Уральский федеральный университет

имени первого Президента России Б. Н. Ельцина

ЛАБОРАТОРНАЯ РАБОТА 1

ОТЧЕТ

Дисциплина «Web программирование»

Проверил:

к.т.н., доцент

Свинцов Д.

Выполнил:

Студент гр.

РИ-380014

Тюнькин А. Е.

Екатеринбург, 2020

1. Создать проект со следующей структурой:

Myproject/

├── about│

└──aboutme.html

└── index.html

В файле index.html написать 2 ссылки с прямым и абсолютным обращением к aboutme.html. В файле aboutme.html создать такие же ссылки на файл index.html.

Для начала создаем каталог в исходной папке. Для этого используем команду mkdir. Следом перемещаемся в созданную директорию командой cd – рисунок 1.

  
Далее создаем в ней файл index.html, используя vim.

Рисунок 1 — создание каталога



Рисунок 2 — создание index.html

Заполняем созданный файл следующим содержимым.



Рисунок 3 — index.html

Как мы можем увидеть, здесь будут две ссылки на файл aboutme.html подкаталога about, которые нам предстоит создать, используя все те же самые команды, что и в первом случае (Рисунок 4).

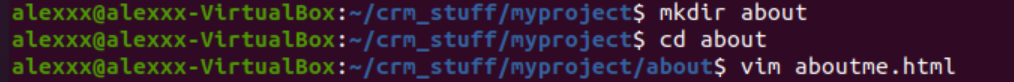


Рисунок 4 — создание директории about, файла aboutme.html

Файл aboutme.html будет выглядет так, как показано на рисунке 5 ниже.

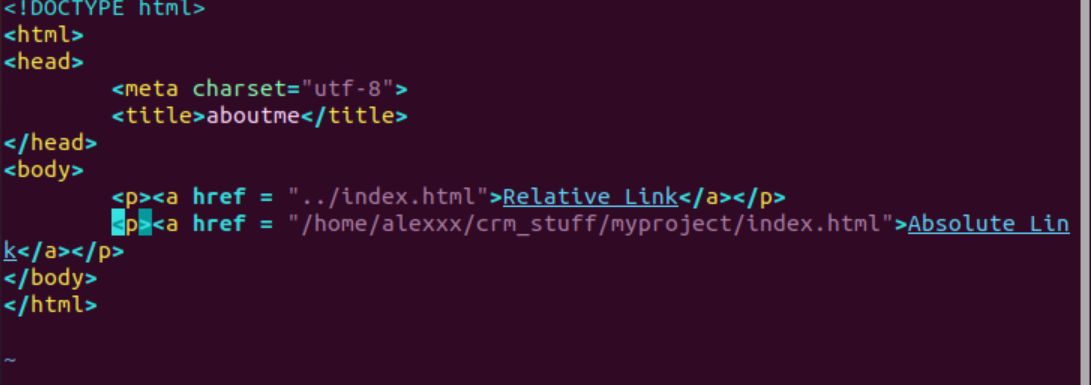


Рисунок 5 — aboutme.html

В браузере эти два файла будут отображаться так, как показано на рисунках 6, 7. Ссылки работают, переходы осуществимы.

