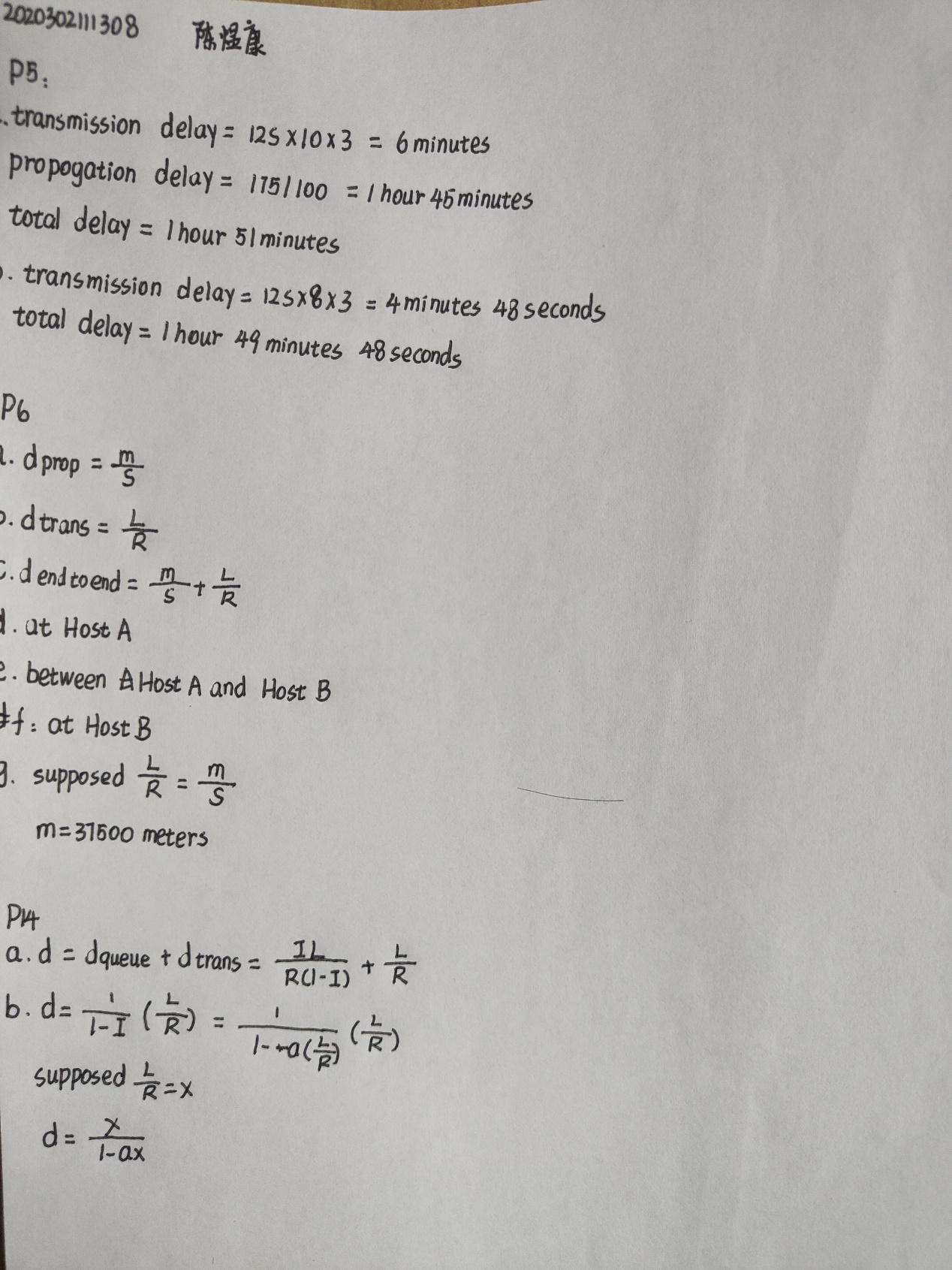
**Week1:**

****

**Week2:**

**P6:**

**Obtain the HTTP/1.1 specification (RFC 2616). Answer the following questions:**

1. **Explain the mechanism used for signaling between the client and server to indicate that a persistent connection is being closed. Can the client, the server, or both signal the close of a connection.**

A persistent connection will be kept util either server or client send a connection header including the connection-token close. Both clients and servers can signal the close of persistent connection.

1. **What encryption services are provided by HTTP?**

No. HTTP doesn’t provide encryption service.

1. **Can a client open three or more simultaneous connections with a given server?**

No. A single-user client should not maintain more than two connections with any server or proxy.

