

计算机网络实习

FTP实验



学生学号： 2020011702

学生姓名： 蒋继庆

学生班级： 计算机20-1班

中国石油大学（北京）计算机系

报告时间：2022年6月9日

目录

1.程序架构 ……………………………………………………………………3

2.实现方法 ……………………………………………………………………3

2.1 client端…………………………………………………………………3

2.2 server端…………………………………………………………………4

3.实验测试结果 ………………………………………………………………4

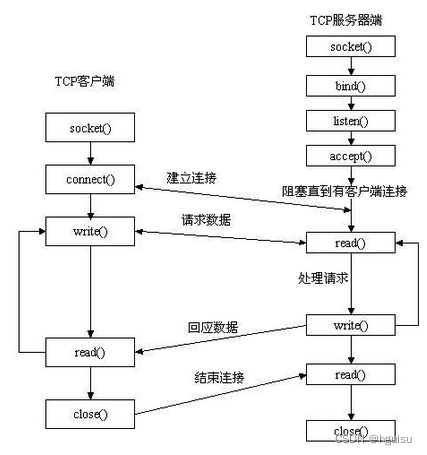
4.遇到的问题及解决方法 ……………………………………………………8

5.思考题解答 …………………………………………………………………8

1. 程序架构

本次实验我在自己的ubuntu上使用c语言通过socket网络编程通信方法实现了一个简单的FTP，可以支持get/put/ls/dir/pwd/cd/mkdir/rmdir/quit/?这些功能。我分别设计了client端和server端。运行时只需将两个文件通过gcc -o client client.c以及gcc -o server server.c编译链接生成client和server两个可执行文件，然后将这两个文件分别放到不同的文件夹（在本机测试时），然后首先./server启动服务器，然后./client 127.0.0.1 8888启动客户端。如果是不同主机的话，需要客户端输入正确的ip地址，即./client ip地址 8888即可连接到服务器。然后是登录验证，当正确的输入了用户名root以及密码1234后，便会连接成功。接下来就是客户端和服务器通过send和recv函数相互通信。本代码可以支持多个客户端连接服务器。

下面这张图片是代码的总体逻辑图，图片来源于CSDN。



1. 实现方法

Client端

**1、主函数：**

（1）用户登录名于用户登陆密码匹配

if ((strcmp(username, NAME) == 0) && (strcmp(userpswd, PSWD) == 0))

printf("Login successfully!\n");

else

return 0;

（2）创建套接字，返回一个可用的套接字描述符

if(( sockfd = socket( AF\_INET, SOCK\_STREAM, 0)) < 0)

（3）指定地址协议簇 、指定端口号

server\_addr.sin\_family = AF\_INET;

server\_addr.sin\_port = htons(atoi(argv[2]));

if (inet\_pton(AF\_INET, argv[1], &server\_addr.sin\_addr) < 0)

{

printf("invalid ip address\n!");

return 1;

}

server\_addr.sin\_addr.s\_addr = inet\_addr(argv[1]);

（4）while无限循环接受命令，进行命令匹配

**2、实现get功能的函数**

（1）将命令送往server,这里的“G%s”中的G相当于控制标志，服务器端通过recv将字符串读入到buf中，根据buf[0]是否等于G来区别是否是命令get

sprintf(buf, "G%s", command+4);

send(sockfd, buf, N, 0);

（2）接收服务器传回的文件，打开文件并写入

fd = open(command+4, O\_WRONLY | O\_CREAT | O\_TRUNC, 0666)

write(fd, buf, flag);

**3、实现put功能函数**

（1）打开文件

fd = open(command+4, O\_RDONLY)

（2）传送命令

sprintf(buf, "P%s", command+4);

send(sockfd, buf, N, 0);

（3）送入文件内容

send(sockfd, buf, flag, 0);

**4、实现ls功能**

（1）定义目录描述符和存储目录的文件信息

DIR \*mydir;

struct dirent \*myitem;

mydir=opendir(".");

myitem = readdir(mydir)