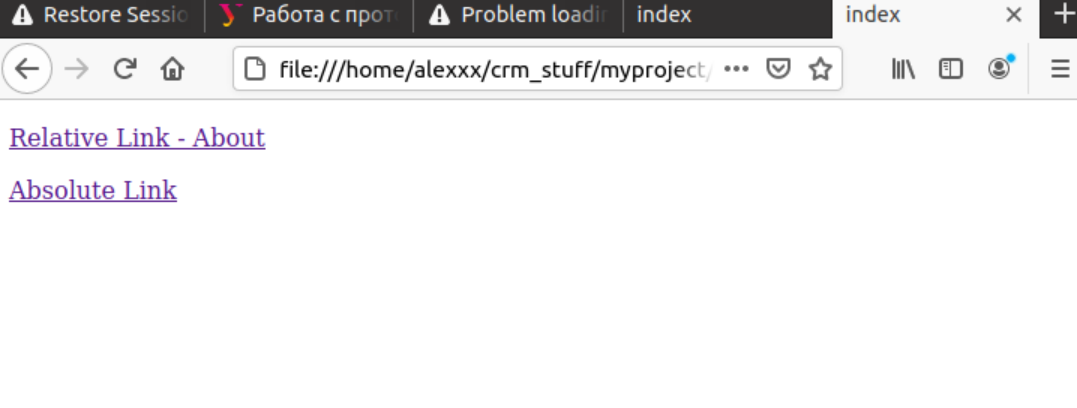


Рисунок 6 — index.html

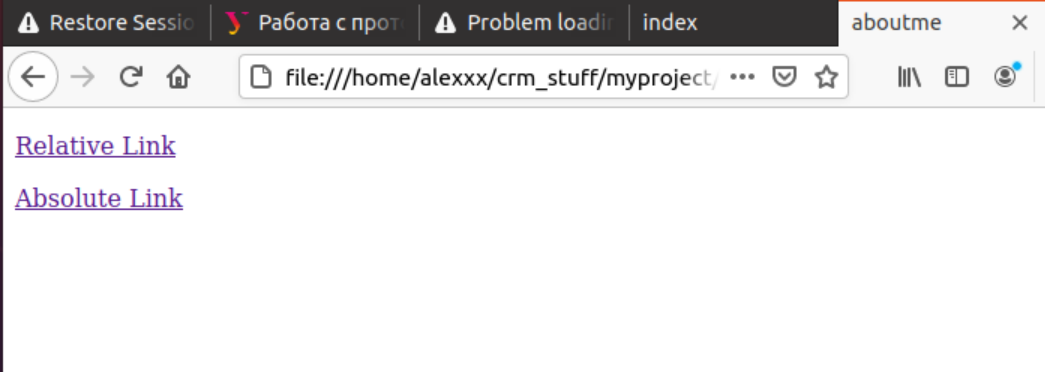


Рисунок 7 — aboutme.html

1. **Подключиться по telnet к http://wikipedia.org и отправить запрос, проанализировать ответ сервера. Описать работу HTTP протокола в данном случае.**

Для этого нам необходимо подключиться к серверу, используя следующую команду:

**telnet wikipedia.org 80** – установка соединения с wikipedia.org по восьмидесятому порту.

Далее незамедлительно, пока сервер не разорвал соединение, вводим следующее:

**GET /wiki/страница HTTP/1.1** - отправляем GET запроса к ресурсу /wiki/страница по протоколу HTTP версии 1.1

**Host: ru.wikipedia.org** – хост, к которому будет адресован запрос.

**User-Agent: Mozilla/5.0 (X11; U; Linux i686; ru; rv:1.9b5) Gecko/2008050509 Firefox/3.0b5** – клиентское приложение, которое мы используем.

