Федеральное агентство связи

Ордена Трудового Красного Знамени федеральное государственное бюджетное образовательное учреждение высшего образования

«Московский технический университет связи и информатики»

Кафедра Математической кибернетики и информационных технологий

Лабораторная работа №3

«Разработка скриптов для автоматизации тестирования функций поисковых систем»

по дисциплине

«Методы отладки и тестирования программных продуктов»

Выполнил:

студент группы БВТ1601

Челядин И.А.

Москва 2019

# Цель работы

Получение практических навыков при анализе возможностей поисковых web-систем.

# Задание

Открыть поисковый сайт. В поисковом поле последовательно набирать поисковые словосочетания Тестирование, Эффективное тестирование, Эффективное тестирование программ и нажимать кнопку «Поиск». После реализации поиска определить количество найденных ссылок во всех трех случаях. Проверить, какие из ссылок и в каком количестве появившиеся на первой странице после третьего поиска, содержатся на первых трех страницах сайтов открытых после реализации первого и второго поиска.

Выполнение

Скрипт осуществляет открытие поискового сайта (Рисунок 1).

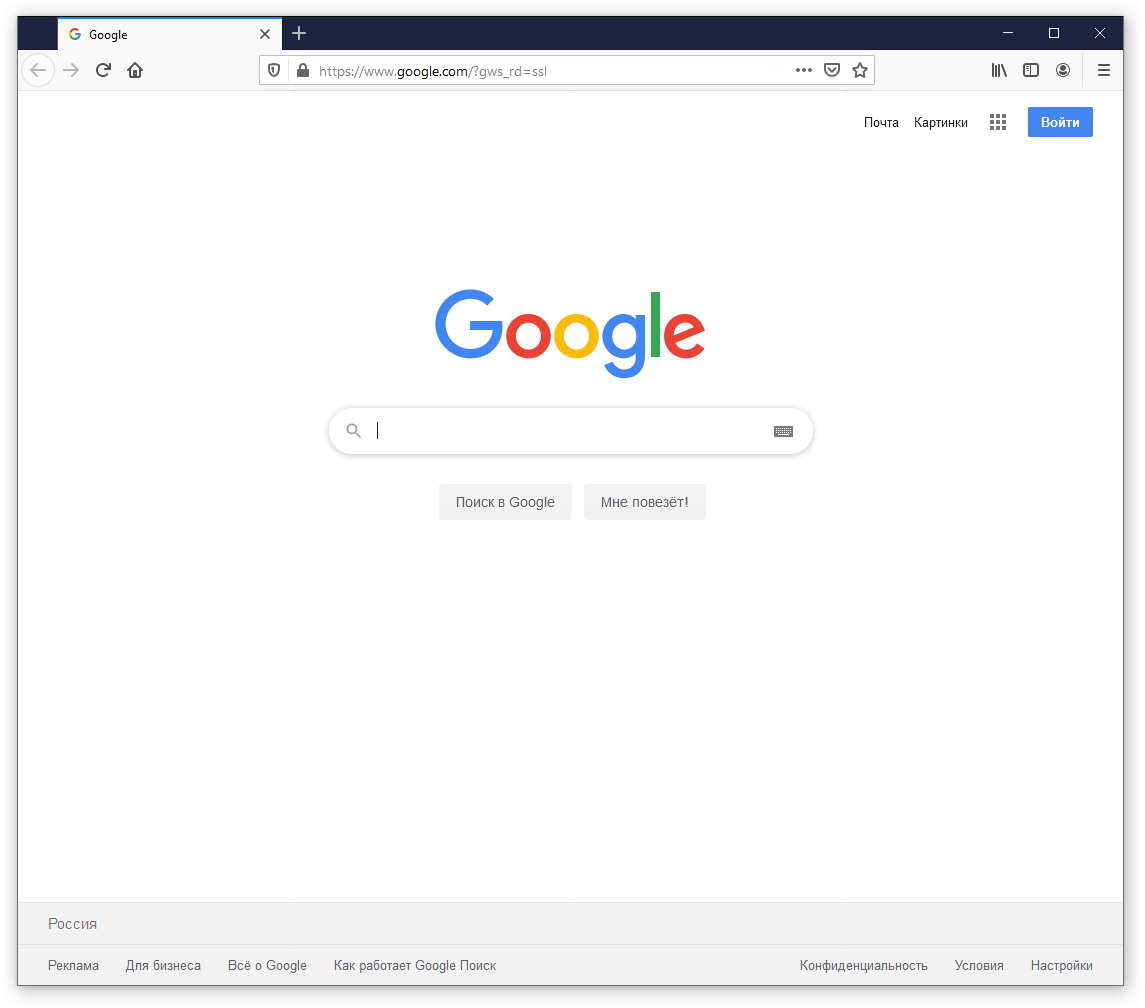


Рисунок 1 – Поисковый сайт

Далее происходит ввод поисковых запросов "Тестирование", "Эффективное тестирование", "Эффективное тестирование программ". На рисунке 2 изображен результат поискового запроса «Эффективное тестирование программ».

