# **一、实验目的**：

更改PintOS的线程调度方式，由默认的FCFS（先来先服务）策略改为抢占式优先级调度方式。

# 二、实验原理：

# 1、修改将线程加入就绪队列的方法 list\_push\_back ，将加入队列的方式由直接加入改为依照优先级有序加入。

# 2、需要将线程加入就绪队列的情况有三种，线程从阻塞态恢复（thread\_unblock），线程创建（thread\_create），线程挂起（thread\_yield），利用（1）中修改的加入方式将这三个函数中线程加入就绪队列的方式改为有序加入。

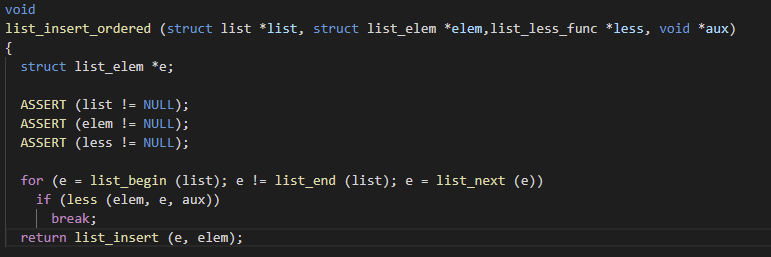
3、thread\_set\_priority函数的作用使改变当前线程的优先级，而当优先级发生改变时影响就绪队列的情况十分常见，此时我们只需直接调用thread\_yield函数便可完成对就绪队列的维护和线程的优先级调度。

4、根据以上原理，维护条件队列和信号量队列为优先级队列，实现线程的优先级调度方式。

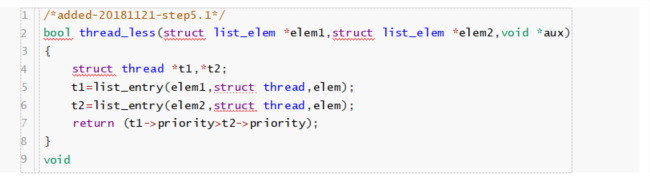
**三、实验方案**

1、修改队列插入方式

为了将线程加入就绪队列的方法改为按照优先级顺序有序加入，我们浏览PintOS中和队列相关的源文件list.c，可以发现两个与队列顺序有关的函数：list\_sort和list\_insert\_ordered函数，其功能分别为将列表排序和按序插入列表，我们在此自然选择使用其后一个插入函数，根据其传入的参数可知我们只需要重新实现一个比较函数 thread\_less即可完成对队列优先级的维护。



2、实现比较函数



定义函数 thread\_less()，在插入就绪队列时确定两个线程的优先级。

3、修改线程从阻塞态恢复（thread\_unblock），线程创建（thread\_create），线程挂起（thread\_yield）三个函数中将线程插入就绪队列的方式



将三个函数中线程插入队列的方式由直接插入改为有序插入。

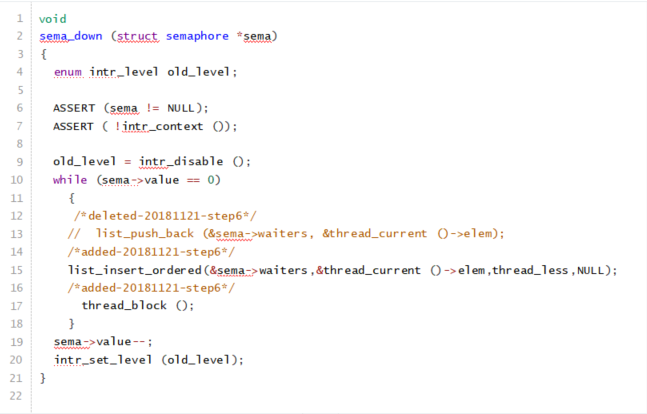
4、在设置线程优先级函数中调用thread\_yield函数，以便当前线程优先级降低后将其挂起，让位给优先级较高的线程，实现抢占式调度。



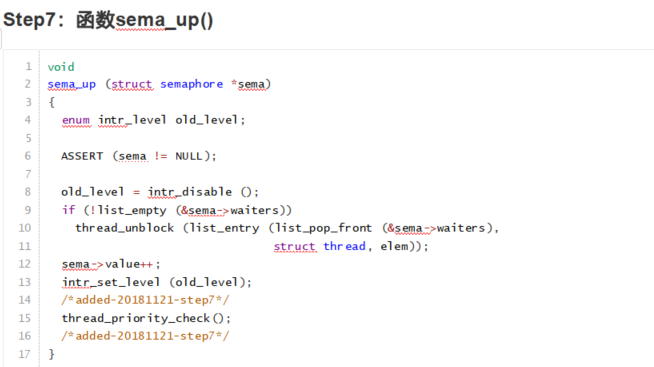
5、实现 thread\_priority\_check() ,在执行v操作时将释放资源后优先级较低的进程挂起。



