Московский государственный технический

университет им. Н.Э. Баумана

Факультет «Информатика и системы управления» Кафедра

«Cистемы обработки информации и управления»



# “Разработка интернет- приложений”

**«Python-классы»**

**Лабораторная работа № 3**

Студент группы ИУ5-53

\_\_\_\_\_\_\_\_\_\_\_\_Атаманов В. В.

Преподаватель

\_\_\_\_\_\_\_\_\_\_\_\_ Гапанюк Е. Ю.

**Москва 2017**

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

## Код программы: base\_client.py

**import** requests

**class** BaseClient:

*# URL vk api*

BASE\_URL = **"https://api.vk.com/method/"**

*# 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):

**if** http\_method == **'GET'**:

response = requests.get(**'{0}?{1}'**.format(self.generate\_url(method), self.get\_params()))

**elif** http\_method == **'POST'**:

response = requests.post(self.generate\_url(method), data=self.get\_json())

**else**:

**return None**

**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

)

## **Код программы**: friends.py

**from** base\_client **import** BaseClient

**from** json **import** loads

**class** Friends(BaseClient):

*# метод vk api*

method = **"friends.get"**

*# GET, POST, ...*

http\_method = **"GET"**

**def** \_\_init\_\_(self, user\_id):

self.\_user\_id = user\_id

**def** get\_params(self):

**return 'user\_id={0}&fields=bdate'**.format(self.\_user\_id)

**def** response\_handler(self, response):

**return** loads(response.text)[**'response'**]

## **Код программы**: user\_id.py

**from** base\_client **import** BaseClient

**from** json **import** loads

**class** UserId(BaseClient):

*# метод vk api*

method = **"users.get"**

*# GET, POST, ...*

http\_method = **"GET"**

**def** \_\_init\_\_(self, username):

self.\_username = username

**def** get\_params(self):

**return 'uids='** + self.\_username

**def** response\_handler(self, response):

data = loads(response.text)[**'response'**]

**return** data[0].get(**'uid'**)

## **Код программы**: main.py

**from** datetime **import** datetime, date

**from** user\_id **import** UserId

**from** friends **import** Friends

**from** random **import** randint, normalvariate

**import** matplotlib.pyplot **as** plt

**if** \_\_name\_\_ == **"\_\_main\_\_"**:

ages\_dct = dict()

get\_user\_id = UserId(input())

get\_friends = Friends(get\_user\_id.execute())

it\_dict = get\_friends.execute()

**for** f **in** range(100000):

**if** f.get(**'bdate'**) **is None**:

**continue**

bdate = f[**'bdate'**].split(**'.'**)

**if** len(bdate) != 3:

**continue**

age = datetime.now(tz=**None**).date() - date(int(bdate[2]), int(bdate[1]), int(bdate[0]))

age = int(age.days / 365)

**if** ages\_dct.get(age) **is None**:

ages\_dct[age] = 1

**else**:

ages\_dct[age] += 1

x\_axis = []

y\_axis = []

items = sorted(ages\_dct.items(), key=**lambda** item: item[0])

**for** key, value **in** items:

x\_axis.append(key)

y\_axis.append(value)

print(key, end=**' '**)

**for** i **in** range(value):

print(**'#'**, end=**''**)

print()

plt.bar(x\_axis, y\_axis, align=**'center'**)

plt.show()

## **Результаты работы:**



