**操作系统原理上机实习**

**一、上机实习要求**

1. **上机实习时间为12学时，限定在5月25日-6月13日内自由安排；**
2. **每个学生需完成三道上机实习题。**
3. **每个学生需独立完成上机实习，不得相互抄袭。如发现，抄袭者和被抄袭者成绩均不及格。**
4. **每个学生需提交实习报告（电子档）一份。应包括如下内容:**

**1）实习题目**

**2）程序功能及设计思路**

**3）数据结构及算法设计**

**4）程序运行情况**

**5）编程中遇到的困难及解决方法、实习心得或良好建议**

1. **每个学生需将上机实习报告和源程序打包发给班级学习委员(邮件中请注明学生的班级、学号和姓名)，再由学习委员打包发到老师的邮箱，用于检查是否存在抄袭现象。(邮箱:51782281@qq.com）**

**二、 评分**

**1、上机实习作为操作系统原理课程平时成绩的重要依据。**

**2、抄袭程序者和被抄袭程序者，平时成绩不及格。**

**3、抄袭报告者和被抄袭报告者，平时成绩不及格。**

1. **上机时间地点**

**时间：上机实习时间为12学时，限定在5月25日-6月13日内自由安排；**

**地点：**

1. **上机题目**

**上机实习分三个实验，具体如下：**

|  |
| --- |
| **实验一  多级队列调度算法** |
| **设RQ分为RQ1和RQ2，RQ1采用轮转法，时间片q=7.**  **RQ1>RQ2，RQ2采用短进程优先调度算法。**  **测试数据如下：RQ1: P1-P5, RQ2: P6-P10**   |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | **进程** | **P1** | **P2** | **P3** | **P4** | **P5** | **P6** | **P7** | **P8** | **P9** | **P10** | | **运行时间** | **16** | **11** | **14** | **13** | **15** | **21** | **18** | **10** | **7** | **14** | | **已等待时间** | **6** | **5** | **4** | **3** | **2** | **1** | **2** | **3** | **4** | **5** |   **实现描述：**  **typedef  struct  tag\_pcb**  **{  char  name[8];**  **int need;//须运行的时间**  **int turn;//周转时间**  **struct  tag\_pcb  \*next;**  **} PCB;**  **PCB  \* RQ1,\*RQ2，\*Finish;**  **int clock=0;  //时钟**  **main ( )**  **{  输入RQ1；**  **输入RQ2；（最好从文件读入）**  **while（ＲＱ１！＝ＮＵＬＬ）**  **｛ 从RQ1中选取一进程Pi准备运行；**  **计算其运行的时间t;**  **clock+=t;  //表示Pi运行t;**  **if (Pi完成）  计算其turn;**  **否则   Pi加入到队尾；**  **}**  **while（ＲＱ2!＝ＮＵＬＬ）**  **{ 从RQ2中选取一进程Pi准备运行；**  **clock+=Pi.need;**  **计算Pi的turn;**  **}**  **输出进程的周转时间；**  **}** |
|  |

|  |
| --- |
| **实验二  银行家算法** |
| **#define  n  5  //进程个数**  **#define  m  3  //资源种类**  **int   Available[m],Alloc[n][m],Need[n][m];**  **main()**  **{**  **int request[m];**  **input( );**  **while (1)**  **{**  **read\_req( );**  **if  (请求结束)   break;**  **(1) if (!(requesti<=Needi))   表示非法请求；**  **(2) if (!(requesti<=Available)) 则Pi阻塞；**   1. **试探性分配**   **Available=Available - Requesti;**  **Alloci=Alloci+Requesti;**  **Needi=Needi-Requesti;**  **(4)若新状态安全，则实际分配资源给Pi,否则取消试探性分配。**  **}**  **}**  **安全状态判别算法：**  **(1)设置Finish=(false,...,false) work=Available**  **(2)循环查找满足下列条件的进程pi //最多循环n次**  **Finish[i]=false且Needi<=work**  **(3)若找到则Finish[i]=true;work=work+Alloci; 转(2)**  **(4)若Finish=(true,...,true) 则安全，否则不安全。**  **测试数据：m=3:种类型的资源（A,B,C,） 进程个数n=5**  **Available=(2,3,3);**   |  |  |  | | --- | --- | --- | |  | **已分配资源数量** | **资源需求量** | | **A B C** | **A B C** | | **P1** | **2 1 2** | **3 4 7** | | **P2** | **4 0 2** | **1 3 4** | | **P3** | **3 0 5** | **0 0 3** | | **P4** | **2 0 4** | **2 2 1** | | **P5** | **3 1 4** | **1 1 0** |   **请求序列如下：**  **a: 进程P2请求资源（0，3，4）**  **b 进程P4请求资源（1，0，1）**   1. **进程P1请求资源（2，0，1）**   **d.进程P3请求资源（0，0，2）** |

|  |
| --- |
| **实验三    动态分区式存贮区管理** |
| **设计一个动态分区式存贮区管理程序，要求支持不同的放置策略。如首次、最佳、最坏。**  **说明：**  **（1）分区描述器rd如下:**   |  |  |  | | --- | --- | --- | | **flag** | **size** | **next** |   **要求空闲区队列按链表组织。**  **主存大小假设为maxsize（单位为节=rd的大小）。**  **（2）主程序结构如下：**  **输入放置策略**  **申请一块内存作为主存**  **循环处理用户的请求（包括申请、释放）**  **需设计两个函数处理用户请求：**  **申请函数 Addr=Request(size)**  **释放函数 Release(addr）**  **（3）数据实例：maxsize=512**  **J1申请162，J2申请64，J3申请120，J4申请86，J1完成，J3完成，J5申请72，J6申请100，J2完成，J7申请36，J8申请60，J4完成，J9申请110，J10申请42。备注：**  **（a）所有大小其单位为节（1节=rd的大小）**  **（b）作业申请n节,实际分配的分区大小应为n+1节。 其中一节作为分区描述器，其他n节提供给作业。**  **（c）已分配区放在高地址处。** |
|  |
| |  |  |  | | --- | --- | --- | | **0** | **512** | **∧** | |  |  |  |  |  |  |  | | --- | --- | --- | | **0** | **349** | **∧** | |  |  |  | | **1** | **163** | **∧** | |  |  |  |     0  **→**  349  350  **开始时 J1申请162节后 返回地址350**  **（d）合并时应考虑四种情况： 假设回收区为r,上邻为f1（f1需搜索自由主存队列）,下邻为f2（f2可直接计算）**  **A)f1空闲，f2已分配；**  **B)f1已分配，f2空闲；**  **C)f1空闲，f2空闲；**  **D)f1已分配，f2已分配；** |
|  |