|  |  |  |
| --- | --- | --- |
|  |  |  |

**Отчет по лабораторной работе № 3 по курсу**

**РИП**

#### Тема работы: "Python­ - классы"

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |

Студент группы ИУ 5-51ц

Щипицин Р.А.

Москва, МГТУ - 2016

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# Описание задания лабораторной работы.

# 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 #

# 2. Порядок работы.

За основу возьмите базовый класс: https://gist.github.com/alexopryshko/a9ffec925d2a5ecb4f731a0ac77479f8

Для реализации методов ВК наследуйтесь от этого базового класса. В классе наследнике необходимо реализовать методы:

● get\_params ­ если есть get параметры (необязательно)

● get\_json ­ если нужно передать данные (необязательно)

● get\_headers ­ если нужно передать дополнительные заголовки (необязательно)

● response\_handler ­ обработчик ответа.

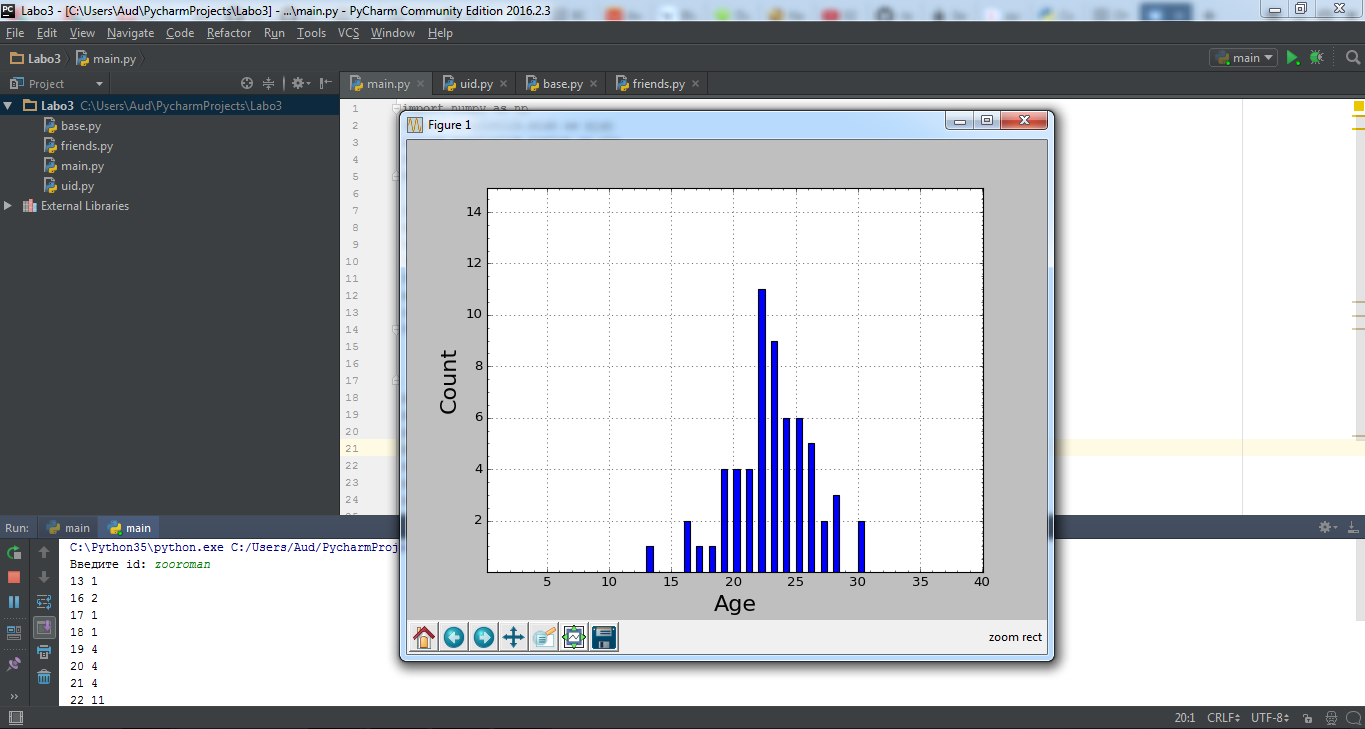
В случае успешного ответа необходим, чтобы преобразовать результат. В случае ошибочного ответа необходим, чтобы сформировать исключение

