Муниципальное бюджетное общеобразовательное учреждение «Средняя общеобразовательная школа № 9 им. В. И. Некрасова» г. Сосновый Бор Ленинградской области

Высокие языки программирования на примере Python 3.1. Приложение для VK на Python

Индивидуальный проект
обучающейся 10а класса
Индейкиной Ирины

Руководитель проекта:
Вадим Викторович Нефедьев,
учитель (информатика)

2020 – 2021 учебный год

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Актуальность: В настоящее время, соцсети стали важной частью нашей жизни. Их используют политики как средство массовой информации, бизнесмены для рекламирования своей продукции, музыканты и художники - для распространения своего творчества. Страничка в соцсети может также служить источником заработка: так, сообщество в ВКонтакте с бизнес-тематикой, имеющее 100 тысяч подписчиков, в среднем приносит своему администратору 100 тыс. рублей в месяц. Всем, кто активно ведёт свою страницу в соцсетях, важно знать, как меняется ее популярность. Для этой цели удобно использовать приложение, отображающее различные признаки "успеха" с течением времени.

Проблема: Многим людям важно отслеживать популярность своих записей на страницах в социальных сетях, но при большом количестве записей делать это с помощью исключительно встроенных функций Вк неудобно.

Цель: Создать программу на языке Python, которая будет собирать данные со страницы пользователя и отображать графически количество выложенных постов в месяц, а также среднее число "лайков" и просмотров на запись.

Задачи:

1. Изучить язык программирования Python
2. Изучить технические особенности сети Вк
3. Написать программу для Вк на Python

Продукт: код программы, анализирующий активность на “стенке” пользователя Вк.

Объект: объектно-ориентированные языки программирования (Python)

Предмет: API и прочие технические параметры социальной сети Вк

Устойчивость: Мне кажется, что продукт этого проекта может быть интересен тем, кто активно ведет аккаунт в соцсетях и хочет видеть динамику его развития. Проект можно продолжить и дальше - усовершенствовать дизайн приложения, сделать аналоги для других социальных сетей, ускорить его работу. Для того, чтобы пользователь мог получить более глубокий анализ своей аудитории, можно было бы добавить функцию анализа наиболее активных

подписчиков: их пол, возраст, и т.д. Для выполнения этого проекта большие деньги не нужны, только компьютер и доступ в Интернет.

Распространение: Этот проект может реализовать любой человек, умеющий программировать и имеющий компьютер. Он может быть интересен человеку из любой страны.

Я создала план мероприятий по целям проекта (приложение 1.)

ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

История развития социальных сетей

Понятие “социальная сеть” появилось еще до изобретения интернета. Его ввел в 1954 году социолог Джеймс Барнс и определил как “социальную структуру, состоящую из группы узлов, которыми являются социальные объекты (люди или организации), и связей между ними (социальных взаимоотношений)”. Проще говоря, это система связей между людьми и группами людей. В социологии этот термин используется до сих пор. В информатике это же словосочетание обозначает “онлайн-платформу, которая используется для общения, знакомств, создания социальных отношений между людьми, которые имеют схожие интересы или офлайн-связи, а также для развлечения (музыка, фильмы) и работы”.

Идея создания такой универсальной сети, объединяющей людей, с помощью компьютерных технологий, принадлежит американскому ученому Джозефу Карл Робнетт Ликлайдеру. Он же 29 октября 1969 сумел переслать на компьютер в другом университете сообщение, положив начало развитию интернета - и с ним социальных сетей.

Первый аналог современных социальных сетей был создан в 1971 году для внутренней сети ARPANET, разработанной американскими военными. Он использовался в основном для передачи файлов и информации с одного компьютера на другой. Следующий этапом развития стали “электронные доски объявлений” (*bulletin board system- BBS*), первую из которых создал в 1978 году Уард Кристенсен. Это были локальные системы общения компьютеров через коммутируемые телефонные сети. Пользователь звонил с помощью программы-терминала на станцию оператора, проходил авторизацию и получал возможность скачать пакет пришедших с прошлого посещения сообщений и загрузить новые, а также прочитать добавленные на сайт объявления и новости. В дальнейшем функциональность BBS была расширена: появилась возможность играть в игры, проводить телеконференции, общаться в чате. Они быстро стали очень популярны. Уже в 1988 году в мире насчитывалось 5000 электронных досок, и вышла эта система из употребления только из-за ее устаревшего метода работы.

