**操作系统实验指导书**

**华中农业大学信息学院计算机科学系**

**目录**

实验一 进程调度 3

实验二 死锁 7

实验三 存储管理 11

实验四 文件系统 14

附录 实验报告参考规范 18

**实验1 进程调度**

**一、目的与要求**

**1. 目的**

进程调度是处理及管理的核心内容，本次实验要求用C语言编写和调试一个简单的进程调度程序。调度算法可以任意选择或自行设计，例如简单轮转法和优先数法等。通过本次实验可以加深各使用进程控制块进行进程调度和各种调度算法的理解及其实施方法。

**2. 要求**

1. 在微型计算机上设计进程控制块（PCB）结构，使其分别适用于简单轮转法和优先数调度算法。PCB通常包括以下信息：进程名、进程优先数（或轮转时间片）、进程所占用的CPU时间、进程的当前状态、当前队列指针等。根据调度算法的不同，PCB结构的内容可以作适当的增删。
2. 调度程序应包含2~3种不同的调度算法，运行时可任选一种，以利于各种算法的分析比较。基本要求是：优先数调度算法和简单循环轮转法。
3. 建立进程就绪队列，对各种不同的算法编制入链子程序，同时应具有显示或打印各进程的运行状态和参数的变化情况，便于观察各进程的调度过程。
4. 本次实验的上机时间为4学时。

**二、实验内容**

* 1. **题目**

采用C语言编写程序，选用优先数调度算法或简单轮转法对五个进程进行调度，每个进程处于运行（Run）、就绪（Ready）和完成（Finish）三种状态之一，并假定起始状态为就绪状态。

* 1. **进程控制块结构**

PCB 进程标识符

连接指针

优先数 | 轮转时间片

占用的CPU时间片数

进程所需时间片数

进程状态

进程控制块的链结构

Run Ready Tail

1

0

……

^

……

……

……

^

Finish

……

……

……

^

其中：Run——表是当前运行进程指针

Ready——就绪队列头指针

Tail——就绪队列尾指针

Finish——完成队列指针

* 1. **算法说明及程序框图**

1. 为了便于处理，程序中进程的运行时间以时间片为单位进行计算，各进程的优先数或轮转时间片数，以及进程需要运行的时间片数，其初始值均有用户给定。
2. 优先数法

进程就绪队列按优先数大小从高到低排列，链首进程首先投入运行。进程每执行一次，进程需要的时间片数减1、该进程的优先数减3。这样，该进程如果在一个时间片中没有完成，其优先数降低一级。接着仍是用该进程降低一级后的优先数与就绪队列中链首进程的优先数进行比较，如果仍是该进程的优先数高或相同，便让该进程继续执行；否则，调度就绪队列的链首进程投入运行。原运行过的进程按其现行优先数大小插入就绪队列，且改变它们对应的进程状态，一直到所有进程都运行完各自的时间片数。

1. 简单轮转法

进程就绪队列按各进程进入的先后顺序排列。进程每次所需处理机的轮转式按其重要程度记入进程控制块中的轮转时间片数记录项。进程执行时，每运行一个时间片，进程还需要的时间片数减1，运行进程占用处理机的时间片数加1，然后比较占用CPU的时间片数是否与该进程的轮转时间片数相等，若相等则说明已达到轮转时间，应将现运行的进程排列就绪队列的末尾，调度队列上的首进程运行，且改变它们的进程状态，直至所有进程完成各自的时间片。

1. 程序框图
   1. 优先数调度

开始

生成并按优先数大小排列进程控制块

链首进程投入运行

时间片到，进程时间片数减1，优先数减3

进程时间片数为0？

撤销该进程

从链首取一个进程投入运行

优先数大于链首进程？

运行进程推出，按优先数插入进程链

进程队列空？

结束

是

否

否

是

图1.1 优先数调度算法流程图

* 1. 时间片轮转调度

开始

生成并按进入次序排列进程控制块链

链首进程投入运行

时间片到，进程时间片数减1，占用CPU时间加1

进程时间片数为0？

撤销该进程

从链首取一个进程投入运行

占用CPU时间已到轮转时间？

运行进程推出，排到运行链尾部

否

否

是 是

进程队列空？

结束

否

是

图1.2 时间片轮转调度算法流程图

**实验2 死锁**

**一、目的与要求**

**1. 目的**

学生应独立的采用高级语言编写一个动态分配系统资源的程序，模拟死锁现象，观察死锁发生的条件，并采用适当的算法，有效地防止死锁的发生。学生应通过本次实验，更直观的了解死锁发生的原因，初步掌握防止死锁、解除死锁的简单方法，加深理解教材中有关死锁的内容。

