Московский государственный технический университет им. Н.Э.Баумана

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

**Разработка интернет-приложений**

#### " Разработка управляющих файлов для режима

#### командной строки ОС "

|  |  |
| --- | --- |
| ИСПОЛНИТЕЛЬ: |  |
| студент группы **ИУ5-41** |
|  |
| **Прокошкин Р.Л.** |

Москва, МГТУ - 2016

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

**1. Цель лабораторной работы**

Знакомство с модулями и ООП в Python, а также работа с сетью. В лабораторной работе необходимо создать набор классов для реализации работы с VK API VK API.

**2. Задание лабораторной работы**

**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 #

**3. Файл проекта**

base.py

**import** requests  
**import** datetime  
**from** collections **import** Counter  
  
**class** BaseClient:  
 *# URL vk api* BASE\_URL = **'http://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):  
  
  
 response = requests.get(self.generate\_url(self.method), params=self.get\_params())  
 print (self.generate\_url(self.method))  
 **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  
 )  
  
  
**class** UserGet(BaseClient):  
 method = **'users.get'  
  
 def** \_\_init\_\_(self, username):  
 self.username = username  
  
 **def** get\_params(self):  
 **return** {  
 **'user\_ids'**: self.username  
 }  
 **def** response\_handler(self, response):  
 **return** response.json()[**'response'**][0][**'uid'**]  
  
  
  
  
  
**class** FriendsGet(BaseClient):  
 method = **'friends.get'  
  
 def** \_\_init\_\_(self, userid):  
 self.userid = userid  
  
 **def** get\_params(self):  
 **return** {  
 **'user\_id'**: self.userid,  
 **'fields'**: **'bdate'** }  
  
  
a = UserGet(**'prokoshkin\_roman'**).execute()  
*#b = a.json()  
#print (a.json()['response'][0]['uid'])*print (a)  
b = FriendsGet(a).execute()  
print (b.json())  
  
mas=[]  
t=b.json()[**'response'**]  
**for** i **in** t:  
 mas.append (i.get(**'bdate'**))  
  
print (mas)  
  
*#q = [1,2,3]  
#print (len(q))*filt=[]  
**for** i **in** mas:  
 **if** (i **is None**):  
 **continue  
 if** len(i.split(**'.'**)) == 3:  
 *#print (i.split('.'))  
 #print (i.split('.')[2])* filt.append(i.split(**'.'**)[2])  
  
print (filt)  
  
d = Counter(filt)  
keys= d.keys()  
keys=list(keys)  
keys.sort()  
**for** i **in** keys:  
 print(i, **':'**, **'#'** \* d[i])

**4. Результат работы**