**5、其余函数大同小异，概括为传送命令到server端，server端执行相应的函数**

Server端

1. **主函数**

（1）定义一个数据流套接字

if(( listenfd = socket( AF\_INET, SOCK\_STREAM, 0)) < 0 )

（2）设定端口和IP地址

server\_addr.sin\_family = AF\_INET;

server\_addr.sin\_port = htons(PORT\_NUM);

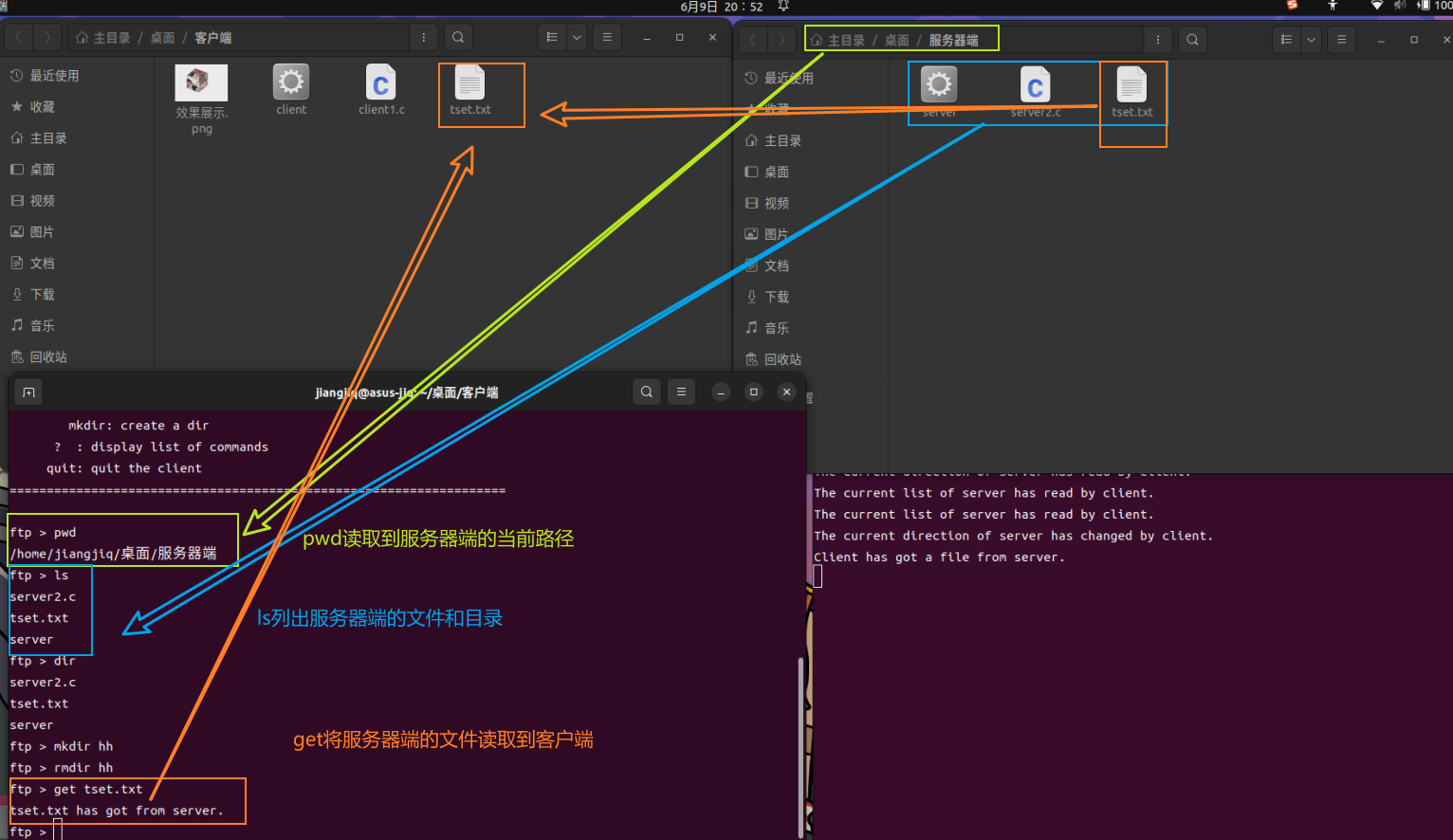
server\_addr.sin\_addr.s\_addr = htonl(INADDR\_ANY);