В 1988 году, финский студент Ярко Ойкариненом изобрел технологию «IRC» (англ. Internet Relay Chat — ретранслируемый интернет-чат), который позволял переписываться в реальном времени. Но первой полноценной социальной сетью в современном понимании считается Classmates.com, которую запустил в 1995 году Рэнди Конрадс. Она была придумана, чтобы люди могли находить своих одноклассников и однокурсников после выпуска. Эта сеть популярна даже сейчас, а ее российский аналог, Одноклассники, вторая по популярности социальная сеть в России.

С распространением интернета, разнообразных социальных сетей стало очень много; перечислить их всех практически невозможно. В 1999 был запущен сайт LiveJournal - популярная и сейчас блог-платформа, где люди могут делиться своими текстами, стихами, статьями. В 2002 году была основана сеть LinkedIn, позволяющая развивать профессиональные контакты, а в 2004 году четыре студента из Гарварда запустили вебсайт Facebook, который совершил революцию в сфере социальных сетей. Сначала он был доступен только для студентов этого университета, но сейчас Facebook самая популярная социальная сеть в мире. В 2006 году Павлом Дуровым был запущен ее русский аналог - ВКонтакте.

Сейчас социальные сети делят на контактные, тематические, видео- и фото-хостинги, форумы и блог платформы. Контактные соцсети - самые популярные: в них зарегистрированы больше половины русскоязычных пользователей соцсетей. Вконтакте- пример такой сети, а также Facebook, Одноклассники, Мой Мир. У тематических соцсетей узкая направленность: например, MySpace объединяет любителей музыки, а LinkedIn служит для профессиональных знакомств. Платформы фото- и видео-хостинга - это сайты, где люди делятся отснятыми материалами. Примеры - Youtube, Flickr.com. Форумы, например, англоязычный Reddit, созданы специально для общения и обсуждения. Наконец, блог-платформы - LiveJournal, LiveInternet - специализированы под ведение блогов.

Большинство соцсетей имеет какую-нибудь систему оценки материала, выложенного пользователями - например, пальцы “вверх” или “вниз” на Youtube,

“сердечки” (“лайки”, “понравилось”) -в ВКонтакте, Инстаграме, а также количество просмотров.

API

API (application programming interface) - описание способов, которыми одна компьютерная программа может взаимодействовать с другой программой (в данном случае - сайтом). Например, многие сайты позволяют проходить регистрацию через существующие аккаунты в социальных сетях или “делиться с друзьями” страницами. Это реализуется через API. Также через API можно взаимодействовать с языками программирования; для этого созданы специальные библиотеки. В своем проекте я использовала библиотеку vk_api.

ПРАКТИЧЕСКАЯ ЧАСТЬ

Ход работы программы

У Питона, помимо встроенных функций, есть много таких, которые нужно устанавливать и загружать в программу функцией *import*. Это делается в начале файла (рис. 1):

```
from vk_api import VkApi
import tkinter as tk
import time
from datetime import datetime

import matplotlib.dates as mdates
from matplotlib.figure import Figure
import matplotlib.pyplot as plt
from matplotlib.ticker import MaxNLocator
from matplotlib.backends.backend_tkagg import FigureCanvasTkAgg

from selenium.webdriver.support import expected_conditions as EC
from selenium import webdriver
from selenium.webdriver.support.ui import WebDriverWait
from webdriver_manager.chrome import ChromeDriverManager

```

Рис. 1

Tkinter- это библиотека для создания графических интерфейсов пользователя(GUI). С помощью ее создадим окно для входа (рис.2):

```
window=tk.Tk()
frame1=tk.Frame(master=window)
log=tk.Label(master=frame1,text='Добро пожаловать!'); log.grid(row=0)
beg=tk.Button(master=frame1, text='Войти',command=login); beg.grid(row=1)
lab=tk.Label(master=frame1,text='User ID (необязательно):'); lab.grid(row=2)
own=tk.Entry(master=frame1);own.grid(row=3,pady=3)

frame2=tk.Frame(master=window)
strt=tk.Label(master=frame2,text='Введите начальную дату(мм/ГГГГ):'); strt.grid(row=0)
strt=tk.Entry(master=frame2);strt.grid(row=1,sticky='NESW')
end=tk.Label(master=frame2,text='Введите конечную дату(мм/ГГГГ):' ); end.grid(row=2)
end=tk.Entry(master=frame2); end.grid(row=3,sticky='NESW')
an=tk.Button(master=frame2, text='Анализировать',command=start); an.grid(row=4)
comm=tk.Label(master=frame2,text='Оставьте пустыми для анализа всей стены' );comm.grid(row=5)

frame3=tk.Frame(master=window)
frame1.pack()

window.mainloop()
```

