**Московский государственный технический университет им. Н.Э. Баумана**

# Разработка интернет-приложений
# Лабораторная работа № 3

“Python – Классы”

С доп. заданием

Выполнил:
студент группы ИУ5-53
Сметанкин К.И.
Подпись:
Дата:

Москва 2017г.

# 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

Разнесите базовый класс, классы наследники и основную программу в разные модули.
Про модули можно прочитать тут:
https://docs.python.org/3/tutorial/modules.html
https://habrahabr.ru/post/166463/

Для выполнения запросов нужно использовать библиотеку *requests*
http://docs.python-requests.org/en/master/

Для обработки дат (дней рождения) используйте встроенную библиотеку *datetime*
https://docs.python.org/3/library/datetime.html

Чтобы установить библиотеку используйте пакетным менеджером *pip*
https://pip.pypa.io/en/stable/quickstart/

Подсказки:

1. Метод get библиотеки requests принимает вторым аргументом словарь get-параметров.
2. Не забывайте, что в классах-наследниках можно перегружать статические поля наследуемого класса.

# Дополнительное задание

Постройте гистограмму с использованием *matplotlib*
http://matplotlib.org/examples/statistics/histogram_demo_features.html

**Файл Base_Client.py**

```python
import requests
import datetime

class BaseClient:
    # URL vk api
    BASE_URL = None
    # метод vk api
    method = None
    # GET, POST, ...
    http_method = None

    # Получение GET параметров запроса
    def get_params(self):
        return None

    # Получение данных POST запроса
    def get_json(self):
        return None

    # Получение HTTP заголовков
    def get_headers(self):
        return None

    # Склейка url
    def generate_url(self, method):
        return '{0}{1}'.format(self.BASE_URL, method)

    # Отправка запроса к VK API
    def _get_data(self, method, http_method):
        response = None
        # todo выполнить запрос
        return self.response_handler(response)

    # Обработка ответа от VK API
    def response_handler(self, response):
        return response

    # Запуск клиента
    def execute(self):
        return self._get_data(
            self.method,
            http_method=self.http_method
        )
```

## Файл UserId.py

```python
import requests
from Base_Client import BaseClient
import datetime
import json


class UserId(BaseClient):
    BASE_URL = 'http://api.vk.com/method/'
    method = 'users.'
    http_method = 'get'

    def __init__(self, name):
        self.name = name

    def get_params(self):
        return 'user_ids=' + self.name

    def response_handler(self, response):
        a = json.loads(response.text)
        b = a['response'][0]
        return b['id']

    def _get_data(self, method, http_method):
        response = None
        response = requests.get(self.BASE_URL + method + http_method + '?' + self.get_params() + '&v=5.68')
        return self.response_handler(response)
```

## Файл Friends.py

```python
import datetime
import json
from datetime import datetime
import requests
from Base_Client import BaseClient
import numpy as np


class Friends(BaseClient):
    BASE_URL = 'http://api.vk.com/method/'
    method = 'friends.'
    http_method = 'get'

    def __init__(self, id):
        self.id = str(id)

    def get_params(self):
        return 'user_id=' + self.id

    def response_handler(self, response):
        ages = {}
        cur_date = datetime.now()
        a = json.loads(response.text)
        e = int(a['response']['count'])
        for i in range(e):
            d = a['response']['items'][i]
            try:
                datel = datetime.strptime(d['bdate'], '%d.%m.%Y')
                delta = (cur_date - datel).days
                age = delta // 365.25
                if age not in ages:
                    ages[age] = ''
                ages[age] += '#'
            except:
                pass
        return ages

    def _get_data(self, method, http_method):
        response = None
        response = requests.get(self.BASE_URL + method + http_method + '?' + self.get_params() + '&fields=bdate&v=5.68')
        # print(response)
        return self.response_handler(response)

    def execute_for_math(self):
        return self._get_data_math(
            self.method,
            http_method=self.http_method
        )
```

```python
            http_method=self.http_method
        )

    def _get_data_math(self, method, http_method):
        response = None
        response = requests.get(self.BASE_URL + method + http_method + '?' + self.get_params() + '&fields=bdate&v=5.68')
        # print(response)
        return self.response_handler_math(response)

    def response_handler_math(self, response):
        ages = []
        cur_date = datetime.now()
        a = json.loads(response.text)
        e = int(a['response']['count'])
        for i in range(e):
            d = a['response']['items'][i]
            try:
                datel = datetime.strptime(d['bdate'], '%d.%m.%Y')
                delta = (cur_date - datel).days
                age = delta // 365.25
                ages.append(age)
            except:
                continue
        return ages
        # np.array(ages)
```

