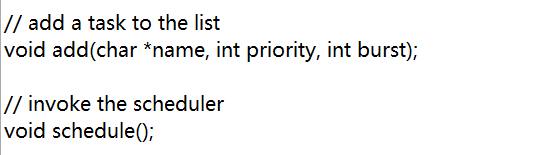
**Project 4: Scheduling Algorithms**

课本中提供了两种可供选择的语言——C或者java。我选择使用C语言来完成该项目。

**（一）问题分析**

这个项目需要实现5种调度算法，需要我补充的函数是schedulers.h中的add()和schedule()。我需要对于每一种调度算法实现这两个函数，用在driver.c的main函数中。



**（二）算法实现**

一、schedule\_fcfs.c

1、实现细节

（1）add()

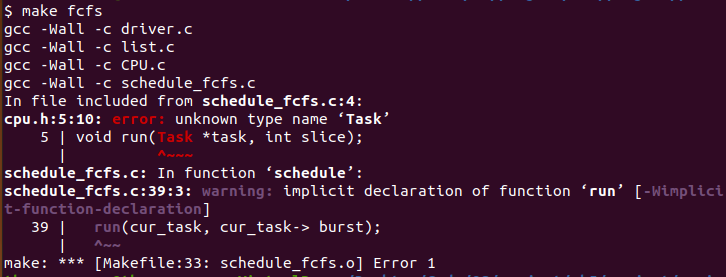
将新建的每个任务结点插入到表头。

（2）schedule()

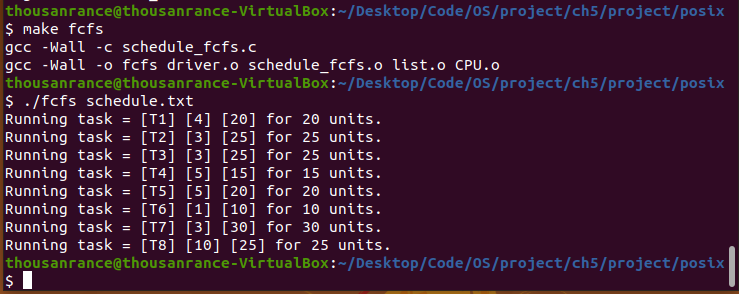
根据先来先服务调度原则，最先来的任务在表的末尾，所以每次调度前都要遍历到表的末尾，调度表尾的任务，最后删除调度过的任务。

2、运行结果

（1）第一次make出现如下错误信息，经检查后发现，cpu.h文件中没有include task.h。



（2）修改后运行正常。



二、schedule\_sjf.c

1、实现细节

（1）add()

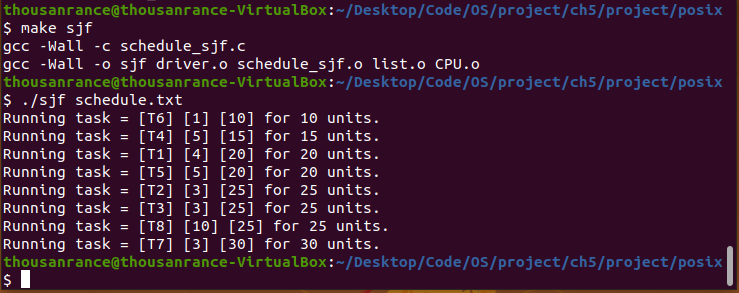
将新建的每个任务结点插入到表头。

（2）schedule()

根据最短服务时间调度原则，每次调度前先都要遍历任务表，找出burst最短的任务，调度该任务，最后删除调度过的任务。

在比较burst时使用“≤”而不是“＜”的意义是：如果burst相等，则按照fcfs原则调度。

2、运行结果



三、schedule\_priority.c

1、实现细节

（1）add()