**Accept: text/html** – типы данных, которые мы можем принять.

**Connection: close** – закрытие соединения.

**Результат выполнения представлен ниже на рисунке 8**.

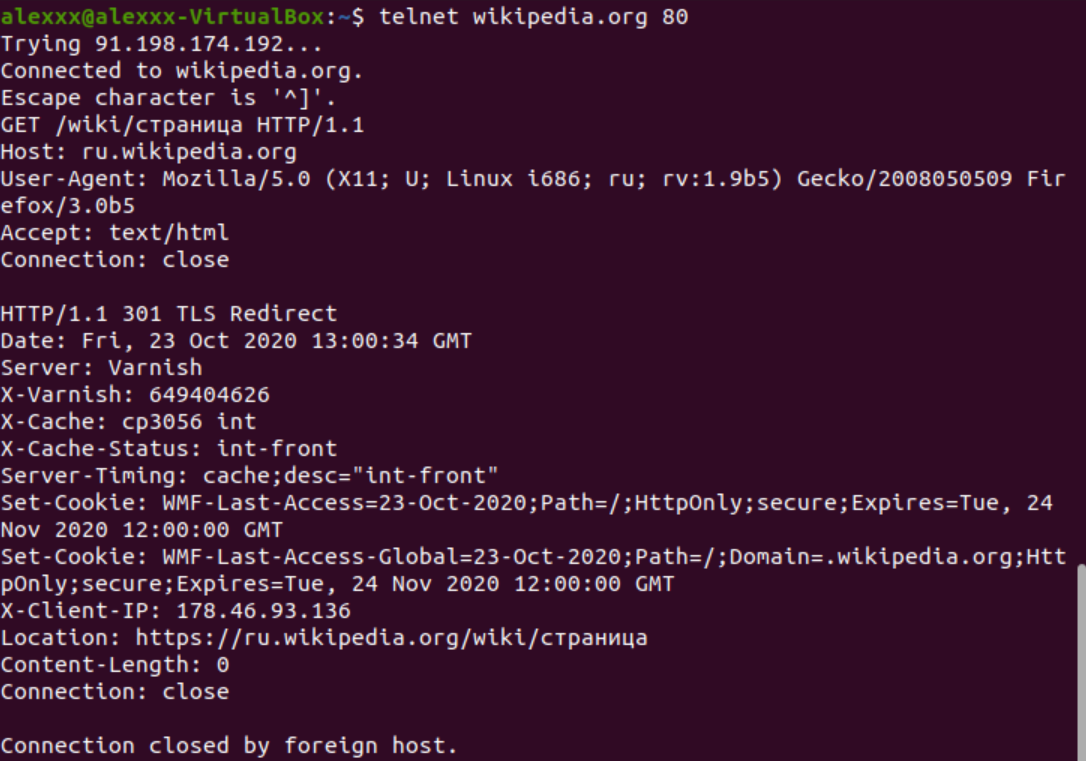


Рисунок 8 — результат выполнения

Как мы можем увидеть, нам пришел ответ с сервера с кодом 301. Мы были перенаправлены на другую страницу с расположением <https://ru.wikipedia.org/wiki/страница>. Запрошенный нами ресурс был перемещен.

1. **Отправить запросы на http://httpbin.org, проанализировать ответ и код состояния. Описать работу HTTP протокола в каждом запросе.**

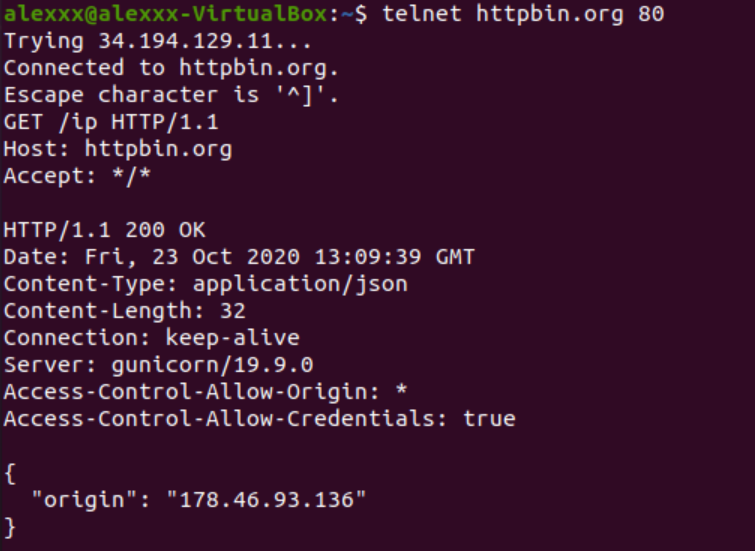
**Запрос и результат его выполнения предоставлены на рисунке 9.** 

Рисунок 9 — результат выполнения

Сначала нами было установлено соединение с сервером по 80 порту, затем введен запрос методом GET ресурса /ip. Далее, вводя параметр “Host: httpbin.org”, мы указываем хост, к котрому наш запрос будет адресован. В поле “Accept” происходит ввод данных, которые мы готовы принять в результате выполнения запроса, а его значение в нашем случае говорит о том, что мы принимаем данные любого типа.