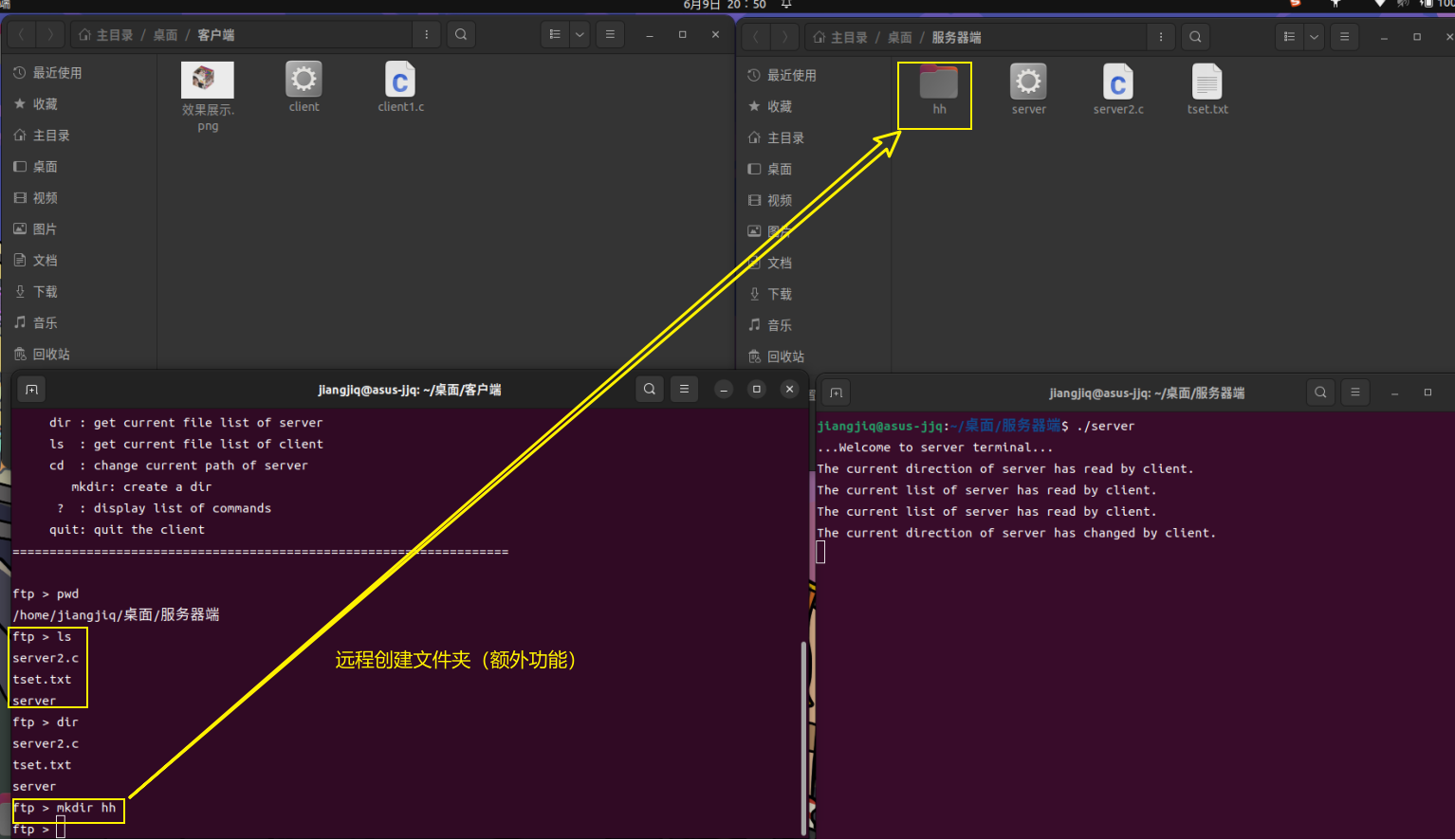
（3）利用while循环进行监听

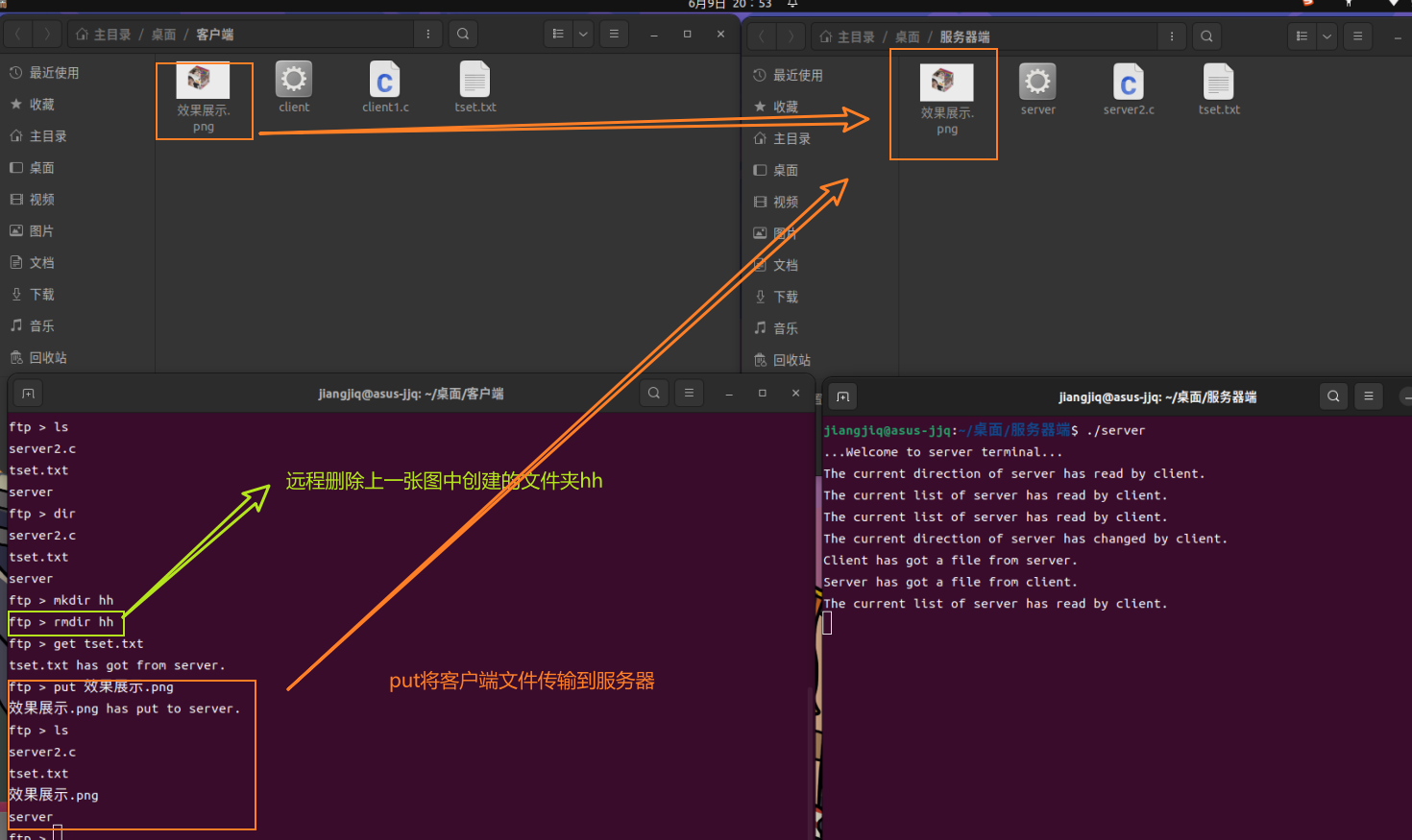
**2、其他函数大同小异**

1. 实验测试结果

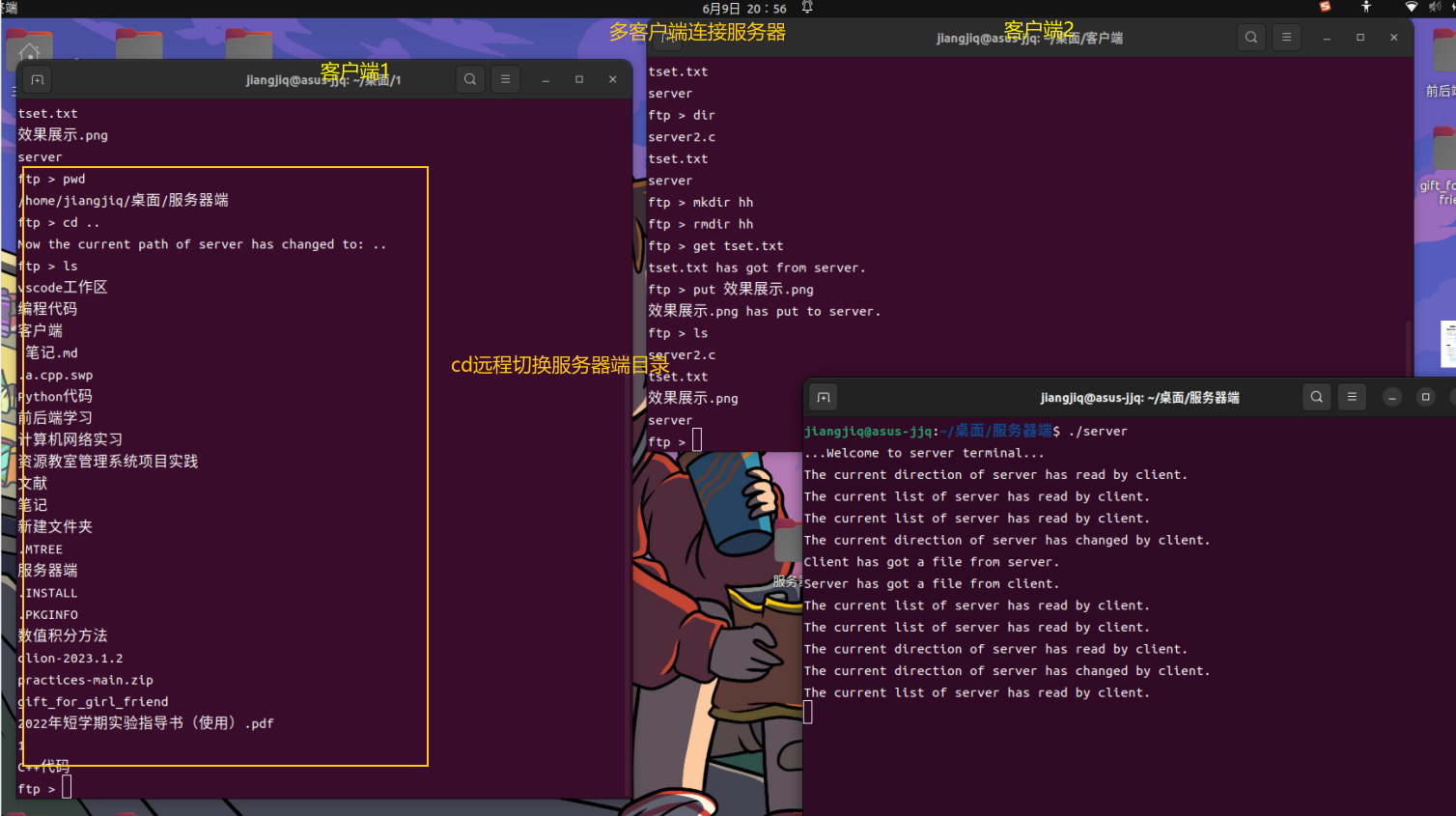
测试前文件夹内容











1. 遇到的问题和解决方法

由于时间缘故，我在实现以上功能的时候其实并不符合RFC959规范，起初不知道如何实现这些函数，于是我参考往届学姐的代码，自己通过send发送指令，通过发送的字符串的首字符buf[0]来区分客户端的功能。但实际应该是有一个控制链接发送命令，数据链接传输文件等。还有登录验证我由于学懂后时间有点来不及了，只是简单的在客户端本机加了个字符串判断来实现登录验证。但其实应该在客户端读取用户名密码，send发送，服务器端recv接收用户名和密码，然后判断是否允许登录，然后将结果返回给客户端，这应该是正解。