**CMPT 371: Data Communications and Networking**

**Programming Project 2**

**Description**

In this programming project, you will develop a small web proxy server which is able to cache web pages. It is a very simple proxy server which only understands simple GET requests, but is able to handle different kinds of objects such as HTML pages and images. This project is based on [Socket Programming Assignment 4](http://nsl.cs.sfu.ca/peters/15/371/Socket4_ProxyServer.pdf) from the textbook companion website, but we have modified the project. The project includes [skeleton code](http://nsl.cs.sfu.ca/peters/15/371/Proxy_Code.pdf) that will provide you with a starting point, but you will also have to make some modifications as described below. Please read the entire description below before starting to develop your code.

Generally, when a client (e.g. a web browser) makes a request, the request is sent to the web server. The web server then processes the request and sends back a response message to the requesting client. In order to improve the performance, a proxy server can be used between the client and the web server and the client requests objects via the proxy server. Now, both the request message sent by the client and the response message delivered by the web server pass through the proxy server. If the proxy server does not find a cached copy of a requested object, then it forwards the client’s request to the web server, and the web server will then generate a response message and deliver it to the proxy server which creates a cached copy. The proxy server then sends the requested object to the client. You should be able to complete this part of the project by filling in the missing code in the [skeleton code](http://nsl.cs.sfu.ca/peters/15/371/Proxy_Code.pdf).   
  
Your proxy server should also correctly handle responses from a web server when requesting an object that is not available from the web server. If the web server receives such a request, it will send a "404 Not Found" response to your proxy server, and such a response usually has no response body. If you base this part of your proxy server on the skeleton code from the companion website, then it will try to read the response body, which does not exist, and it will throw an exception. Instead, you need to modify the skeleton code so that your proxy server handles this exception and sends a "Not Found" response to the client (browser) with a response body "404 Not Found".  
  
Your proxy server should respond correctly when requests with the following format are entered into your browser:  
http://servername:serverport/webserver/filename\*  
  
Examples of requests with this format are the following:  
http://localhost:8888/www.google.com  
http://localhost:8888/www.google.com/index.html  
http://192.168.1.1:8888/www.google.com  
http://192.168.1.1:8888/www.google.com/index.html  
  
Your proxy server should also handle incorrect formats that are different from the above.   
  
You can develop and test your proxy server with both the browser and proxy server running on the same computer (using localhost as the IP address). However, the code that you submit for your proxy server must also work with the browser and proxy server running on different computers. In this case, localhost will be replaced by the IP address of the computer on which your proxy server is running. Your proxy server should be executed with a command of the form:  
python ProxyServer.py proxyserverIP proxyserverPort

**Submission**

1. Complete the proxy server code according to the instructions above being sure to comment your code well.
2. Test your code. If there are any bugs or unresolved issues, you should describe them and explain your attempts to resolve them.
3. Include instructions explaining how to start your proxy server and anything else that you want us to know about your program. It must be possible for us to test your code without making any changes to the code.
4. Submit an archive containing your proxy server code and your documentation (instructions, discussion of unresolved issues) online at [https://courses.cs.sfu.ca](https://courses.cs.sfu.ca/).

<http://nsl.cs.sfu.ca/peters/15/371/Socket4_ProxyServer.pdf>

这个网址是框架结构