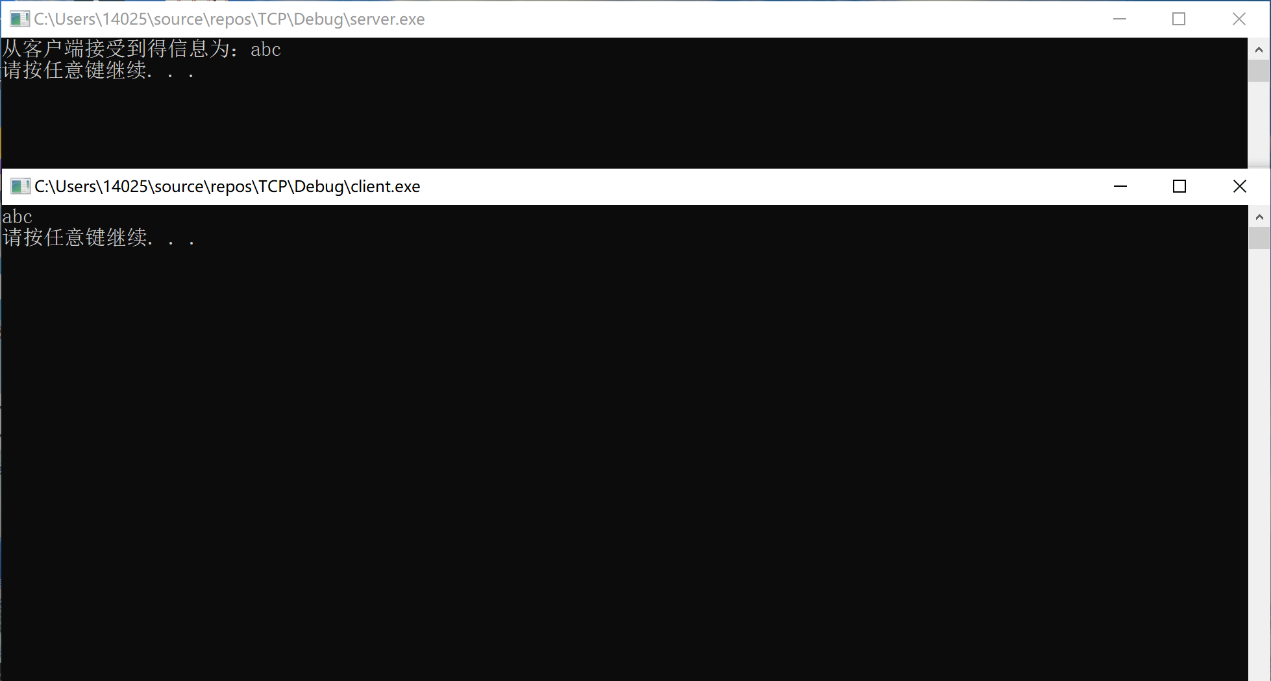
1. **Either a server or a client may close a transport connection between them if either one detects the connection has been idle for some time. Is it possible that one side starts closing a connection while the other side is transmitting data via this connection? Explain.**

Possible. While client is preparing to send a new request, server might detect that the connection has been idle for some time and it will close it.

**P12：**

**Write a simple TCP program for a server that accepts lines of input from a client and prints the lines onto the server’s standard output. (You can do this by modifying the TCPServer.py program in the text.) Compile and execute your program. On any other machine that contains a Web browser, set the proxy server in the browser to the host that is running your server program; also configure the port number appropriately. Your browser should now send its GET request messages to your server, and your server should display the messages on its standard output. Use this platform to determine whether your browser generates conditional GET messages for objects that are locally cached.**

**Result:**



**Code:**

#include<iostream>

#include<string>

#include<winsock2.h>

#pragma warning(disable : 4996)

using namespace std;

#pragma comment(lib,"ws2\_32.lib")

const int BUF\_SIZE = 100;

int main()

{

WSADATA wsadata;

WSAStartup(MAKEWORD(2, 2), &wsadata);

SOCKET serSock = socket(AF\_INET, SOCK\_STREAM, IPPROTO\_TCP);

sockaddr\_in sockAddr;

memset(&sockAddr, 0, sizeof(sockAddr)); //用0填充每一个字节

sockAddr.sin\_family = PF\_INET; //使用IPv4地址

sockAddr.sin\_addr.s\_addr = inet\_addr("127.0.0.1"); //具体的ip地址

sockAddr.sin\_port = htons(1234); //具体的端口号

//绑定套接字

bind(serSock, (SOCKADDR\*)&sockAddr, sizeof(sockAddr));

listen(serSock, 20);

//接受客户端请求

SOCKADDR clientAddr;

int clientAddr\_size = sizeof(clientAddr);

SOCKET clientSock = accept(serSock, (SOCKADDR\*)&clientAddr, &clientAddr\_size);

//向客户端发送消息

/\*char str[] = "hello world!!";

char\* str1 = str;

send(clientSock, str, strlen(str1) + sizeof(char), NULL);\*/

char infoBuff[MAXBYTE] = { 0 };

recv(clientSock, infoBuff, MAXBYTE, NULL);

cout << "从客户端接受到得信息为：" << infoBuff << endl;

//关闭套接字

closesocket(clientSock);

closesocket(serSock);

WSACleanup();

system("pause");

return 0;

}