一、考试时间：120 分钟
二、考试答案检查：所有源代码必须在考试结束前在网络学堂完成提交。考试结束后开始考试代码的现场演示，检查代码版本以网络学堂为准，迟交代码酌情扣分。
三、代码提交方式：在本地机器上以自己的“学号”为名称建立目录，在此目录中为每个题目建立子目录（如 1、2 ……），将答案源码拷贝到相应目录中，再将整个目录压缩成 zip 或 rar 文件，通过网络学堂上传答案。
四、请确保上载内容正确，责任自负。

前两道题每题 30 分，第三题 40 分。

1、远程绘图：通过 Qt Socket API 编写网络通信程序，实现在客户端对话框输入绘图数据信息，点击绘图按钮后，在服务器端实时绘出指定的图形。

客户端显示一个对话框，通过该对话框可以指定：一个圆形、一个长方形、一个三角形。第一行一个水平布局定义圆形，包括：“圆心横坐标 x”、“圆心纵坐标 y”、“圆半径”，最后是一个按钮（“绘圆”）。类似的，第二行一个水平布局定义长方形，第三行一个水平布局定义三角形。每个控制元素由一个 QLabel 部件（显示：“圆心横坐标 x”等）和一个输入框组成。

在远程的服务器的对话框窗口内，每隔 50 个像素绘制一条坐标线，横纵均有。

程序运行时，在客户端的输入框填好相应信息，点击对应的“绘圆”按钮，则在远程的服务器的对话框中以相应的坐标点为圆心，以相应半径值为半径画一个黄色的实心圆。以此类推，点击对应的“绘长方形”或“绘三角形”按钮，则在远程的服务器绘制对应的长方形和三角形。

要求，可以更改设置，实现多次远程绘图。

评分标准（30 分）

（1） 完成客户端布局、服务器端对话框，得 5 分；

（2） 完成客户端和服务器端数据传输，得 15 分，其中圆、长方形、三角形各 5 分；

（3） 完成服务器端的图形绘制，得 5 分。

（4） 支持多次远程绘图，得 5 分。

2、并行计算（console 即可）：将归并排序并行化，要求一主四副共五个线程，使用四个子线程并行计算。

归并排序的基本原理：是将两个排好序的序列组合成一个新的有序序列。归并排序把待排序序列分为若干个子序列，每个子序列是有序的。然后再把有序子序列合并为整体有序序列。

如　设有数列{6，202，100，301，38，8，1}

初始状态：　[6] [202] [100] [301] [38] [8] [1] //将数组分成有序子序列

i=1　[6 202] [100 301] [8 38] [1]　//将相邻的两有序序列组成新的有序序列

i=2　[6 100 202 301] [1 8 38]　//同上

i=3　[1 6 8 38 100 202 301]　//得到最终结果

数据在文件中以每行一个数据单元<key，value>的方式存储；主线程每次把一行读到内存中，并根据 Key % 4 的值计算好其分类，将该数据项放入对应子线程的输入缓冲区中。该输入缓冲区用单向链表结构存储，也就是每个结点的结构体定义为

```
Struct dataNode{
   int key;
   int val;
   struct dataNode * next;
}
```

每个缓冲区用一个指针 first 指向第一个元素。开始时 first 赋值为 NULL。

主线程增加数据项需要修改 first 指针，子线程摘除数据项进行处理也需要修改 first 指针。所以需要用互斥锁保护该输入缓冲区，也就是每次读、写 first 都要先获得互斥锁。

子线程不断的从输入缓冲区中摘取数据项。等子线程发现输入缓冲区为空（first==NULL）时，则休眠 1 秒钟后再去查看输入缓冲区。如果主线程发现数据分发完毕，则给 4 个输入缓冲区分别插入一个 key 为-1 的数据项。子线程发现 key 为-1 的数据项后，便知道所有数据读取完毕，于是开始归并排序。子线程计算结束后，则退出。

主线程分发完数据后，便去等待四个子线程运行的结束。等待四个子线程运行都结束后，主线程负责将最终的四个有序子序列合并成一个有序的序列输出到结果文件。

实现中，要求四个子线程的函数体是同一个，通过参数来区分四者的角色和输出结果。不要求图形界面。

评分标准（30 分）

（1） 将数据集正确读入内存，得 5 分。

（2） 使用互斥锁，正确实现父线程与子线程对输入缓冲区的共享，得 10 分。

（3） 子线程能够正确给出每类数据的结果，得 10 分。

（4） 主线程能够知道四个线程都完成，并在文件中写入最终的正确结果，得 5 分。

3、**简单文件服务器（用到 TCP、UDP、Thread）**：客户端通过 UDP 向文件服务器发送文件下载请求，文件服务器按照请求内容所示通过 TCP 跟该客户端建立连接并发送对应文件。

具体步骤描述如下：

（1）文件服务器通过 UDP 协议在端口 4000 上，等待客户端的文件请求。

（2）客户端通过 GUI 指定下载的文件路径名，并点击“下载”按钮。为了简便，这里假定用户知道文件服务器上的目录信息。用户使用 UDP 协议将该文件下载请求发给文件服务器。请求信息中包括：客户端的 TCP 监听的 IP 地址、端口、文件路径名，具体请求协议格式自己定义。

（3）文件服务器接到请求后，解析该请求，获得：客户端的 TCP 监听的 IP 地址、端口、文件路径名等信息。

（4）服务器根据请求信息中的 IP 地址、端口号，主动跟客户端建立 TCP 连接；打开请求的文件并通过该连接传递给客户端。传输完毕后，服务器关闭该 TCP 连接。

（5）客户端在文件传输过程中，显示下载进度。每个客户端预留五个下载进度条，最多支持同时五个下载任务。每个进度条前，显示下载的文件路径名。接收文件线程将文件保存到本地文件夹中，并且每完成 1%的文件传输就更新一次下载进度。

（6）一个文件服务器，同时支持多个客户端。

注：每个客户端显示五个下载进度条，下方再显示一个下载文件路径指定输入窗口和一个下载按钮。每次新加下载任务，都要检查五个进度条中是否有空闲的，如果没有，则表示已经有 5 个下载任务。文件传输 100%后，进度条回收。

评分标准（40 分）：

（1） 客户端图形界面布局工作正确，得 10 分。

（2） 客户端到服务器，基于 UDP 协议实现请求的发送、解析，得 10 分。

（3） 正确实现服务器到客户端的 TCP 连接、文件传输、进度更新，得 10 分。

（4） 支持同时 5 个文件下载任务，得 5 分。

（5） 实现服务器对多个客户端的并发服务，得 5 分。