Proxy lab 实验报告

一、basic：

这一部分可以参考书上的代码，main 函数还是打开监听套接字、不断接收连接请求，执行事务、关闭连接。主要区别在处理函数 doit：

处理函数中利用 parse_uri 函数得到解析后的请求头和请求行，然后利用 send_to_server 函数向服务器发出请求，然后将返回的信息保存起来即可。

其中头部信息利用 map 结构储存，但是由于使用的是 c 语言，所以是采用的 struct 数组实现。在解析 uri 的时候要注意不同部分之间的分隔符。

二、Concurrency：

这一部分采用了最简单的实现方式，即用 thread 来处理每个 connection，只需要写一个 thread 函数，其中注意将线程分离，让这个线程计数结束后自己回收资源，然后将 doit 函数放到里面即可。

三、Cache：

这一部分参考书上的读者-写者问题部分的思路，利用 reader 和 writer 函数实现 cache 的读写。在 main 函数中加入 cache 的初始化；在处理函数中先利用 reader 函数判断是否是之前请求过的数据，如果是，直接返回，不是再继续操作，然后在处理函数的最后将数据存到 cache 里即可。

实验结果：

```
4: godzilla.jpg
   Fetching ./tiny/godzilla.jpg into ./.proxy using the proxy
   Fetching ./tiny/godzilla.jpg into ./.noproxy directly from Tiny
   Comparing the two files
   Success: Files are identical.
5: tiny
   Fetching ./tiny/tiny into ./.proxy using the proxy
   Fetching ./tiny/tiny into ./.noproxy directly from Tiny
   Comparing the two files
   Success: Files are identical.
Killing tiny and proxy
basicScore: 40/40

*** Concurrency ***
Starting tiny on port 26958
Starting proxy on port 19537
Starting the blocking NOP server on port 10555
Trying to fetch a file from the blocking nop-server
Fetching ./tiny/home.html into ./.noproxy directly from Tiny
Fetching ./tiny/home.html into ./.proxy using the proxy
Checking whether the proxy fetch succeeded
Success: Was able to fetch tiny/home.html from the proxy.
Killing tiny, proxy, and nop-server
concurrencyScore: 15/15

*** Cache ***
Starting tiny on port 23308
Starting proxy on port 16331
Fetching ./tiny/tiny.c into ./.proxy using the proxy
Fetching ./tiny/home.html into ./.proxy using the proxy
Fetching ./tiny/csapp.c into ./.proxy using the proxy
Killing tiny
Fetching a cached copy of ./tiny/home.html into ./.noproxy
Success: Was able to fetch tiny/home.html from the cache.
Killing proxy
cacheScore: 15/15

totalScore: 70/70
```