**2. 要求**

1. 设计一个几个并发进程共享m个系统资源的系统。进程可动态的申请资源和释

放资源，系统按各进程的申请动态的分配资源。

1. 系统应能显示各进程申请和释放资源的过程，能显示系统动态分配资源的过程，便于观察和分析。
2. 系统应能选择是否采用防止死锁的算法，应设计多种防止死锁的算法，并能任意选择其中任何一种投入运行，同时要求，在不采用防止死锁算法时观察死锁现象发生的过程，在使用防止死锁的算法时，了解在同样条件下防止死锁发生的过程。
3. 本次实验的上机时间为2～4学时。
4. 所有程序均应采用C语言编写程序，特别需要的部分可采用汇编语言便程序。

**二、实验内容**

1. **题目**

本次实验采用银行算法防止死锁的发生。设有3个并发进程共享10个系统资源。在3个进程申请系统资源之和不超过10时，当然不可能发生死锁，因为各个进程资源都能满足。在有一个进程申请的系统资源数超过10时，必然会发生死锁。应该排队这两种情况。程序采用人工输入各进程的申请资源序列。如果随机给各进程分配资源，就可能发生死锁，这也就是不采用防止死锁算法的情况。假如，按照一定的规则，为各进程分配资源，就可以防止死锁的发生。示例采用银行算法。这是一种犹如“瞎子爬山”的方法，即探索一步，前进一步，行不通，再往其它方向试探，直至爬上山顶。这种方法是比较保守的，所花的代价也不小。

1. **算法与框图**

*银行算法*，顾名思义是来源于银行的借贷业务，一定数量的本金要应付各种客户的借贷周转，为了防止银行因资金无法周转而倒闭，对每一笔贷款，必须考察其最后是否能归还。研究死锁现象时就碰到类似的问题，有限资源为多个进程共享，分配不好就会发生每个进程都无法继续下去的死锁僵局。

银行算法的原理是先假定某一次分配成立，然后检查由于这次分配是否会引起死锁，即剩下的资源是不是能满足任一进程完成的需要。如这次分配是安全的（不会引起死锁），就实施本次分配，再作另一种分配试探，一直探索到各进程均满足各自的资源请求，防止了死锁的发生。

程序框图见图2.1，图2.2和图2.3。

输出出错信息

找出一个未完成的

进程作为现行进程

现行进程

已完成所有

请求

收回该进程占用的资源，标志该进程结束，如果全部进程结束，置ADVANCE为‘flase’，否则’true’

找出一个未完成的进程作为现行进程

申请量

超过最大需

求量

死锁处理程序

进程已全部结束？

开始

输入选择标志

OPTION

输入各进程动态申请

资源数组AP

输入各进程最资源需求

量数组VPMAXCLAIM

最大资源

需求量超过系统

资源数

输出出错信息

关闭输出文件

结束

是

否

否 是

否

是

否

是

图2.1 银行家算法流程1

剩余资源

能否满足现行进程？

请求

开始

当前分配

状态安全吗？

（safe）

承认已假定分配给现行进程的资源输出信息6，置ADVANCE=true

采用死锁预防算法？

收回已假定分配给现行进程的资源，输出信息4，置ADVANCE=true

承认已假定分配给现行进程的资源，输出信息5，置ADVANCE=true

结束

收回已假定分配给现行进程的资源输出信息3

如不采用死锁预防算法，将现行进程死锁位置1

采用死锁预防算法，如已有进程死锁位置ADVANCE=false

否

是

是

否

是

否

图2.2 银行家算法流程2

置未完成进程的T标志为1

置已完成进程的T标志为0

PROGRESS是否为“ture”？

开始

置局部变量PROGRESS为“ture”

（即是否有进程标志位从1变0）

判断是否还存在不能问成的进程，若不存在，置SAFE 为“ture”，否则为“false”

结束

置PROGRESS为 “false”

全部进程已经完成？（即各进程T标志均为0）

现行进程所占资源与剩余资源之和是否超过该资源的最大申请量？

现行进程能够完成，把其所占资源加到剩余资源中去，置‘0’标志T（表示该进程能完成）置PROGRESS为 ‘true’

否

是

否

是

否

是

图2.3 银行家算法流程3

**实验3 存储管理**

**一、目的与要求**

**1. 目的**

提高内存管理的效率始终是操作系统研究的重要课题之一，虚拟存储技术是用来提高存储容量的一种重要方法，所以，本项实验的目的是让学生独立地设计几个常用的存储分配算法，并用高级语言编写程序对各种算法进行分析比较，评测其性能的优劣，从而加深对这些算法的了解。