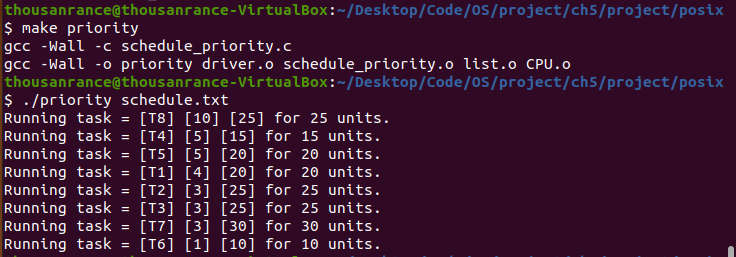
将新建的每个任务结点插入到表头。

（2）schedule()

根据优先级调度原则，每次调度前先都要遍历任务表，找出优先级最高，即priority最大的任务，调度该任务，最后删除调度过的任务。

在比较priority时使用“≥”而不是“＞”的意义是：如果priority相等，则按照fcfs原则调度。

2、运行结果



四、schedule\_rr.c

1、实现细节

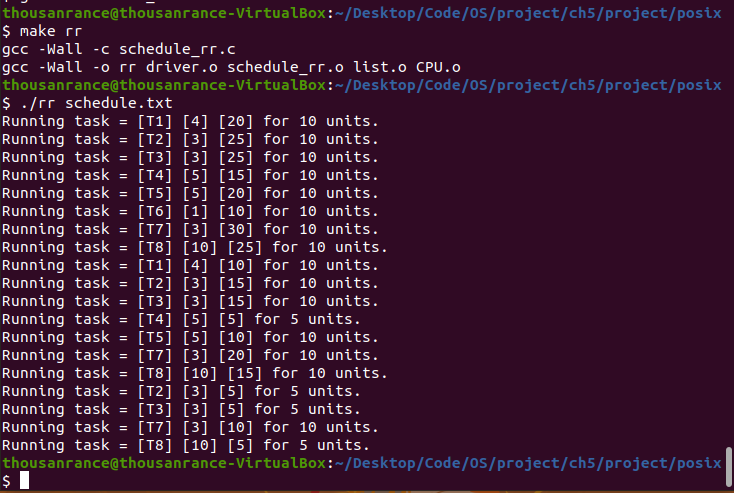
（1）add()

将新建的每个任务结点插入到表头。

（2）schedule()

在一个时间周期内，按照fcfs调度原则选择要调度的任务，如果该任务burst小于一个时间周期，则执行完该任务后删除该任务，直接进入下一个时间周期，选取下一个要调度的任务；如果该任务burst大于一个时间周期，则该任务只执行一个时间周期，一个时间周期结束后将该任务从任务列表尾部删除，将其burst减少一个时间周期，在将其加入到任务列表头部，即视为最后一个到达的任务。

2、运行结果



五、schedule\_priority\_rr.c

1、实现细节

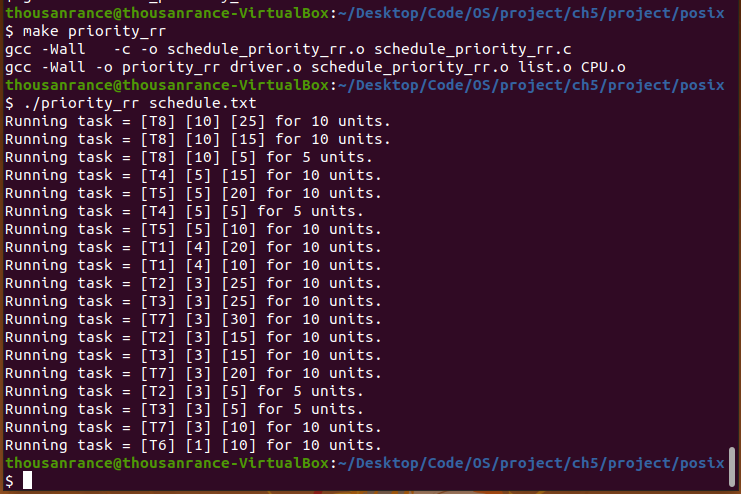
（1）add()

将新建的每个任务结点插入到表头。

（2）schedule()

