 /*模拟实现LINUX进程调度的静态优先级算法和时间片轮转算法引入LINUX调度 */ #include gt. #include gt. #include gt. #include gt. #define RUN 1 #define SLEEP 0 #define READY 2 #define DEG_SCHEDULE #define NUM 6 struct OSPCB { int PcbName . /*进程名字*/ int ReqCount. /*进程执行计数*/ int RunTime. /*进程执行时间数*/ int Prority. /*进程优先级*/ int PcbStatus. /*进程状态*/ int PcbTime. /*进程时间片*/ struct OSPCB* prev. struct OSPCB *next. }. struct ProcessQueue /*模拟CPU调度队列*/ { struct OSPCB *PointerHead. /*指向进程链表头*/ int PcbNumber. /*CPU每次调度计数器*/ }. //static struct CriticalResource //{ // int flag. // char BufferVoice[2000]. //} static int flag. void *Function(int *arg). void InitPcb(struct OSPCB *pcb). int Schedule(struct ProcessQueue *queue). void InheritSchedule(struct OSPCB *pcb). int main(void) { int i,ret. struct OSPCB *pNewPcb,*pNew. struct ProcessQueue *pNewQueue. int a[4][4] = {{1,1,0,1},{2,2,0,2},{3,3,0,3},{4,4,0,4}}. pNewQueue = (struct ProcessQueue *)malloc(sizeof(struct ProcessQueue)). pNewQueue-gt.PcbNumber = 0. for(i = 0. i gt.PcbName = a[i][0]. pNewPcb-gt.RunTime = a[i][2]. pNewPcb-gt.PcbStatus = READY. pNewPcb-&gt.PcbTime = 3. InitPcb(pNewPcb). 