Код состояния в результате выполнения запроса – 200, что говорит нам о его успешном выполнении. Результат выполнения запроса – данные в формате json, предоставляющие нам ip адрес. Длина ответа – 32 символа.

Далее выполним следующий запрос с методом GET с набором аргументов:

**GET /get?foo=bar&1=2&2/0&error=True HTTP/1.1**

**Host: httpbin.org**

**Accept: \*/\***

Результат его выполнения представлен на рисунке 10.



Рисунок 10 — результат выполнения

Результат выполнения запроса представлен с состоянием 200, что означает, что запрос был выполнен успешно. В результате запроса нам был передан ответ в формате json – аргументы, переданные нами серверу.

Далее отправим запрос методом POST. Он будет выглядеть, как показано на рисунке 11.

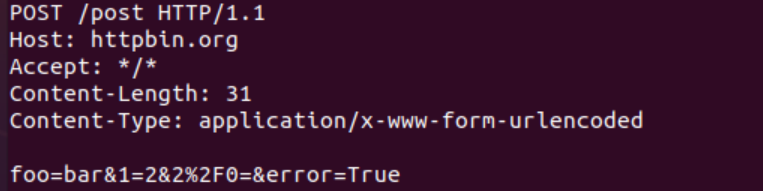


Рисунок 11 — POST-запрос

В поле Content-Length вводим длину запроса в символах, в нашем случае это 29, включая символ переноса. Результат выполнения запроса представлен на рисунке 12. Завершился с кодом 200.

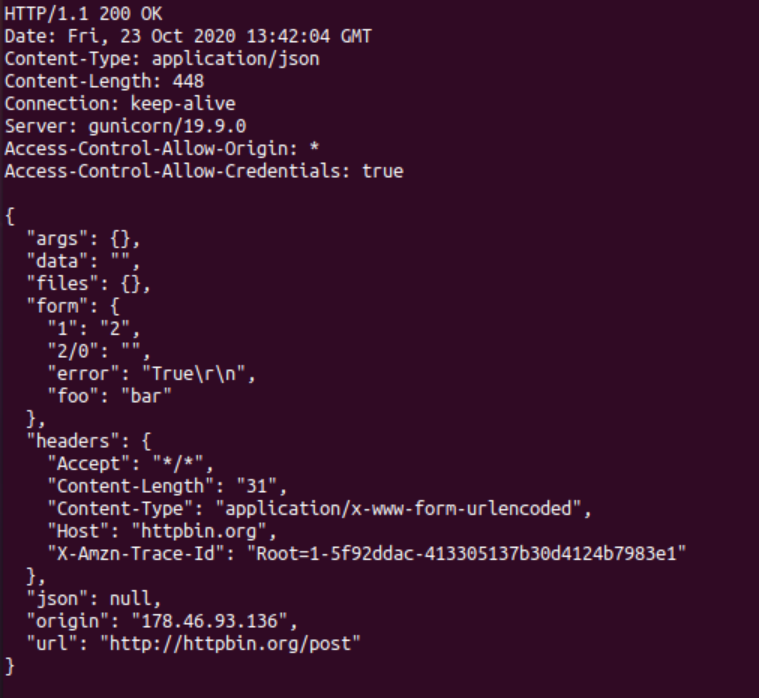


Рисунок 12 — результат выполнения

При введении запроса, с неверным указанием его длины, пусть будет 20, мы получим следующий результат (рисунок 13). Код 400, Bad Request. Вполне ожидаемая ситуация.

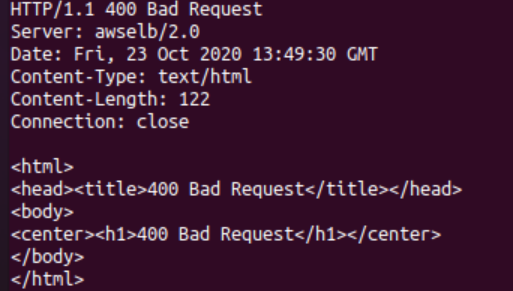


Рисунок 13 — результат выполнения

Далее мы отправляем запрос на установку куки. Запрос и результат его выполнения предоставлены на рисунке 14.