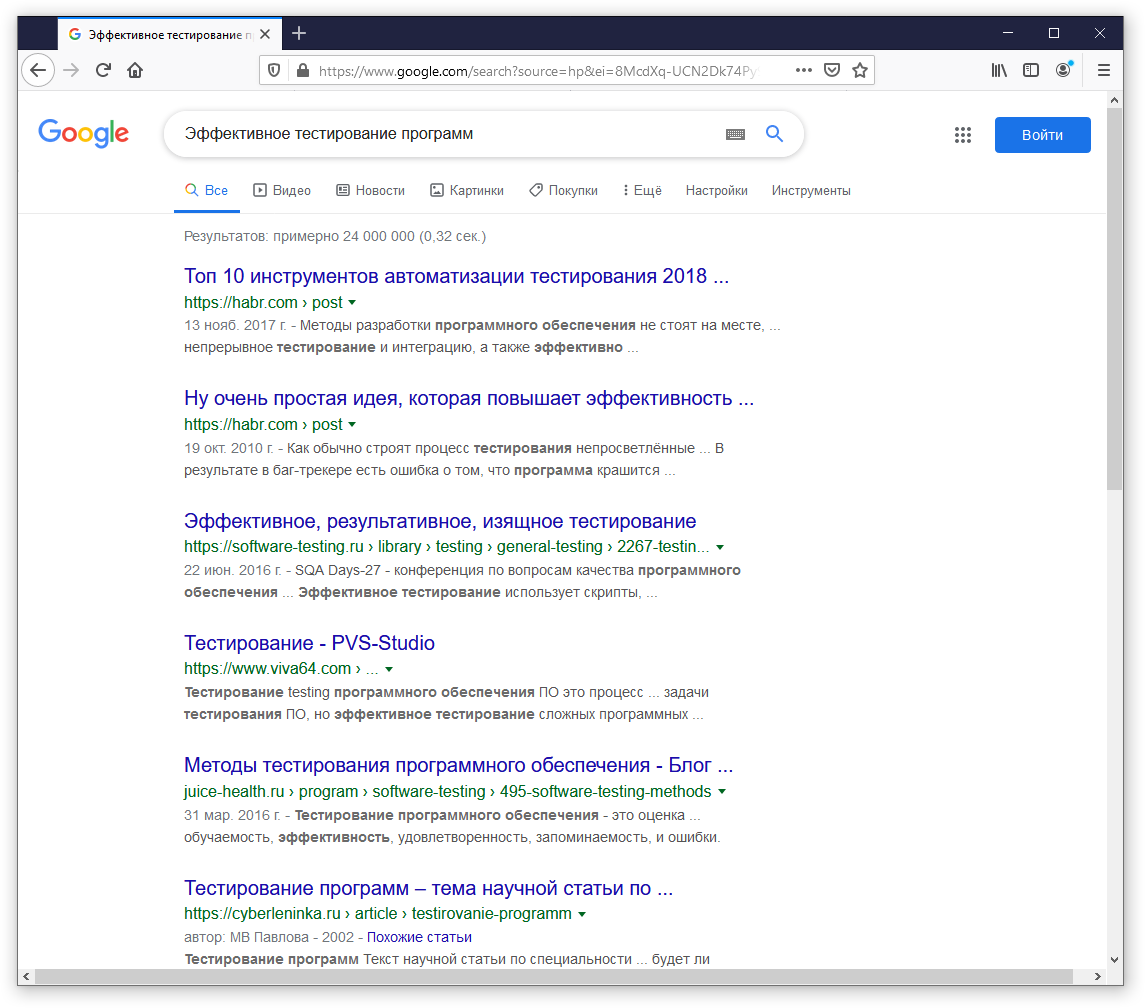


Рисунок 2 – Результат поискового запроса «Эффективное тестирование программ»

Скрипт выводит в консоль после каждого запроса кол-во результатов. После этого скрипт записывает в список все ссылки с первой страницы третьего запроса для того, чтобы сравнить сколько ссылок появится на первых трех страницах первого и второго запроса.