Рис.2

Сейчас показывается только содержимое frame1(рис. 3):

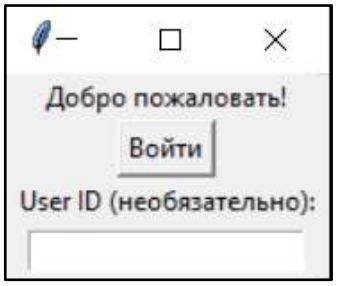

Рис.3

API - мощный инструмент. С его помощью любой человек из любой точки мира может получить личную информацию, если она не защищена. Чтобы не происходили утечки, сайты должны контролировать доступ к API - как пример, изначальная цель моего проекта была в том, чтобы выделить мессенджер Вк в отдельную программу, но недавно ВКонтакте закрыло доступ к сообщениям для сторонних разработчиков- поэтому для получения любой информации о пользователе требуется “токен”, или ключ. Для доступа к разным данным, более или менее конфиденциальным, существуют разные ключи. Например, у каждого приложения Вк есть сервисный ключ, передав который можно получить только некоторые общедоступные данные. Более личные данные можно получить с ключом доступа пользователя. Получить нужную нам информацию можно как через ключ доступа пользователя, так и через сервисный ключ приложения, но есть шанс, что с сервисным ключом вернется неполная информация, если некоторые записи видны только друзьям, или если аккаунт приватный - тогда доступ вообще будет закрыт.

Если пользователь нажмет сразу на кнопку “Войти”, откроется окно браузера, где он сможет пройти аутентификацию Вк и получить ключ доступа пользователя. Если же сначала ввести *id* номер страницы, запрос будет отправлен с сервисным ключом. Это описано здесь (рис. 4):

```
def login():
    global owner_id,vk_session,vk
    owner_id=own.get()
    if len(owner_id)==0:
        driver = webdriver.Chrome(ChromeDriverManager().install())
        url = "https://oauth.vk.com/authorize?client_id=7693051&scope=wall&redire
        driver.get(url)
        wait = WebDriverWait(driver,240)
        wait.until(EC.url_contains('expires_in=86400&user_id='))
        t_url=driver.current_url
        token=t_url[t_url.find('access_token=')+13:t_url.find('&')]
```

```
        owner_id=t_url[t_url.rfind('=')+1:]
        driver.quit()

        vk_session = VkApi(token=token)
        vk = vk_session.get_api()
    else:
        vk_session = VkApi(token='b86fcf73b86fcf73b86fcf73b0b81aad88bb86fb86fcf73
        vk = vk_session.get_api()

    frame1.forget()
    frame2.pack()
```

Рис.4

С помощью библиотеки *selenium* посылается запрос по ссылке “url”. Когда пользователь зайдет в систему, браузер будет перенаправлен на другую ссылку, в которой будет содержаться ключ доступа. Общий вид ссылки: *https://oauth.vk.com/blank.html#access_token=****здесь_строка_с_токеном****&expires_in=86400&user_id=****id*. *“Expires_in=86400”* означает, что срок действия токена- одни сутки. Функцией *current.url* достается ссылка и дальше обрабатывается как строка, чтобы добыть токен и id пользователя.