在一个时间周期内，按照priority调度原则选择要调度的任务，如果该任务burst小于一个时间周期，则执行完该任务后删除该任务，直接进入下一个时间周期，选取下一个要调度的任务；如果该任务burst大于一个时间周期，则该任务只执行一个时间周期，一个时间周期结束后将该任务从任务列表删除，将其burst减少一个时间周期，再将其加入到任务列表头部，即视为最后一个到达的任务。

2、运行结果



**（三）Further Challenges**

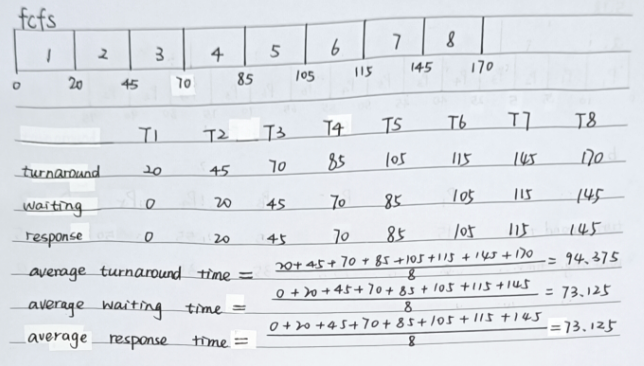
1、Each task provided to the scheduler is assigned a unique task (tid). If a scheduler is running in a SMP environment where each CPU is separately running its own scheduler, there is a possible race condition on the variable that is used to assign task identifiers. Fix this race condition using an atomic integer.·Sol：

已在每个调度算法源文件的schedule()函数中通过\_\_sync\_fetch\_and\_add(&tid\_value)函数实现。

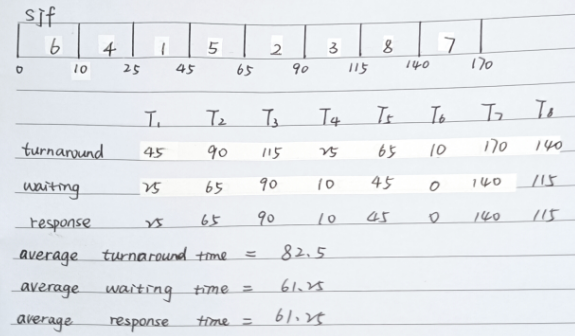
2、Calculate the average turnaround time, waiting time and response time for each of the scheduling algorithms.

·Sol：

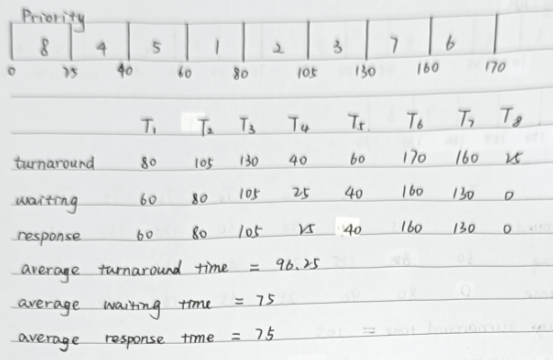
1、FCFS



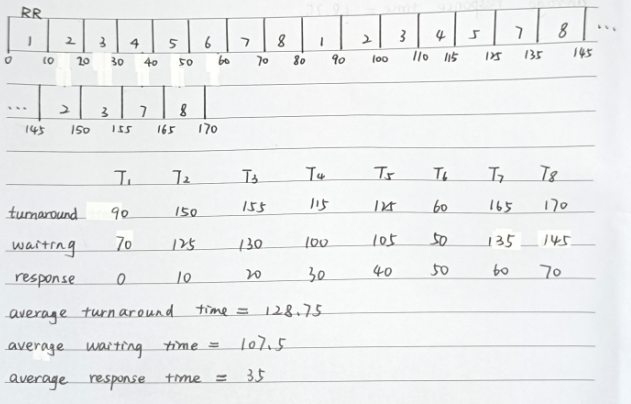
2、SJF



3、Priority



4、RR



5、Priority RR