Рисунок 14 — результат запроса на установку cookies

Как мы можем видеть, мы получили ответ с кодом состояния 302 – код, сигнализирующий о том, что запрошенный ресурс временно перемещен на другую страницу. Результат выполнения запроса – перенаправление на другую страницу.

Далее нам нужно посмотреть список установленных куки. Для этого нам необходимо отправить следующий запрос:

**GET /cookies HTTP/1.1**

**Host: httpbin.org**

**Accept: \*/\***

Результат предоставлен на рисунке 15.

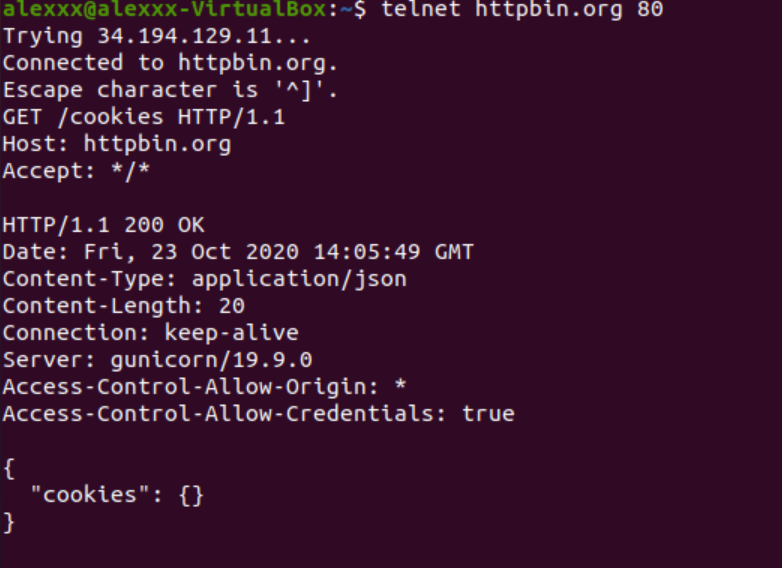


Рисунок 15 — результат выполнения

Как мы можем видеть, запрос был выполнен успешно, в результате запроса нам был передан ответ в формате json, из которого мы можем заключить, что список куки пуст.

Далее нам нужно отправить запрос на перенаправление. Запрос и результат его выполнения представлены на рисунке 16.

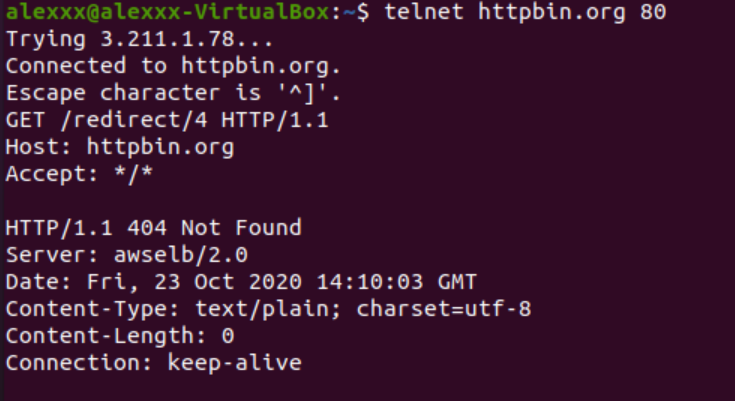


Рисунок 16 — результат запроса на перенаправление

Результат выполнения говорит нам о том, что ресурс не найден. Это мы можем заключить исходя из кода состояния 404.