Если авторизация прошла успешно, появляется следующее окно (рис. 6), в противном случае - текст ошибки (рис. 5):

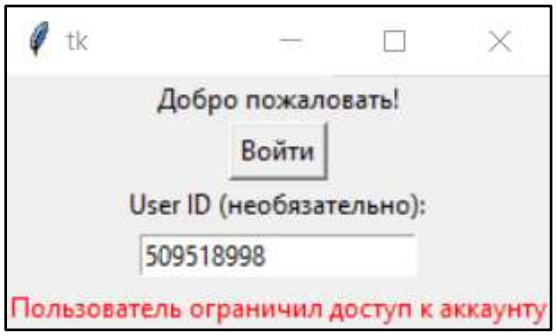

Рис. 5

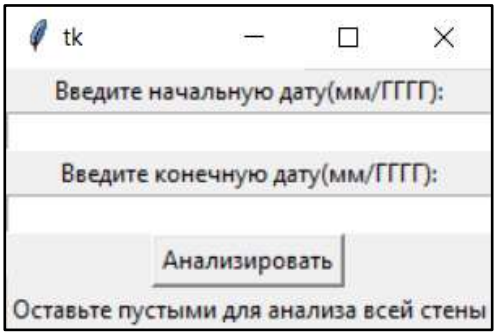

Рис.6

```
def start():
    global strt,end
    wall=vk.wall.get(owner_id=owner_id,count=100,filter='owner',extended=1)
    name=(wall['profiles'][0].get('first_name')+' ' +wall['profiles'][0].get('last_name'))
    count=wall['count']
    sdate_l=strt.get(); edate_l=end.get()

    if len(sdate_l)==0 or len(edate_l)==0:
        edate=wall['items'][0]['date']
        sdate=vk.wall.get(owner_id=owner_id,offset=(count-1),filter='owner')['items'][0]['date']
        last_date=sdate
```

После нажатия кнопки “Анализировать”, выполняется следующий код (рис. 7):

```
    else:
        sdate=time.mktime(time.strptime(sdate_l, "%m/%Y"))
        edate=time.mktime(time.strptime(edate_l, "%m/%Y"))
        last_date=vk.wall.get(owner_id=owner_id,offset=(count-1),filter='owner')['items'][0]['date']
    frame2.forget()
```

Рис. 7

В строке *wall=vk.wall.get(owner_id=owner_id, count=100, filter=owner)*, программа посылает запрос к серверу Вк и возвращает данные о первых ста записей, выложенных указанным пользователем. Для каждой записи возвращается дата создания, количество "лайков", просмотров, описание самой записи, и прочая информация в виде словаря.

Введенные пользователем даты надо перевести в формат Unix - количество секунд, прошедшее с 1 января 1970 - для того, чтобы их можно было сравнивать с датами Вк. Если окна для ввода оставлены пустыми, то метками начала и конца периода становятся даты первой и последней выложенной записи.

Следующий этап (рис. 8):

```
off=0
posts={}
br=''
while br=='':
    for i in wall['items']:
        if i['date']>edate:
            pass
        elif i['date']<sdate:
            br='y'
            break
        else:
            curr_date=datetime.fromtimestamp(i['date']).strftime('%b/%Y')
            if curr_date not in posts:
                posts[curr_date]=[0,0,0]
            likes=i['likes']['count']
            try:
                views=i['views']['count']
            except KeyError:
                views=0
            posts[curr_date]=[(posts[curr_date][0])+1, posts[curr_date][1]+likes,

    if br=='' and i['date']!=last_date:
        off+=len(wall['items'])
        wall=vk.wall.get(owner_id=owner_id,offset=off,count=100,filter='owner')
    else:
        br='y'
```

