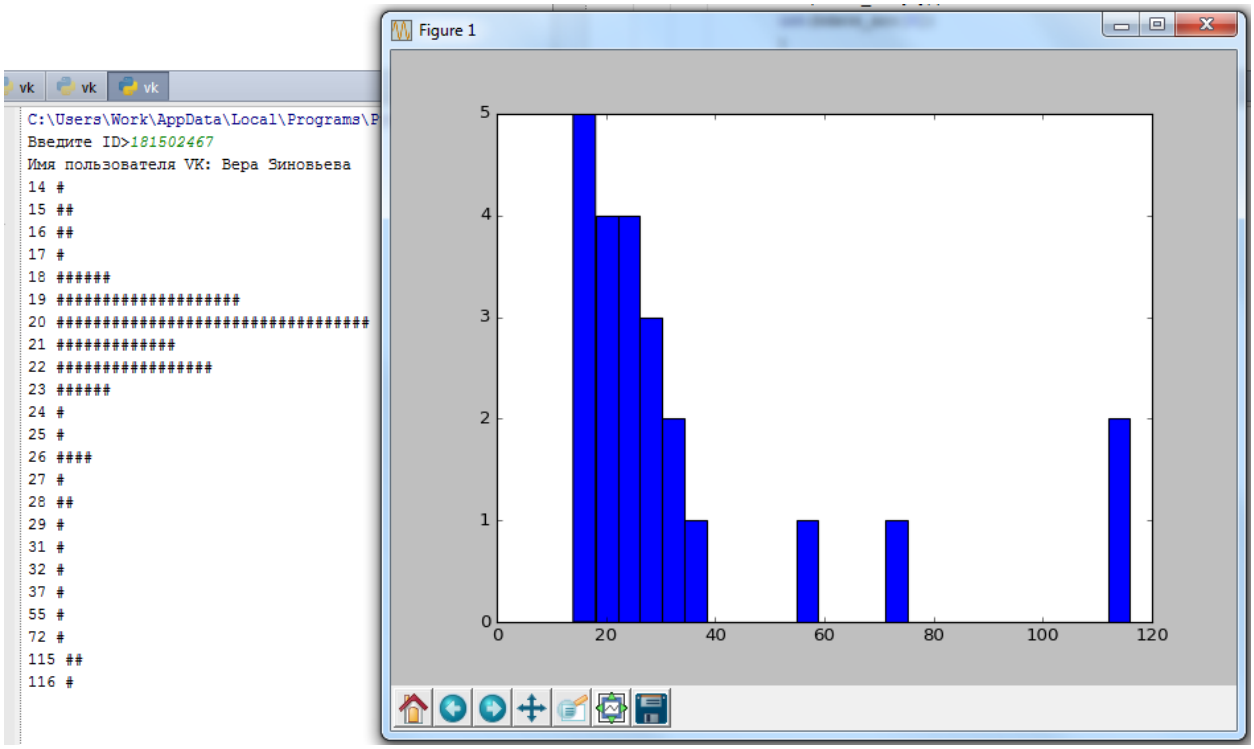
Для решения задачи нужно обратиться к двум методам VK API

1) users.get - для получения vk id по username
2) friends.get - для получения друзей пользователя. В этом методе нужно передать в get параметрах fields=bdate для получения возраста. Нужно принять во внимание, что не у всех указана дата рождения

Описание методов можно найти тут:
https://vk.com/dev/methods

```
import BaseClient
import requests
import json
import datetime


class VkClient(BaseClient.BaseClient):
    def __init__(self, method):
        self.BASE_URL = 'https://api.vk.com/method/'
        self.method = method
        self.http_method = 'get'

    def get_json(self):
        pass

    def get_dict_data(self):#получаем словарь в результате запроса к серверу
        dict_data = json.loads(self.execute())
        return dict_data

    def _get_data(self, method, http_method):#
        response = None
        if http_method == 'get':
            response = requests.get(self.generate_url(method))
        return self.response_handler(response)

    def response_handler(self, response):#
        dict_data = ""
        dict_data = json.loads(response.text)
        if "error" in dict_data.keys():
            raise Exception
        return response.text


def create_args(args):#создаем словарь
    return "&".join("%s=%s" % (key, value) for key, value in args.items())


def check_bdatt(obj):#проверяем, полученное значение даты существует, и если существует, задан ли год
    return ("bdate" in obj.keys()) and (len(obj["bdate"].split('.')) == 3)
```

Результаты работы программы:




```python
if __name__ == "__main__":
    name = input("Введите ID>")#ввод id пользователя через консоль
    try:
        vk_name = VkClient('users.get?user_ids=' + name)#создаем объект класса VkClient с параметром id пользователя
        name = vk_name.get_dict_data()#вызываем метод класса VkClient
        if (name is None) or ('uid' not in name['response'][0]):
            raise Exception
    except Exception:
        print('Вы ввели некорректную информацию')
        exit()

    # Выведем имя и фамилию пользователя
    print('Имя пользователя VK: ' + name['response'][0]['first_name'] +
          ' ' + name['response'][0]['last_name'])

    # Сегодняшняя дата (для вычисления возраста)
    now = datetime.date.today()
    # Словарь для кол-ва людей определенного возраста
    old_arr = {}#словарь, в котором ключи возраста,а значения-количество людей данного возраста

    # Аргументы для выборки друзей
    args = {
        "user_id": name['response'][0]['uid'],
        "fields": "bdate",
        "v": "5.57"
    }

    try:
        vk = VkClient('friends.get?' + create_args(args))#получаем список друзей в виде словаря
        tmp = vk.get_dict_data()["response"]["items"]#
    except Exception:
        print('Что-то произошло при попытке получить пользователя')
        exit()
```

```
    for x in tmp:
        if check_bdatt(x):#если указан год рождения
            bdate_arr = x["bdate"].split('.')#разбиваем дату рождения по точочкам
            bdate = datetime.date(#преобразуем дату
                    int(bdate_arr[2]),
                    int(bdate_arr[1]),
                    int(bdate_arr[0])
                    )
            diff = int((now - bdate).days / 365)#получаем возраст
            if diff not in old_arr.keys():#если такого возраста еще не было
                old_arr[diff] = 0#добавляем его
            old_arr[diff] = old_arr[diff] + 1#еще один человек определенного возраста

    # массив для графика
    new_arr=[]
    # Выведем в более менее приличном формате
    for key in sorted(old_arr):
        print(str(key) + ' ' + '#'.join('' for i in range(old_arr[key] + 1)))# выводим возраст + столько #, сколько людей такого возраста
        new_arr.append(key)#добавляем возраст в массив для вывода графика

import matplotlib.pyplot as plt

plt.hist(
    new_arr, # в зависимости от количества 1,2,3 строится гистограмма
    25 # а это как бы длина оси x
    )

plt.show()
```