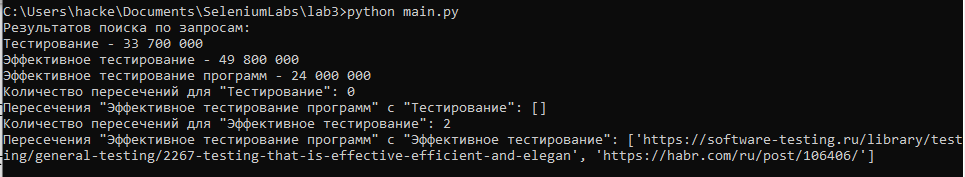


Рисунок 2 - Вид консоли после выполнения скрипта

Листинг 3.1. Содержание файла "main.py"

from selenium import webdriver

from selenium.webdriver.common.keys import Keys

from selenium.webdriver.support.ui import WebDriverWait

import selenium.common.exceptions as exceptions

from selenium.webdriver.common.by import By

from selenium.webdriver.support import expected\_conditions as EC

import re

import time

driver = webdriver.Firefox()

wait = WebDriverWait(driver, 10)

TEST\_SEARCHES = ["Тестирование", "Эффективное тестирование", "Эффективное тестирование программ"]

# Поиск

def search(text):

search\_line = driver.find\_element\_by\_class\_name("gLFyf.gsfi")

search\_line.clear()

search\_line.send\_keys(text+Keys.ENTER)

time.sleep(1.5)

# Смена страницы

def change\_page(new\_page = 1):

driver.find\_element\_by\_xpath(".//table[@id=\"nav\"]/tbody/tr/td["+str(new\_page+1)+"]/a").click()

time.sleep(1.5)

# Получение всех ссылок на странице

def get\_all\_links\_on\_page(search\_text, page = 1):

search(search\_text)

if (page > 1):

change\_page(page)

results = driver.find\_elements\_by\_class\_name("r")

links = []

for result in results:

links.append(result.find\_element\_by\_tag\_name("a").get\_attribute("href"))

return links

# Получение количества результатов

def get\_results\_count(search\_text):

search(search\_text)

result\_count = driver.find\_element\_by\_id("resultStats")

results = re.search(r"( {1}\d{1,3})+",result\_count.text)

return results.group(0)

# Выполнение первой части задания

def get\_links\_count():

print("Результатов поиска по запросам:")

print(TEST\_SEARCHES[0] + " -" + get\_results\_count(TEST\_SEARCHES[0]))

print(TEST\_SEARCHES[1] + " -" + get\_results\_count(TEST\_SEARCHES[1]))

print(TEST\_SEARCHES[2] + " -" + get\_results\_count(TEST\_SEARCHES[2]))

# Пересечение списков

def intersection(l1, l2):

intersection = []

for elem in l1:

if elem in l2:

intersection.append(elem)

return intersection

# Обработка пересечения двух списков с третьим

def get\_lists\_intersections(first, second, all):

first\_intersection = intersection(first,all)

second\_intersection = intersection(second,all)

return (first\_intersection, second\_intersection)

# Выполнение второй части задания

def get\_links\_repeats():

all\_links = get\_all\_links\_on\_page(TEST\_SEARCHES[2])

links\_on\_first\_request = []

links\_on\_second\_request = []

# Ищем ссылки на трех страницах первого и второго запроса

for i in range(1,4):

links\_on\_first\_request.extend(get\_all\_links\_on\_page(TEST\_SEARCHES[0],i))

links\_on\_second\_request.extend(get\_all\_links\_on\_page(TEST\_SEARCHES[1],i))

# Находим пересечения

intersections = get\_lists\_intersections(links\_on\_first\_request,links\_on\_second\_request,all\_links)

# Выводим информацию

print("Количество пересечений для \""+TEST\_SEARCHES[0]+"\": "+str(len(intersections[0])))

print("Пересечения \""+TEST\_SEARCHES[2]+"\" с \""+TEST\_SEARCHES[0] + "\": "+str(intersections[0]))

print("Количество пересечений для \""+TEST\_SEARCHES[1]+"\": "+str(len(intersections[1])))

print("Пересечения \""+TEST\_SEARCHES[2]+"\" с \""+TEST\_SEARCHES[1] + "\": "+str(intersections[1]))

if \_\_name\_\_ == "\_\_main\_\_":

driver.get("https://www.google.com")

get\_links\_count()

get\_links\_repeats()

driver.close()