1. 修改P操作函数sema\_down()，当信号量为0时，将等待资源的线程加入就绪队列的方法由直接加入改为按照线程的优先级顺序加入。



1. 修改V操作函数sema\_up(), 释放资源时检查当前线程的优先级，较低则将其挂起。



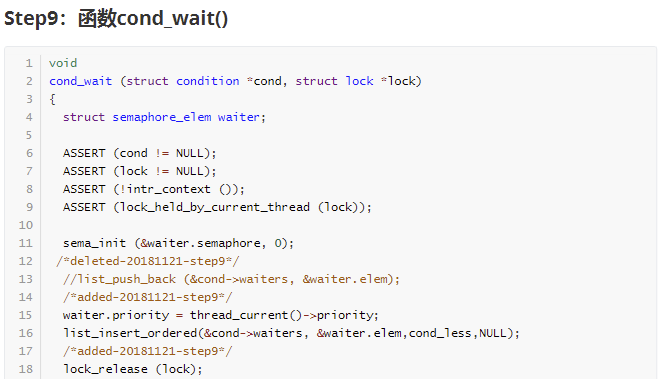
1. 修改信号量结构体semaphore\_elem，加入优先级元素，以便确定条件队列中等待线程的优先级顺序。



1. 定义函数cond\_less() , 用于比较条件队列中两个等待线程优先级的大小，以便维护条件队列。



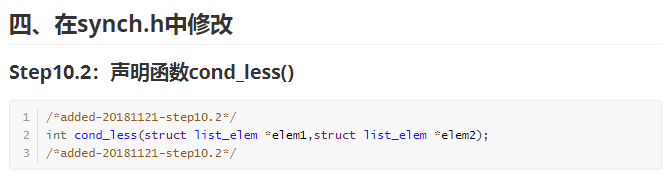
1. 修改cond\_wait() 函数，将维护条件队列的方式改为按照优先级加入队列，确保所需条件完成后，线程的调度方式仍为优先级调度方式。



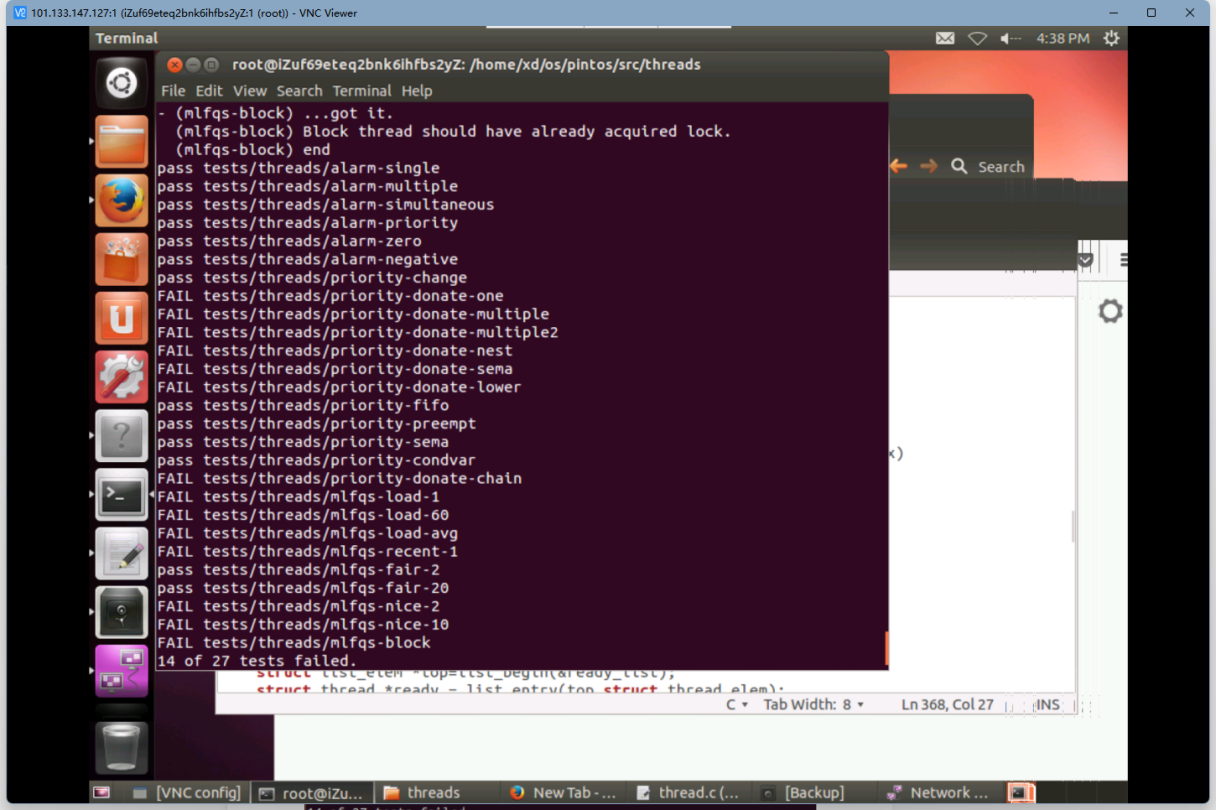


1. 在头文件中声明实验中所实现的函数





1. **实验结果**

****

cd到src/thread下执行make check 指令，编译检查后，有13个测试点通过。