Задача по программированию сокетов 1: Веб-сервер

В этой лабораторной вы познакомитесь с основами программирования сокетов для TCP-соединений на языке Python: как создавать сокет, привязывать его к определенному адресу и порту, а также отправлять и получать HTTP-пакеты. Вы также изучите некоторые форматы HTTP-заголовков.
Вам предстоит создать веб-сервер, который будет обрабатывать один HTTP-запрос. Ваш веб-сервер должен будет принять и проанализировать HTTP-запрос, получить требуемый файл из файловой системы сервера, создать ответное HTTP-сообщение, состоящее из запрошенного файла и предваряющих его строк заголовка, а затем отправить ответ непосредственно клиенту. Если требуемый файл на сервере отсутствует, клиенту должно вернуться HTTP-сообщение `404 Not Found`.

Код

Ниже представлена заготовка программы веб-сервера. Вам нужно ее завершить. Места, которые вы должны заполнить своим кодом, помечены как **`#Начало вставки`** и **`#Конец вставки`**. В каждом таком месте может содержаться одна или несколько строк вашего кода.

Запуск сервера

Поместите файл HTML (например, HelloWorld.html) в тот же каталог, где находится серверная программа. Запустите программу-сервер. Определите IP-адрес хоста с запущенным сервером (например, 128.238.251.26). На другом хосте откройте браузер и введите в его адресную строку соответствующий URL. Например:

```
http://128.238.251.26:6789/HelloWorld.html
```

HelloWorld.html – это имя файла, который вы поместили в каталог сервера. Обратите внимание на номер порта после двоеточия. Вы должны использовать здесь тот же порт, что указан в коде сервера. Мы использовали порт 6789. Браузер должен отобразить содержимое файла HelloWorld.html. Если вы опустите значение `:6789`, браузер по умолчанию обратится к порту 80, и загрузится страница с сервера, если конечно ваш сервер «слушает» порт 80.
После этого попробуйте загрузить файл, которого на сервере нет. Вы должны получить сообщение `404 Not Found`.

Задание

Завершить серверный код и сделать снимки экрана клиентского браузера, подтверждающие получение HTML-файла с сервера.

Шаблон кода веб-сервера на языке Python

```
#импортируем модуль для работы с сокетами
from socket import *
serverSocket = socket(AF_INET, SOCK_STREAM)
#Подготавливаем сокет сервера
#Начало вставки

#Конец вставки
while True:
      #Устанавливаем соединение
      print 'Готов к обслуживанию...'
      connectionSocket, addr = #Начало вставки       #Конец вставки
      try:
            message = #Начало вставки     #Конец вставки
            filename = message.split()[1]
            f = open(filename[1:])
            outputdata = #Начало вставки  #Конец вставки
            #Отправляем в сокет одну строку HTTP-заголовка
            #Начало вставки

            #Конец вставки
            #Отправляем содержимое запрошенного файла клиенту
            for i in range(0, len(outputdata)):
                  connectionSocket.send(outputdata[i])
            connectionSocket.close()
      except IOError:
            #Отправляем ответ об отсутствии файла на сервере
            #Начало вставки

            #Конец вставки
            #Закрываем клиентский сокет
            #Начало вставки

            #Конец вставки
serverSocket.close()
```

Дополнительные задания

1. Разработанный вами веб-сервер обрабатывает только один HTTP-запрос. Реализуйте многопоточный сервер, который мог бы обслуживать несколько запросов одновременно. Сначала создайте основной поток (процесс), в котором ваш модифицированный сервер ожидает клиентов на определенном фиксированном порту. При получении запроса на TCP-соединение от клиента он будет устанавливать это соединение через другой порт и обслуживать запрос клиента в отдельном процессе. Таким образом, для каждой пары запрос-ответ будет создаваться отдельное TCP-соединение в отдельном процессе.
2. Вместо использования браузера напишите собственный HTTP-клиент для тестирования вашего веб-сервера. Ваш клиент будет подключаться к серверу с помощью TCP-соединения, отправлять ему HTTP-запрос с помощью метода GET и отображать ответ сервера в качестве результата.
   Клиент должен будет в качестве входных параметров принимать аргументы командной строки, определяющие IP-адрес или имя сервера, порт сервера и полный путь хранения файла на сервере. Формат команды для запуска клиента следующий:
   `client.py` *`хост_сервера порт_сервера имя_файла`*