**实验一 处理器管理-优先数调度算法**

**一、实验目的**

在多道程序或者多任务系统中，同时处于就绪态的进程有若干个。在单处理器系统中，处于运行态的进程最多只有一个，也就是说能运行的进程数远小于就绪态进程个数，所以必须进行进程调度，按照一定的调度算法，即依照某种原则或策略从就绪态队列中选中进程占用处理器。本实验要求学生设计一个优先数调度算法来模拟实现处理器调度过程。

**二、实验要求**

（1）假定进程控制块中包含内容为：

|  |
| --- |
| 进程名 |
| 链接指针 |
| 进程的优先数 |
| 估计运行时间 |
| 进程状态 |

其中进程名即进程标示符；链接指针是按照进程到达进程就绪队列的时间链接

进程的优先数由用户自己指定，并且优先数值越小，优先级越高，调度时，总是选择优先级高的进程占用CPU；估计运行时间可由设计者任意指定一个时间值；

（2）根据需要，设计一个队首指针指向5个进程(进程名为A,B,C,D,E)组成的就绪队列中的第一个进程，再设一个当前指针指向当前正在运行的进程。

（3）进程调度时，采用静态优先调度。

（4）用非抢占式方式分配处理器。

（5）要求所设计的程序必须有输出语句，用于输出进程调度和运行一次后进程的变化。比如：每次被选中的进程名，就绪队列中进程的排队情况，进程控制块的动态变化。

一、基本实现：

1、数据结构

Struct{

进程名

链接指针

进程的优先数 （设计好优先数的次序： 例如5个优先级 分别是0-4 假定0最高，4最低）

估计运行时间

进程状态

}

2、变量

运行进程的指针

就绪队列的指针

估计时间

3、程序：

 main（）

{

初始化指针

输入若干进程

（对每次输入的进程依据优先数排到就绪队列） （临时变量：遍历指针）

运行调度程序（函数）！

处理系统结束后回收相关资源！

}

调度（指向运行进程的指针，就绪队列指针）

{

while（就绪队列指针不为空）

{摘就绪队列头节点；

用运行进程指针指向；

取估计时间=》估计时间

输出正在运行的进程！

Timer(估计时间)

进程运行结束；

Free（指向运行进程的节点指针）

}

}

二、新增阻塞（等待）态：

1、数据结构

Struct{

进程名

链接指针

进程的优先数 （设计好优先数的次序： 例如5个优先级 分别是0-4 假定0最高，4最低）

估计运行时间

输入输出时间链指针

进程状态

}

struct{

输入输出类型

输入输出开始时间

输入输出时间

下一个输入输出节点指针}

2、变量

运行进程的指针

就绪队列的指针

等待队列的指针

估计时间

3、程序：

 main（）{

初始化指针

输入若干进程

（对每次输入的进程依据优先数排到就绪队列） （临时变量：遍历指针）

运行调度程序（函数）！

处理系统结束后回收相关资源！

}

调度（指向运行进程的指针，就绪队列指针，等待队列指针）

{ while（就绪队列指针不为空）

{摘就绪队列头节点；

用运行进程指针指向；

输出正在运行的进程！

取输入输出信息，确定运行估计时间=》估计时间

判断（是否执行结束）

{Timer(估计时间)

进程运行结束；

Free（指向运行进程的节点指针）

}

else{

调用**调度用等待处理函数（等待队列指针，正在运行的进程指针（要切换为等待态））**}

}}

**调度用等待处理函数（等待队列指针，正在运行的进程指针（要切换为等待态））**

**{**

**输出切换进入等待队列的进程**

**将当前进程放入等待队列**

**摘下当前输入输出节点，获得输入输出时间=》等待时间**

**timer（等待时间）**

**等待事件结束，将该进程切换放入就绪队列**

**}**

**实验二 处理器管理-银行家算法**

**一、实验目的**

本实验要求学生采用银行家算法有效避免资源分配中的死锁发生。用银行家算法设计一个程序，判断当前资源分配状态是不是安全状态；如果是安全状态，当系统接收新的资源申请时，系统是否发生死锁，如发生，请找出参与死锁的进程。如果不发生死锁，则找出一组能够顺利推进的进程顺序。