Рис. 8

Здесь перебором просматривается весь список данных, выискивая те записи, у которых даты входят в указанный промежуток; информация о количестве записей, просмотров и лайков в месяц сохраняется в словарь. Поскольку за раз получить больше информацию больше чем о сотне записей нельзя, если записей много, приходится посылать запрос несколько раз, меняя параметр *offset*- “сдвиг” от последней даты. Здесь следует отметить, что если в диапазон попадают записи, выложенные раньше марта 2017 года, количество просмотров у них всегда равно 0, потому что счетчик просмотров был добавлен на сайт только тогда. Различные параметры затем сохраняются в отдельные переменные - они будут отображаться на оси y.

```
  #Определяются переменные для графиков
mon=[i[:i.find('/')] for i in posts.keys()][::-1]
yrs=["'"+i[-2:] for i in posts.keys()][::-1]
pos_yr=sorted(set([yrs.index(i)+1 for i in yrs]))
yrs=sorted(set(yrs))
pos=[i+1 for i in range(len(mon))]

nums=[val[0] for val in posts.values()][::-1]
likes=[val[1]//val[0] for val in posts.values()][::-1]
views=[val[2]//val[0] for val in posts.values()][::-1]

#Создается график
graphs=Figure(figsize=(15,5))
p_gr = graphs.add_subplot(131); rects = p_gr.bar(pos, nums,color='lightblue')
p_gr.set_title('Записи в месяц'); p_gr.grid(linewidth=0.5)

v_gr = graphs.add_subplot(132); rects = v_gr.bar(pos,views)
v_gr.set_title('Ср. кол-во просмотров на запись');  v_gr.grid(linewidth=0.5)

lk_gr = graphs.add_subplot(133); rects = lk_gr.bar(pos, likes,color='darkblue')
lk_gr.set_title('Ср. кол-во "понравилось" на запись');  lk_gr.grid(linewidth=0.5)

for i in p_gr, v_gr,lk_gr:
    i.yaxis.set_major_locator(MaxNLocator(integer=True))
    i.xaxis.set_major_locator(MaxNLocator(integer=True))
    i.set_xticks(pos_yr); i.set_xticklabels(yrs)
    i.set_xticks(pos, minor=True); i.set_xticklabels(mon,minor=True)
    i.tick_params(axis='x', which='minor', length=0,rotation=45); i.tick_params(axis='x', whic
    i.xaxis.remove_overlapping_locs = False

graphs.tight_layout()
canvas = FigureCanvasTkAgg(graphs, master=frame3)
canvas.get_tk_widget().grid(row=1,columnspan=8)
```

Рис.9

На рисунке 9 - следующий шаг. Для оси x создается список целых чисел длинной в список дат (pos). На этих координатах будут подписи месяцев. Затем в отдельную переменную (pos_yr) сохраняются позиции смены года, где будут подписи годов. Графики рисуются и форматируются, добавляются в окно приложения.

И наконец, создаются снова ячейки для ввода дат и смены пользователя - чтобы можно было перерисовать график с другими параметрами, не выходя из программы (рис 10.):

```
greet=tk.Label(master=frame3,text='Пользователь: {}'.format(name)); greet.grid(row=0)
strt1=tk.Label(master=frame3,text='Начало периода (мм/ГГГГ):' ); strt1.grid(row=2,column=0,sticky='E')
strt=tk.Entry(master=frame3);strt.grid(row=2,column=1,sticky='W',padx=2)
end1=tk.Label(master=frame3,text='Конец периода (мм/ГГГГ):' ); end1.grid(row=2, column=2,sticky='E',padx
end=tk.Entry(master=frame3); end.grid(row=2,column=3,sticky='W')
an1=tk.Button(master=frame3, text='Перерисовать',command=start); an1.grid(row=2,column=4,padx=12)
strt1=tk.Label(master=frame3,text='Сменить ID:' ); strt1.grid(row=2,column=5,sticky='E')
us=tk.Entry(master=frame3);us.grid(row=2,column=6,sticky='W',padx=2)

frame3.pack()
```

Рис. 10

Результат на рисунке 11:

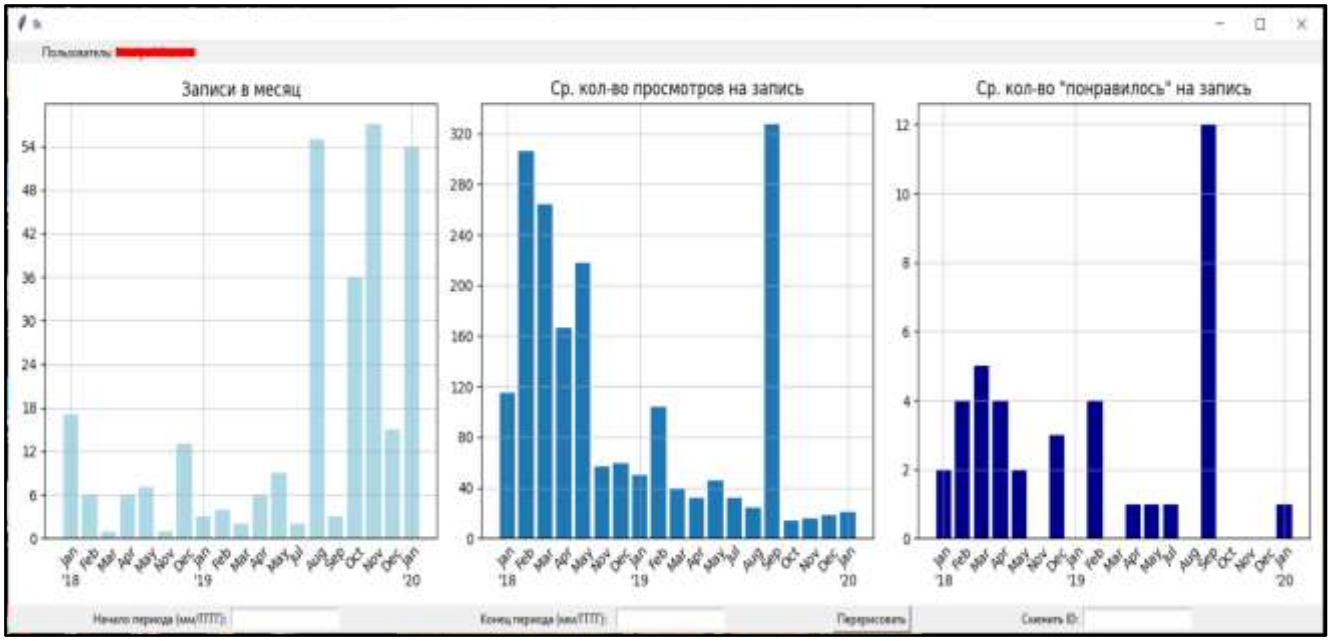

Рис. 11

У этого приложения на данный момент есть три существенных недостатка: во-первых, оно работает довольно медленно; во-вторых, если в диапазоне много дат, то подписи на оси x перекрывают друг друга; наконец, одна из используемых мною функций, *graphs.tight_layout()*, размещающая графики относительно друг друга, по заявлению библиотеки, скоро перестанет работать.

Полный код программы представлен в приложении 2.

Экспорт

Чтобы программу можно было отправить другому человеку, ее надо экспортировать. Если просто отправить файл с текстом программы, то нет гарантии, что у получателя будет та же версия Питона и те же библиотеки, которые были использованы в этом проекте. Поэтому есть шанс, что программа не будет работать. Чтобы этого избежать, нужно экспортировать программу в формате .exe. Для этого можно использовать модуль *pyinstaller*, его можно запустить с командной строки компьютера. Он позволяет упаковать в один .exe-файл все модули, нужные для работы программы.

ЗАКЛЮЧЕНИЕ

Цель этого проекта была в том, чтобы сделать приложение, позволяющее анализировать популярность и активность на стене пользователя. Я считаю, что, в целом, цель была достигнута. Есть, однако, недостатки, которые следовало бы устранить.

СПИСОК ЛИТЕРАТУРЫ

Электронные ресурсы:

BBS. — Текст : электронный // Википедия, свободная энциклопедия : [сайт]. — URL: https://ru.wikipedia.org/wiki/BBS (дата обращения: 16.03.2021).

Социальная сеть. — Текст : электронный // Википедия, свободная энциклопедия : [сайт]. — URL: https://ru.wikipedia.org/wiki/Социальная_сеть (дата обращения: 15.03.2021).

Как скомпилировать Python код в .exe. — Текст : электронный // Make-info.com : [сайт]. — URL: https://www.make-info.com/kak-skompilirovat-python-kod-v-exe/ (дата обращения: 15.03.2021).

Все о социальных сетях. Влияние на человека. Часть 1. — Текст : электронный // Seclgroup : [сайт]. — URL: https://seclgroup.ru/article-vse-o-socialnyh-setjah-vlijanije-na-cheloveka.html (дата обращения: 15.03.2021).

Giovanni, N. How the Internet was born: A stuttered hello / N. Giovanni. — Текст : электронный // The Conversation : [сайт]. — URL: https://theconversation.com/how-the-internet-was-born-a-stuttered-hello-67903 (дата обращения: 15.03.2021).

Типы социальных сетей. — Текст : электронный // Медиасфера : [сайт]. — URL: https://www.media-sfera.com/internet-promo/smm/types/ (дата обращения: 15.03.2021).

Зиновьева, Алёна История социальных сетей: появление и развитие / Алёна Зиновьева. — Текст : электронный // Sciencepop : [сайт]. — URL: https://sciencepop.ru/istoriya-sotsialnyh-setej-poyavlenie-i-razvitie/ (дата обращения: 15.03.2021).

