

**实 验（实训）报 告**

**项 目 名 称**  实验报告2

**所属课程名称**  操作系统

**项 目 类 型**  验证/设计型

**实验(实训)日期**  2024.11.20

**班 级**  软件工程1班

**学 号**  220110750401

**姓 名**  陈亦秦

**指导教师**  陈伟锋

浙江财经大学教务处制

|  |
| --- |
| **一、实验（实训）概述：（简单复述实验要求，1页以内，红字部分应删除）** |
| **【目的及要求】**  **【基本原理】**  **【实施环境】** |
| **二、实验（实训）内容：** |
| **【实验（实训）过程】（步骤、记录、数据、程序等）**  实验5：(gcc使用)  Hello和fork运行      Processes运行    Communication 运行    实验6:  #include<stdio.h>  #include<stdlib.h>  typedef struct PCB {  int ID; //进程标识数  int priority;//优先数  int cpuTime;//已用时间  int allTime;//总共还需时间  int startBlock;//运行多少时间片后阻塞  int blockTime;//在阻塞队列多长时间后进入就绪状态  int state;//进程状态1表示运行，2表示就绪，3表示阻塞，0表示运行完  struct PCB\* next;  }PCB, \* pcb;  pcb p1, p2, lastp;  pcb ReadyQueue, BlockQueue, RunPCB;  pcb accQueue, acclast;//完成队列  void GetPCB();  void Run();  void PushReady(pcb p1);  void ShowQueue(int cputime);  void GetPCB() {  FILE\* fp;  if ((fp = fopen("F://pcb.txt", "r")) == NULL) {  printf("打开文件失败\n");  exit(1);  }  p1 = (pcb)malloc(sizeof(PCB));  while (fscanf(fp, "%d %d %d %d %d %d %d", &p1->ID, &p1->priority,  &p1->cpuTime, &p1->allTime, &p1->startBlock, &p1->blockTime, &p1->state) == 7) {  p1->next = NULL;  if (p1->state == 2) PushReady(p1);  p1 = (pcb)malloc(sizeof(PCB));  }  }  void Run() {  int CupTime = 0;  RunPCB = ReadyQueue;  ReadyQueue = ReadyQueue->next;  ShowQueue(0);  for (; 1; CupTime++) {  //更新值  if (RunPCB) {  RunPCB->cpuTime++;  RunPCB->allTime--;  RunPCB->priority -= 3;//优先数减3  if (RunPCB->startBlock > 0)  RunPCB->startBlock--;  }  p1 = ReadyQueue;  while (p1) {//优先数加1  p1->priority++;  p1 = p1->next;  }  if (RunPCB) {//判断RunPCB的状态  if (RunPCB->allTime == 0) {  acclast->next = RunPCB;  acclast = RunPCB;  acclast->state = 0;  RunPCB = NULL;  }  }//判断RunPCB  if (RunPCB == NULL) {//更新RunPCB  RunPCB = ReadyQueue;  if (ReadyQueue) ReadyQueue = ReadyQueue->next;  }  else if (ReadyQueue && RunPCB->priority < ReadyQueue->priority) {//优先值小了  PushReady(RunPCB);  RunPCB = ReadyQueue;  ReadyQueue = ReadyQueue->next;  }  if (RunPCB) RunPCB->state = 1;  ShowQueue(CupTime + 1);  if (ReadyQueue == BlockQueue && BlockQueue == RunPCB && RunPCB == NULL) break;  }  }  void PushReady(pcb p1) {//以priority降序  p1->state = 2;  lastp = p2 = ReadyQueue;  while (p2 && p2->priority >= p1->priority)  {//查找第一个比p1小的pcb  lastp = p2;  p2 = lastp->next;  }  if (lastp == p2) {//插到队头  p1->next = ReadyQueue;  ReadyQueue = p1;  }  else {//插到队中  p1->next = p2;  lastp->next = p1;  }  //前插队列  }  void ShowQueue(int cputime) {//显示两个队列，和运行的程序  printf("\n----------------------------------------------");  printf("----------------------------------------------\n");  printf("\t当前时间片：%d\n", cputime);  printf("\t%10s %10s %10s %10s %10s %10s %10s\n",  "ID", "priority", "cpuTime", "allTime", "startBlock", "blockTime", " state");  printf("正在运行程序：\n");  p1 = RunPCB;  if (RunPCB)  printf("\t%10d %10d %10d %10d %10d %10d %10d\n",  p1->ID, p1->priority, p1->cpuTime, p1->allTime, p1->startBlock, p1->blockTime, p1->state);  printf("\n就绪队列：\n");  p1 = ReadyQueue;  while (p1) {//优先数加1  printf("\t%10d %10d %10d %10d %10d %10d %10d\n",  p1->ID, p1->priority, p1->cpuTime, p1->allTime, p1->startBlock, p1->blockTime, p1->state);  p1 = p1->next;  }  printf("\n阻塞队列：\n");  p1 = BlockQueue;  while (p1) {//阻塞时间-1  printf("\t%10d %10d %10d %10d %10d %10d %10d\n",  p1->ID, p1->priority, p1->cpuTime, p1->allTime, p1->startBlock, p1->blockTime, p1->state);  p1 = p1->next;  }      **【结论与讨论】（结果、分析）** |
| **三、指导教师评语及成绩：** |
| **评语：**  **成绩： 指导教师签名：**  **批阅日期：** |