**二、实验要求**

设计3个并发进程，共享系统的12个同类不可抢占的资源。各个进程动态进行资源的申请和释放。把各个进程对资源的使用情况记录在进程控制块中（如下表格式所示）。

|  |
| --- |
| 进程名 |
| 进程的状态 |
| 当前申请量 |
| 资源需求总数 |
| 已占用资源数 |
| 能否执行完标志 |

PCB

假定这组进程当前使用资源情况如下表所示，要求编程判断当前是否安全状态，并输出该结果。如果是安全状态，则从键盘任意输入三个进程的资源申请数，再次判断是否处于安全状态？如果是安全状态，请给出并输出无死锁发生的一组进程推进的顺序。

假定资源当前分配状态如下：

|  |  |  |
| --- | --- | --- |
| 进程 | 已占资源数 | 最大需求数 |
| P1 | 2 | 9 |
| P2 | 5 | 10 |
| P3 | 2 | 4 |

**实验三 作业调度-响应比高优先调度算法**

**一、实验目的**

作业调度算法是指依照某种原则或策略从后备作业队列中选取作业的方法。响应比最高者优先算法是通过计算输入井后备队列中每个作业的响应比大小，从中选择响应比最高的作业装入主存，这样既考虑了作业的等待时间，又考虑了作业的运行时间。本实验要求学生设计一个响应比最高者优先算法来模拟实现作业调度过程。

**二、实验要求**

假设本系统仍采用单道批处理系统，忽略设备工作时间和系统进行调度所花的时间。要求从键盘输入作业个数N，及每个作业的作业名、作业入井时间、估计运行时间。请编程输出采用响应比最高者优先算法得到的每个作业调度序号、作业名、作业入井时间、开始调度时间、运行时间、结束时间、周转时间， 以及所有作业的平均周转时间。

**实验四 存储管理-**

**可变分区存储管理的空间分配与去配**

**一、实验目的**

要求掌握存储管理中的典型算法，理解各种存储管理的工作原理，特别是可变分区存储管理中最先适应分配算法、最优适应分配算法、最坏适应分配算法以及空间回收算法的工作原理，试采用C语言编程，模拟实现算法功能。

**二、实验要求**

设计一个可变式分区分配的存储管理方案，并模拟实现分区的分配和回收过程。

假定主存空间为静态分配。为实现分区的分配和回收，需要已分配分区表和系统空闲分区表描述当前系统状态。已分配分区表包括分区编号、已分配分区长度、分区的起始地址等信息。系统空闲分区表包括分区编号、分区长度、分区的起始地址等信息。用户根据需要提出对主存空间大小的申请，系统按照一定的分配策略，找出能满足请求的空闲区进行分配。如果满足要求，输出分配完成后已分配分区表和空闲区表的信息，否则输出“无空闲区分配”。

用户根据需要释放主存空间，实现空间的回收，并输出空间回收后已分配分区表和空闲区表的信息。

**实验五 存储管理-**

**请求分页式存储管理的页面置换**

**一、实验目的**

要求了解虚拟存储技术的特点，理解并掌握请求页式存储管理的页面置换算法，试采用C语言编程，模拟实现请求页式存储管理中页面置换算法。

**二、实验要求**

在分页式虚拟存储管理中，要求通过键盘输入分配给一个作业的物理块数和作业依次访问的10个页面号，采用先进先出（FIFO）页面置换后，顺序输出缺页中断时所淘汰的页面号，并计算缺页中断率。

**实验六 设备管理-电梯调度算法**

**一、实验目的**

任何一个对磁盘的访问请求，应给出访问磁盘的存储空间地址：柱面号、磁头号和扇区号。在启动磁盘执行I/O操作时，应先把移动臂移动到指定的柱面，再等待指定的扇区旋转到磁头位置下，最后让指定的磁头进行读/写，完成信息传送。移臂调度是根据访问者指定的柱面位置来决定执行次序的调度。本实验要求学生设计一个电梯调度算法来模拟实现磁盘移臂调度过程。

**二、实验要求**

假设要求从系统中输入N个需访问的柱面号，当前磁头的移动方向由键盘输入(1代表磁头从外往内移动，-1代表磁头由内往外移动)，当前磁头刚完成访问序号为M的柱面，请编程输出采用电梯调度算法得到的柱面访问序列号，同时输出读/写磁头总共移动的距离(用柱面数表示)