ПРИЛОЖЕНИЯ

Приложение 1.

№	Мероприятие	Сроки
1	Изучить язык программирования Python (курс Питонтьютор, другие ресурсы)	конец ноября
2	Изучить теоретическую информацию по теме проекта	декабрь - январь
3	Написание программы, решающей проблему проекта	январь-март

Приложение 2.

```
from vk_api import VkApi
import vk_api.exceptions
import tkinter as tk
import time
from datetime import datetime
from matplotlib.figure import Figure
import matplotlib.pyplot as plt
from matplotlib.ticker import MaxNLocator
from matplotlib.backends.backend_tkagg import FigureCanvasTkAgg,NavigationToolbar2Tk
from selenium.webdriver.support import expected_conditions as EC
from selenium import webdriver
from selenium.webdriver.support.ui import WebDriverWait
from webdriver_manager.chrome import ChromeDriverManager

def auth():
    global vk_session,vk,owner_id
    #Вход пользователя, получение кода доступа пользователя
    driver = webdriver.Chrome(ChromeDriverManager().install())
    url =
"https://oauth.vk.com/authorize?client_id=7693051&scope=wall&redirect_uri=https%3A%2F
%2Foauth.vk.com%2Fblank.html&response_type=token&display=page"
    driver.get(url)
    wait = WebDriverWait(driver,240)
    wait.until(EC.url_contains('expires_in=86400&user_id='))
    t_url=driver.current_url
    token=t_url[t_url.find('access_token=')+13:t_url.find('&')]
    owner_id=t_url[t_url.rfind('=')+1:]
    driver.quit()

    vk_session = VkApi(token=token)
    vk = vk_session.get_api()
```

```
def login():
    global owner_id,vk_session,vk
    owner_id=own.get()
    if len(owner_id)==0:
        auth()
        frame1.forget()
        frame2.pack()
    else:
        try:
        #Работа с сервисным ключом приложения
            vk_session =
VkApi(token='b86fcf73b86fcf73b86fcf73b0b81aad88bb86fb86fcf73e7bf5d1f6809ce9cb3dd02
88')
            vk = vk_session.get_api()
            wall=vk.wall.get(owner_id=owner_id,count=1)
            frame1.forget()
            frame2.pack()
        except vk_api.exceptions.ApiError:
            err=tk.Label(master=frame1,text='Пользователь ограничил доступ к
аккаунту',fg='red')
            err.grid(row=4)

def start():
    #Определение временных рамок
    global strt,end,own,owner_id,canv,tool,frame3
    wall=vk.wall.get(owner_id=owner_id,count=100,filter='owner',extended=1)
    name=vk.users.get(user_ids=owner_id)
    name=name[0].get('first_name')+' '+name[0].get('last_name')
    count=wall['count']
    print(count)
    sdate_l=strt.get(); edate_l=end.get()
```

```
    if len(sdate_l)==0 or len(edate_l)==0:
        edate=wall['items'][0]['date']
        sdate=vk.wall.get(owner_id=owner_id,offset=(count-1),filter='owner')['items'][0]['date']
        last_date=sdate
    else:
        sdate=time.mktime(time.strptime(sdate_l, "%m/%Y"));
edate=time.mktime(time.strptime(edate_l, "%m/%Y"))
        last_date=vk.wall.get(owner_id=owner_id,offset=(count-
1),filter='owner')['items'][0]['date']
    frame2.forget()

    #Обработка полученных данных
    off=0
    posts={}
    br=''
    while br=='':
        for i in wall['items']:
            if i['date']>edate:
                pass
            elif i['date']<sdate:
                br='y'
                break
            else:
                curr_date=datetime.fromtimestamp(i['date']).strftime('%b/%Y')
                if curr_date not in posts:
                    posts[curr_date]=[0,0,0]
                likes=i['likes']['count']
                try:
                    views=i['views']['count']
                except KeyError:
                    views=0
```

```
            posts[curr_date]=[(posts[curr_date][0])+1, posts[curr_date][1]+likes,