Для решения задачи нужно обратиться к двум метод 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

**Практика с дополнительными требованиями**

**Main.py**

**import** numpy **as** np  
**import** matplotlib.mlab **as** mlab  
**import** matplotlib.pyplot **as** plt  
**from** uid **import** \*  
**from** friends **import** \*  
  
input\_id = input(**'Введите id: '**)  
user = GetID(input\_id)  
user\_id = user.execute()  
friends\_client = GetFriends(user\_id)  
friends = friends\_client.execute()  
ages = []  
counts = []  
**for** (age, count) **in** friends:  
 print(**'{} {}'**.format(int(age), int(count)))  
 ages.append(int(age))  
 counts.append(int(count))  
plt.grid()  
plt.minorticks\_on()  
plt.axis([0, 90, 0, 50])  
plt.figure(num=1, figsize=(8, 6))  
plt.xlabel(**'Age'**, size=20)  
plt.ylabel(**'Count'**, size=20)  
plt.bar(ages, counts, width=0.5)  
plt.show()

**Base.py**

**class** BaseClient:  
  
 BASE\_URL = **None** method = **None** http\_method = **None  
 def** get\_params(self):  
 **pass  
 def** get\_json(self):  
 **pass  
 def** get\_headers(self):  
 **pass  
 def** generate\_url(self, method):  
 **return '{0}{1}'**.format(self.BASE\_URL, method)  
 **def** \_get\_data(self, method, http\_method):  
 response = **None** *#* ***todo выполнить запрос* return** self.response\_handler(response)  
 **def** response\_handler(self, response):  
 **return** response  
 **def** execute(self):  
 **return** self.\_get\_data(  
 self.method,  
 http\_method=self.http\_method  
)

**Friends.py**

**from** base **import** \*  
**import** requests  
**import** json  
**from** datetime **import** datetime  
**class** GetFriends(BaseClient):  
 BASE\_URL = **'https://api.vk.com/method/friends.get'** http\_method = **'GET'  
 def** \_\_init\_\_(self, uid):  
 self.uid = uid  
 **def** get\_params(self):  
 **return 'user\_id='** + str(self.uid) + **'&fields=bdate'  
 def** response\_handler(self, response):  
 **try**:  
 obje = json.loads(response.text)  
 friends = obje.get(**'response'**)  
 ages = []  
 **for** friend **in** friends:  
 b\_date = friend.get(**'bdate'**)  
 **if** b\_date **is None or** b\_date.count(**'.'**) < 2:  
 **continue** b\_date = datetime.strptime(b\_date, **"%d.%m.%Y"**)  
 n\_date = datetime.now()  
 ages.append(int((n\_date - b\_date).days) // 365.2425)  
 uniqages = list(set(ages))  
 **return** sorted([(x, ages.count(x)) **for** x **in** uniqages], key=**lambda** x: x[0])  
 **except**:  
 **raise** Exception(**"У пользователя нет друзей, либо они недоступны{}"**.format(self.uid))  
 **def** \_get\_data(self, method, http\_method):  
 response = requests.get(self.BASE\_URL + **'?'** + self.get\_params())  
 **return** self.response\_handler(response)

**Uid.py**

**from** base **import** \*  
**import** requests  
**import** json  
**class** GetID(BaseClient):  
 BASE\_URL = **'https://api.vk.com/method/users.get'** http\_method = **'GET'  
 def** \_\_init\_\_(self, name):  
 self.name = name  
 **def** get\_params(self):  
 **return 'user\_ids='** + self.name  
 **def** response\_handler(self, response):  
 **try**:  
 obje = json.loads(response.text)  
 **return** obje.get(**'response'**)[0].get(**'uid'**)  
 **except**:  
 **raise** Exception(**"Данный пользователь не найден {}"**.format(self.name))  
 **def** \_get\_data(self, method, http\_method):  
 response = **None** response = requests.get(self.BASE\_URL + **'?'** + self.get\_params())  
 **return** self.response\_handler(response)