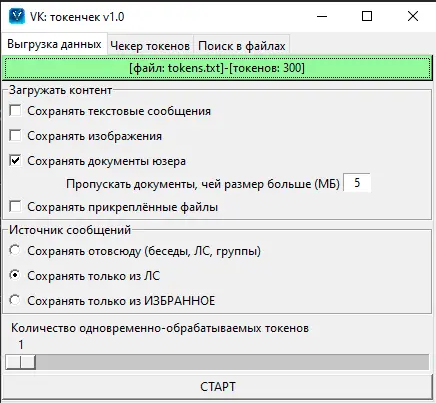
**Интерфейс:**​

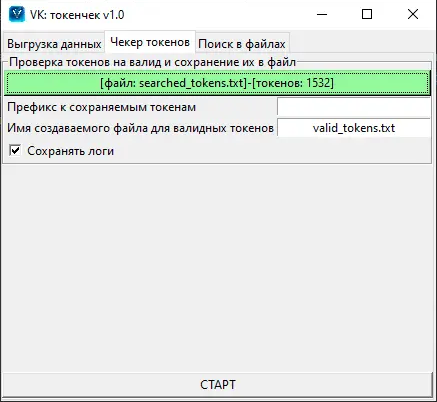
Главное окно содержит 3 вкладки: "выгрузка данных", "чекер токенов" и "поиск в файлах". В первой вкладке содержатся элементы управлением выгрузкой на ПК данных из профиля.



​

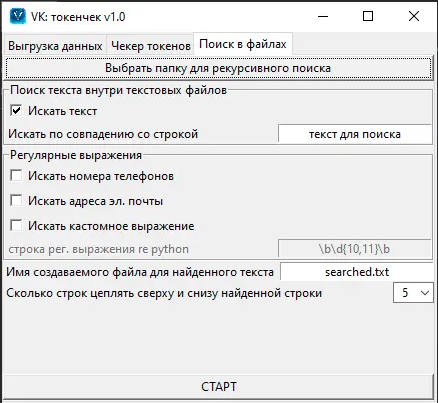
Для начала выбираем txt-файл с токенами (токены должны быть записаны: на каждой строке отдельный токен, начинающийся с vk1.a).  
Далее выбираем, что нам нужно выгрузить из профиля: текстовые сообщения, изображения (из диалогов), документы юзера (вкладка "файлы" в вк -> ), прикреплённые к сообщению файлы.  
Далее выбираем, из каких диалогов нужно выгружать сообщения. По дефолту выбран радиобаттон "только из ЛС". Это значит, что будут загружены только сообщения из диалогов с людьми. Если поставить "сохранять только из ИЗБРАННОЕ", то будут выгружены все сообщения из диалога с самим собой.  
Далее мы задаём, сколько потоков запустить для обработки токенов. По дефолту стоит 1. Максимум 100.  
После того как нажмём старт, запустится обработка токенов. Графическое окно программы пропадёт (скроется), а вся инфа о текущем прогрессе будет выводиться в командную строку.  
Все файлы сохраняются в папку users\_dialogs, которая создастся рядом со скриптом. В этой папке для кажддого юзера создается папка, чьё название = имя\_фамилия юзера, чей токен обрабатывается. В этой папке будут созданы txt-файлы с диалогами, чьи названия будут = имя\_фамилия того, с кем диалог. Сообщения в txt-файлах записаны с датой отправки и id отправителя.  
После загрузки всех нужных данных, графическое окно снова отобразится.

**Далее, вкладка "чекер токенов"**



​

тут мы можем выбрать файл, в котором содержатся токены. Не важно, как они записаны. Скрипт сам находит их среди текста в файле (ищет подстроку "vk1.a", и от точки вхождения берёт текст до +220 символов (длина токена)). Если нужно, указываем префикс (этот текст будет добавлен к каждому токену при их сохранении в файл). В лог сохраняется: токен и данные об аккаунте.  
Задаём имя выходного файла, который будет создан, и в который будут сохранены валидные токены.  
Нажимаем старт, и ждём. Окно скрывается, а в командной строке виден процесс обработки. После обработки токенов окно вновь будет отображено.  
​

**вкладка "поиск в файлах"**​

Тут мы может произвести поиск нужного текста во всех сообщениях, которые мы выгрузили. Выбираем папку, в которой будет произведён рекурсивный поиск (если не выбрать папку, то будет обработан тот каталог, в котором находится скрипт). Вводим слово, которое ищем (скрипт пройдёт по всем файлам диалогов, и будет сохранять в файл все сообщения, в которых обнаружена подстрока, которую вы ввели). Если нам не нужно искать точное совпадение с подстрокой, а нужно искать регулярными выражениями, то снимаем галку "искать текст", и ставим галку в нужном пункте. Если нам нужно найти все адреса эл. почт, то ставим галку "искать адреса эл. почты", если номера то ставим галку "искать номера телефонов". Если нужно задать своё регулярное выражение, то активируем третью галку, и вводим нашу регулярку (тут используется модуль re пайтона. Как составлять регулярки для него - гуглите). Задаём имя файла, в который будут сохранены все найденные сообщения. Далее выставляем, сколько сообщений цеплять сверху и сниху сообщения, в котором была найдена нужная нам инфа. Если ставим 1, то значит, что для каждого сообщения, в котором будет найдена искомая подстрока, будет сохранено 1 предшествующее сообщение, и одно последующее сообщение. Удобно, когда юзер в одном сообщение отправил кому то имейл, а следующим сообщением пароль. Таким образом мы выцепим и мыло и пароль :)  
Дальше всё как обычно. Нажимаем старт, окно скрывается, и в командную строку выводится инфо. После окончания окно снова отображается.  
  
Пока на этом всё. Скрипт пилил для себя. Но, думаю, кому то кроме меня он может пригодится. Буду расширять его функционал, и новые версии выкатывать сюда. В планах прокси, управление поведением аккаунтами, рассылка сообщений и т.д.

Используемые модули:​

* tkinter
* vk\_api
* os
* datetime
* requests
* multiprocessing
* concurrent.futures
* collections
* sys
* re