(posts[curr_date][2])+views]

        if br==" and i['date']!=last_date:
            off+=100
            wall=vk.wall.get(owner_id=owner_id,offset=off,count=100,filter='owner')
        else:
            br='y'

      #Определяются переменные для графиков
    mon=[i[:i.find('/')] for i in posts.keys()][::-1]
    yrs=[""+i[-2:] for i in posts.keys()][::-1]
    pos_yr=sorted(set([yrs.index(i)+1 for i in yrs]))
    yrs=sorted(set(yrs))
    pos=[i+1 for i in range(len(mon))]

    nums=[val[0] for val in posts.values()][::-1]
    likes=[val[1]//val[0] for val in posts.values()][::-1]
    views=[val[2]//val[0] for val in posts.values()][::-1]

    #Создается график
    graphs=Figure(figsize=(15,5))
    p_gr = graphs.add_subplot(131); rects = p_gr.bar(pos, nums,color='lightblue')
    p_gr.set_title('Записи в месяц')

    v_gr = graphs.add_subplot(132); rects = v_gr.bar(pos,views)
    v_gr.set_title('Ср. кол-во просмотров на запись')

    lk_gr = graphs.add_subplot(133); rects = lk_gr.bar(pos, likes,color='darkblue')
    lk_gr.set_title('Ср. кол-во "понравилось" на запись')

    for i in p_gr, v_gr,lk_gr:
```

```
        i.grid(linewidth=0.5)
        i.yaxis.set_major_locator(MaxNLocator(integer=True))
        i.xaxis.set_major_locator(MaxNLocator(integer=True))
        i.set_xticks(pos_yr); i.set_xticklabels(yrs)
        i.set_xticks(pos, minor=True); i.set_xticklabels(mon,minor=True)
        i.tick_params(axis='x', which='minor', length=0,rotation=45); i.tick_params(axis='x',
which='major', length=20,width=0)
        i.xaxis.remove_overlapping_locs = False
    graphs.tight_layout()

    tool=tk.Frame(master=window)
    canv=tk.Frame(master=window)
    canvas = FigureCanvasTkAgg(graphs, master=canv)
    canvas.draw()
    canvas.get_tk_widget().update()
    canvas.get_tk_widget().grid()

    toolbar = NavigationToolbar2Tk(canvas,tool)
    toolbar.update(); toolbar.pack()

    #Создаются иконки frame3
    strt1=tk.Label(master=frame3,text='Начало периода (мм/ГГГГ):' );
strt1.grid(row=1,column=0,sticky='E')
    strt=tk.Entry(master=frame3);strt.grid(row=1,column=1,sticky='W',padx=2)
    end1=tk.Label(master=frame3,text='Конец периода (мм/ГГГГ):' );
end1.grid(row=1,column=2,sticky='E',padx=8)
    end=tk.Entry(master=frame3); end.grid(row=1,column=3,sticky='W')
    an1=tk.Button(master=frame3, text='Перерисовать',command=start);
an1.grid(row=1,column=4,padx=12)
    greet=tk.Label(master=frame3,text='Пользователь: {}'.format(name),font=("DejaVu Sans",
13)); greet.grid(row=0,column=0)
    canv.grid(row=0,sticky='NESW')
```

```
    tool.grid(row=1)
    frame3.grid(row=2)

#Основное окно и frame2
window=tk.Tk()
frame1=tk.Frame(master=window)
log=tk.Label(master=frame1,text='Добро пожаловать!',font=("Helvetica", 14));
log.grid(row=0)
beg=tk.Button(master=frame1, text='Войти',command=login); beg.grid(row=1)
lab=tk.Label(master=frame1,text='User ID (необязательно):',font=("Helvetica", 10));
lab.grid(row=2)
own=tk.Entry(master=frame1);own.grid(row=3,pady=3)

frame2=tk.Frame(master=window)
strt=tk.Label(master=frame2,text='Введите начальную дату(мм/ГГГГ):'); strt.grid(row=0)
strt=tk.Entry(master=frame2);strt.grid(row=1,sticky='NESW')
end=tk.Label(master=frame2,text='Введите конечную дату(мм/ГГГГ):' ); end.grid(row=2)
end=tk.Entry(master=frame2); end.grid(row=3,sticky='NESW')
an=tk.Button(master=frame2, text='Анализировать',command=start); an.grid(row=4)
comm=tk.Label(master=frame2,text='Оставьте пустыми для анализа всей стены'
);comm.grid(row=5)

frame3=tk.Frame(master=window)
frame1.pack()
window.mainloop()
```