1. **Создать HTML форму c action="http://httpbin.org/post" method="POST" и enctype="multipart/form-data"**

**Добавить в форму поля firstname, lastname, group, message (textarea), myimg (file).**

**Проверить результат отправки данных формы.**

**Проанализировать ответ. Описать работу HTTP протокола в данном случае.**

Форма в ее итоговой реализации выглядит следующим образом (Рисунок 17):



Рисунок 17 — содержание html-файла с формой, содержащий пост-запрос к серверу httpbin.org

Вид формы при ее открытии в браузере (Рисунок 18):

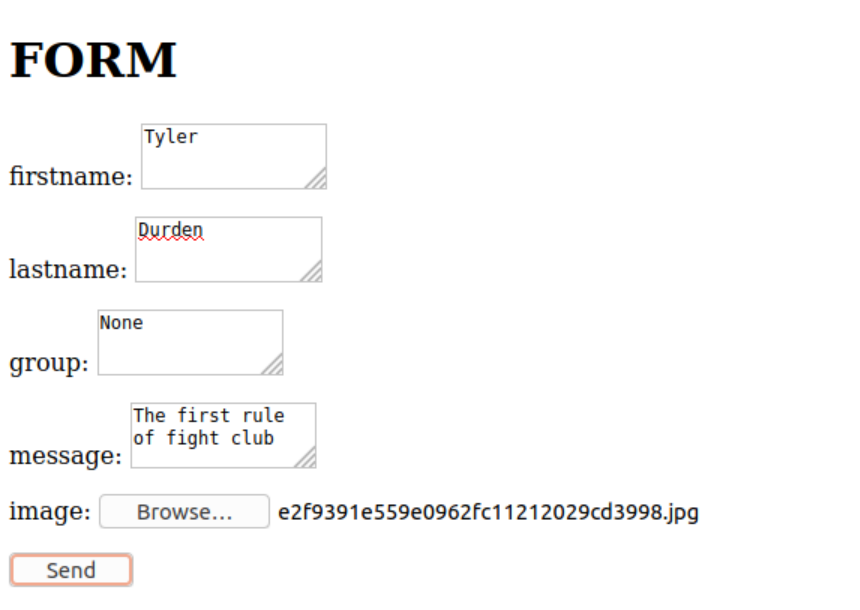


Рисунок 18 — форма в браузере

Результат отправки формы (рисунки 19 и 20):



Рисунок 19 — результат отправки формы

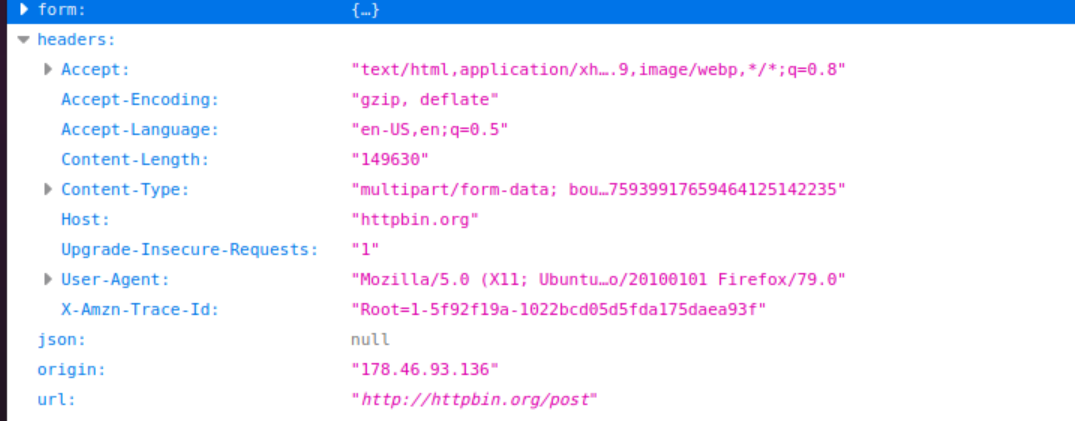


Рисунок 20 — результат отправки формы

Результат — форма успешно отправлена на сервер.

В ходе данной лабораторной работы были успешно усвоены основные принципы работы с telnet.