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»

ЛР3. Python-классы

Исполнитель:

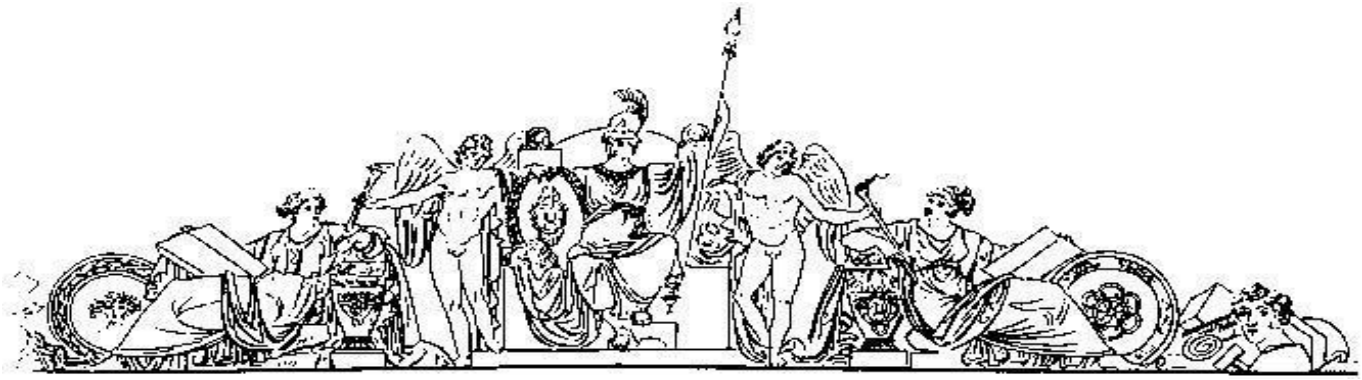
Студент группы РТ5-51

Чечнев А.А.

Преподаватель:

Гапанюк Ю. Е.

«___»_____



Москва 2017 г.

Цель работы:

В этой лабораторной работе необходимо познакомиться с модулями и ООП в Python, а также освоить работу с сетью. Кроме того, необходимо создать набор классов для реализации работы с
VK API.

Листинг:

progone.py

```
# coding=utf-8
from clientVK import *
from gist import Gist


def main():
    username = input('Введите логин вк: ')
    # username = "fs44fs4fs"
    # username = "allling"
    # username = "tolaysha"

    client_get_id = ClientGetID(username)
    user_id = client_get_id.execute()

    if client_get_id.is_success():
        print("ID: ", user_id)
    else:
        print('Нету такого айди, мэн')
        return

    # find age list
    friends_ages_list = ClientGetFriendsAges(user_id).execute()
    if not friends_ages_list:
        print('УУУУУ, нет друзей')
        return
    else:
        print("Ages: ", friends_ages_list)
        # write gist
        username_friend_gist = Gist(friends_ages_list)
        username_friend_gist.printGist()

    # show gist
    title = "Ages of Users "
    title_x = "Ages"
    title_y = "Users"
    username_friend_gist.showBar(title, title_x, title_y)

main()
```

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

    def __init__(self):
        # Инициализация
        self.success = True

    def get_params(self):
        return None

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

    def _get_data(self, method, http_method):
        # Отправка запроса к VK API
        # todo выполнить запрос
        try:
            response = requests.get(self.BASE_URL + self.method + "." + self.http_method,
params=self.get_params())
        except Exception:
            raise SystemExit("Нет ответа сервера VK: проверьте интернет соединение")

        return self.response_handler(response)

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

    def is_success(self):
        # Проверка найден ли ID
        return self.success

    def execute(self):
        # Запуск клиента
        try:
            self.success = True
            return self._get_data(
                self.method,
                http_method=self.http_method
            )
        except Exception as e:
            print('Косячок братан: ' + str(e))
            self.success = False
```

clientVK.py

```
# coding=utf-8
import base_client
from datetime import datetime


class MyError(Exception):
    pass


class ClientGetID(base_client.BaseClient):
    # метод vk api
    method = "users"
    # GET, POST, ...
    http_method = "get"

    def __init__(self, username):
        # Инициализация
        super().__init__()
        self.json_data = None
        self.username = username

    def get_params(self):
        # Получение логина
        return {
            "user_ids": self.username
        }

    def response_handler(self, response):
        # Получение ID пользователя
        self.json_data = response.json()
        print(self.json_data)
        try:
            return self.json_data["response"][1]["uid"]

        except IndexError as err:
            raise MyError
            # raise SystemExit("Сервер не дает ответа на выражение json_data['response'][0]['uid']")

    def get_json(self):
        # Получить json строку
        return self.json_data


def calculate_age(born, today):
    # Вычисление возраста
    return today.year - born.year - ((today.month, today.day) < (born.month, born.day))


class ClientGetFriendsAges(base_client.BaseClient):
    # метод vk api
    method = "friends"
    # GET, POST, ...
    http_method = "get"
```

```
    def __init__(self, user_id):
        # Инициализация
        super().__init__()
        self.json_data = None
        self.user_id = user_id

    def get_params(self):
        return {
            "user_id": self.user_id,
            "fields": "bdate"
        }

    def response_handler(self, response):
        # Получение списка возрастов
        self.json_data = response.json()
        ages = list()
        today = datetime.utcnow()

        for friend in self.json_data["response"]:
            date_of_birth = friend.get("bdate")
            try:
                date_of_birth = datetime.strptime(date_of_birth, "%d.%m.%Y")
            except Exception:
                continue

            ages.append(calculate_age(date_of_birth, today))

        return ages

    def get_json(self):
        return self.json_data()
```

gist.py

```
import matplotlib.pyplot as plt


class Gist:
    # данные гистаграммы

    def __init__(self, age_list):
        self._ages_list = sorted(age_list)
        self.age_dictionary = dict()
        for age in self._ages_list:
            self.age_dictionary.update(
                {age: self.age_dictionary.get(age, 0) + 1})

    def get_data(self):
        return self._ages_list

    def printGist(self):
```

```
    for age, count in self.age_dictionary.items():  # dict.items возвращает пары
        print(str(age).ljust(4) + ":" + "#" * count)

def showBar(self, title1, title_x, title_y):
    plt.bar(list(self.age_dictionary.keys()),
            self.age_dictionary.values(), color='g', width=0.9, linewidth=20)
    plt.show()
```