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

Факультет «Информатика и системы управления»

Кафедра «Системы обработки информации и управления»



Лабораторные работы по курсу:

**«Разработка Интернет Приложений»**

**Лабораторная работа №3**

Исполнитель:

Студентка группы ИУ5-54

Лысковцев К.А.

Преподаватель:

Гапанюк Ю.Е,

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Москва 2017г.

***Задание***

Вход: username или vk\_id пользователя

Выход: Гистограмма распределения возрастов друзей пользователя, поступившего на вход

***Листинг***

**base\_client.py:**

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) -> object:

return self.\_get\_data(

self.method,

http\_method=self.http\_method

)

**classes.py:**

import requests

import base\_client

import json

import datetime

class GetId(base\_client.BaseClient):

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

method = 'users.get'

http\_method = 'get'

# Отправка запроса к VK API

def \_get\_data(self, method, http\_method):

screen\_name = input()

response = requests.get(base\_client.BaseClient.generate\_url(self, GetId.method), params={'user\_ids': screen\_name})

return self.response\_handler(response)

# Обработка ответа от VK API

def response\_handler(self, response):

if response.status\_code == requests.codes.ok:

res = str(response.json()['response'][0]['uid'])

return res

else:

print("Error!")

class FriendsAnalytics(base\_client.BaseClient):

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

method = 'friends.get'

http\_method = 'post'

user\_id = None

def \_\_init\_\_(self, vk\_id):

self.user\_id = vk\_id

# Отправка запроса к VK API

def \_get\_data(self, method, http\_method):

data = {'user\_id': self.user\_id,'count': '5000', 'fields': 'bdate'}

response = requests.post(base\_client.BaseClient.generate\_url(self, self.method), data=data)

return self.response\_handler(response)

# Обработка ответа от VK API

def response\_handler(self, response):

if response.status\_code == requests.codes.ok:

return response

else:

print("Error!")

def Diagram(self,response):

if response.json()['response'] is not None: # Есть ли друзья

blist = list()

for person in response.json()['response']:

if person.get('bdate') is None: # Человек с не пустой датой

continue

else:

try:

blist.append(datetime.datetime.strptime(person['bdate'], '%d.%m.%Y'))

except:

continue

today = datetime.datetime.today()

agelist = list()

for b in blist:

age = today.year - b.year - 1

if today.month <= b.month:

if today.day <= b.day:

age += 1

agelist.append(age)

agelist.sort()

temp = agelist[0]

print(agelist[0], end='\t')

for a in agelist:

if a != temp:

print()

print(a, end='\t')

print('#', end='')

temp = a

else:

print('#', end='')

# Запуск клиента

def execute(self) -> object:

res = self.\_get\_data(self.method, http\_method=self.http\_method)

if res == "Error!":

print("Error!")

return res

# Печать диаграммы

self.Diagram(res)

**main.py:**

import requests

import datetime

import json

import base\_client

import classes

a = classes.GetId().execute()

print(a)

b = classes.FriendsAnalytics(a)

b.execute()

Вывод:

youra\_avdeev

154129192

14 ###

15 #########

16 #######

17 #######

18 ######################

19 ######################################################################################

20 ##################################################################################################

21 ###################################################################################

22 ########################################################################

23 ###########################################

24 #######################################

25 ############################

26 ######################################################

27 ########################################################

28 ######################################################

29 ################################################################

30 ####################################################

31 ###########################################################

32 ########################################

33 ######################################

34 ####################################

35 ###################################

36 ##################################

37 #################################

38 ##################################

39 ##################################

40 ###############################

41 ##################

42 #################

43 ##############

44 #################

45 ##################

46 ##################

47 #############

48 ############

49 #############

50 #######

51 #########

52 ###########

53 #########

54 #########

55 ########

56 #########

57 #########

58 #############

59 ######

60 #####

61 ###

62 #######

63 ###

64 #######

65 ###

66 #######

67 #

68 ####

69 ##

70 ####

71 ####

72 ##

73 #

74 #

76 #

77 #

83 ##

90 #

93 #

94 #

95 ##

96 ##

98 #

99 #####

100 #

104 #

105 #

106 #

107 ##

108 #

111 ####

112 #

113 #

114 ##

115 ##########################

116 #