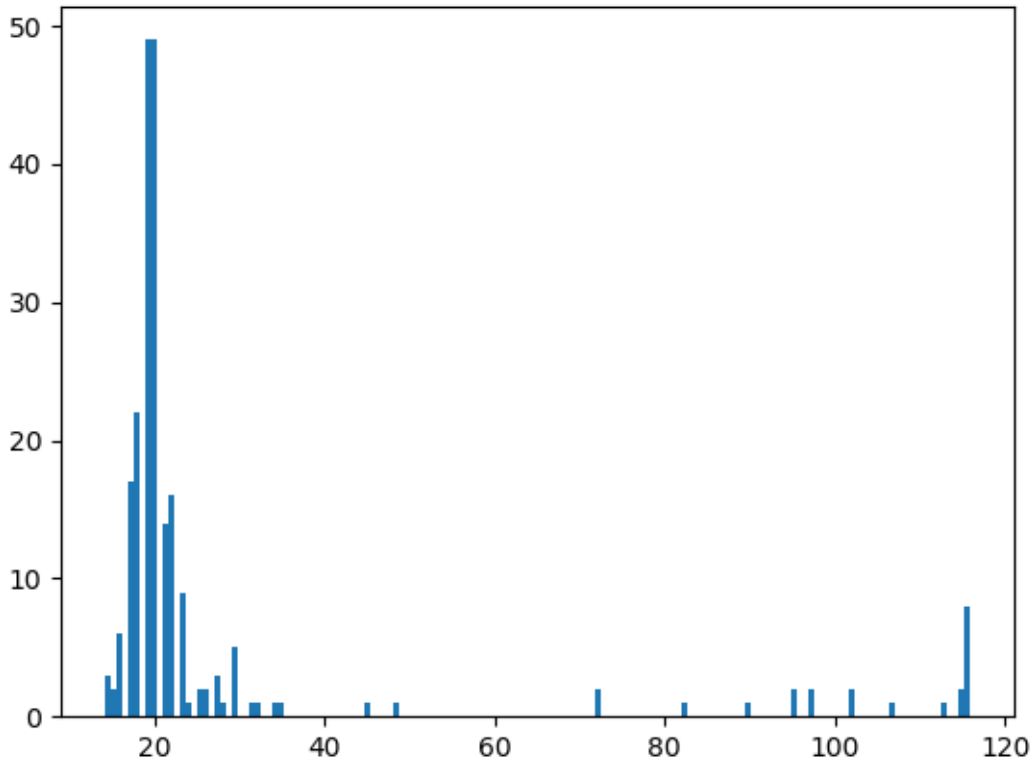
**Файл lab3.py**

```python
from UserId import UserId
from Friends import Friends
import matplotlib.pyplot as plt
import numpy as np


if __name__ == '__main__':
    # обычная гистограмма
    name = UserId('taron997').execute()
    print(name)
    fr = Friends(name).execute()
    statistic = sorted(fr.items(), key=lambda x: x[0])
    for i in statistic:
        print('{} => {}'.format(i[0], i[1]))

    # гистограмма matplotlib
    fr_math = Friends(name).execute_for_math()
    # print(fr_math)
    plt.hist(fr_math, 150)
    plt.show()
```

**Результат выполнения программы**

**Гистограмма распределения возрастов друзей пользователя**

```
"C:\Program Files (x86)\Python36-32\python.exe" "D:/Учеба/5 сем/РИП/lab3/lab3.py"
14377480
14.0 => ###
15.0 => ##
16.0 => ######
17.0 => ################
18.0 => ######################
19.0 => ##################################################
20.0 => ##################################################
21.0 => ##############
22.0 => ################
23.0 => #########
24.0 => #
25.0 => ##
26.0 => ##
27.0 => ###
28.0 => #
29.0 => #####
31.0 => #
32.0 => #
34.0 => #
35.0 => #
45.0 => #
48.0 => #
72.0 => ##
82.0 => #
90.0 => #
95.0 => ##
97.0 => ##
102.0 => ##
107.0 => #
113.0 => #
115.0 => ##
116.0 => ########
```


50
40
30
20
10
0
20
40
60
80
100
120