我的代码存在的许多不满足RFC959的地方主要是在端口，登录验证，两个端口，控制链接和数据链接分开这些地方。很大一部分原因是自己之前没有学过socket编程，走了一些弯路，来不及完整实现了。我在之后的假期也会遵循规范好好实现一下正解的代码。

1. 思考题解答

（1）在 FTP 中，为什么要建立两个 TCP 连接来分别传送命令和数据？

FTP协议需要建立两个TCP连接，一个用于控制，一个用于数据传输。这是因为FTP协议需要进行双向通信，同时需要传输大量的数据。如果使用一个TCP连接，会导致控制信息和数据混杂在一起，使得数据传输过程中的控制信息无法及时传达，从而影响传输效率和可靠性。因此，FTP协议采用两个TCP连接，一个用于传输控制信息，另一个用于传输数据，以保证数据传输的高效性和可靠性。

（2）主动方式和被动方式的主要区别是什么？为何要设计这两种方式？

FTP协议中，主动方式和被动方式是两种不同的数据传输方式。

主动方式：在主动方式下，客户端在建立控制连接后，向服务器端的数据端口发起连接请求，服务器端则在自己的控制端口上监听客户端的数据连接请求，并在收到请求后建立数据连接。

被动方式：在被动方式下，客户端在建立控制连接后，向服务器端的控制端口发起PASV命令请求，服务器端则在自己的数据端口上监听客户端的数据连接请求，并将数据端口号告知客户端，客户端再通过该端口与服务器建立数据连接。

主要区别：

1. 主动方式需要服务器端主动向客户端发起数据连接请求，而被动方式则是客户端主动向服务器端发起数据连接请求。

2. 主动方式下，服务器端的数据端口固定为20，而被动方式下，服务器端的数据端口是动态分配的。

为何要设计这两种方式？

主动方式和被动方式是为了克服FTP协议在NAT（网络地址转换）下的问题而设计的。在NAT环境下，FTP协议的主动方式会遇到一些问题，因为NAT会修改FTP协议中的IP地址和端口号，导致服务器无法将数据传送到客户端。为了解决这个问题，被动方式应运而生。被动方式下，服务器端主动告知客户端数据端口号，客户端再通过该端口与服务器建立数据连接，从而避免了NAT带来的问题。区别：主动模式要求客户端和服务器端同时打开并且监听一个端口以建立连接。在这种情况下，客户端由于安装了防火墙会产生一些问题。所以有了被动模式。被动模式只要求服务器端产生一个监听相应端口的进程，这样就可以绕过客户端安装了防火墙的问题。

（3）当使用 FTP 下载大量小文件的时候，速度会很慢，这是什么缘故？可以怎样改进？

由于文件太小无法充分利用带宽。解决方法：小文件打包上传，充分利用带宽

每次寻找文件花费很多时间，性价比低。解决方法：开启多线程服务

1. 致谢

在整个实现代码的过程中，我很感谢卢玥辰学姐和查夏如学姐的代码，这为我提供了一些思路，也让我对整个过程有了一个更深的理解，目前的代码大多数是基于卢玥辰学姐的代码，我在这个基础上增加了一些新的额外功能。但我明白多客户端连接应该使用多进程等，我会在之后的假期好好研究后一位学姐的实现了所有功能的正解做法。