**2. 要求**

本实验要求学生用C语言独立编写分区分配算法、回收算法、请求式分页分配算法。在请求式分页分配算法中，通过程序的执行结果来分析计算不同页面淘汰算法情况下的访问命中率，并以此来比较各种算法的优劣，同时，还要求分析改变页面大小和实际存储容量对计算结果的影响，为选择好的算法，合适的页面尺寸和实存容量提供依据。

本次实验的上机时间为2～4学时。

**二、实验内容**

**1. 分区分配算法**

本实验要求采用首次适应算法和最佳适应算法两种分区分配的内存管理算法。

(1) 建立分区描述器：分区描述器可根据自己编写程序的需要来建立，描述器本身所包含的内容以描述清楚内存分区情况为准。

(2) 建立自由主存队列：针对两种不同的放置策略（首次适应和最佳适应）来建立相应的队列结构。

(3) 用C语言编写实现首次适应算法和最佳适应算法的程序。

(4) 用C语言编写回收算法。

**2. 请求式分页存储管理算法**

本实验要求采用请求式分页存储算法，淘汰算法采用先进先出算法FIFO和最近最少使用页面淘汰算法（LRU）。

设逻辑空间大小为128K，页面尺寸分别为2、4、6、8、10、12、14、16K，内存容量为8至64页。

1. 先进先出算法FIFO：该算法的实质是选择作业中在主存驻留时间最长的一页淘汰，这种算法容易实现，例如分配一个作业的存储块数为m，则只需建立一张m个元素的队列表Q(0)、Q(1)、…、Q(m-1)和一个替换指针。这个队列是按页调入主存的一页。如图4-1所示，某时刻调入主存四个块，（即m=4），它们按页进入主存的先后顺序为4、5、1、2，当需要置换时，总是淘汰替换指针所指向的那一页，见图3.1.

（指向最老的一页）

页 号

2

4

5

1

替换指针

图3.1 队列表的实现示意图

新调进的页面装入主存后，修改相应的队列元素，然后将替换指针往前移动，使其指向当前最老的一页。

(2) 最近最少使用页面淘汰算法（LRU）：这是一种经常使用的方法，有多种不同的实施方案。这里采用不断调整页表链的方法，即总是淘汰页表链链首的页面，而把新访问的页面插入链尾。如果当前调用页面已在页表内，则把它再次调整到链尾。这样就能保证最近使用的页面总处于靠近链尾部分，而不常使用的页面被移到链首，逐个被淘汰。

(3) 程序流程图如图3.2所示

开 始

产生给定长度符合某种假定的指令地址流

Alg=FIFO/LRU?

为每一指令地址形

成对应的访问页号

置初值：size=1

assigned=4

输入淘汰算法Alg

用FIFO淘汰算法计算命中率

最近最少使用页面淘汰算法计算命中率

输出完整结果

结 束

图3.2 请求式分页存储算法流程图

**实验4 文件系统**

**一、目的与要求**

**1. 目的**

文件系统是操作系统的一个重要组成部分，也是与用户关系极为密切的部分。学生应独立的用高级语言编写和调试一个简单的文件系统，模拟文件管理的工作过程，从而对各种文件操作命令的实质内容和执行过程有比较深入的了解，掌握它们的实施方法，加深对教材中有关内容的理解。

**2. 要求**

1. 设计一个n个用户的文件系统，每个用户最多可保存m个文件。
2. 限制用户在一次运行中只能打开一个文件。
3. 系统应能检查输入命令的正确性，出错时要能显示出错原因。
4. 对文件必须设置保护措施，如只能执行，允许读，允许写等。在每次打开文件时，根据本次打开的要求，再次设置保护级别，即可有二级保护。
5. 对文件的操作至少应有下面几条命令：

create 建立文件

delete 删除文件

open 打开文件

close 关闭文件

read 读文件

write 写文件

1. 本次实验的上机时间为2~4学时。

**二、实验内容**

**1. 题目**

1. 本次实验设计一个共有10个用户的文件系统，每个用户最多可保存10个文件，一次运行过程中用户可同时打开5个文件。
2. 程序采用二级文件目录，即设置了主文件目录（MFD）和用户文件目录（UFD）。前者应包含文件主（即用户）及他们的目录区指针；后者应给出每个文件主占有的文件目录，即文件名、保护码、文件长度以及它们存放的位置等。另外为打开文件设置了运行文件目录（AFD），在文件打开时应填入打开文件号，本次打开保护码和读写指针等。
3. 为了便于实现，对文件的读写作了简化，在执行读写命令时，只修改读写指针，并不进行实际文件的读写操作。

**2. 算法与框图**

1. 因系统小，文件目录的检索使用了简单的线性搜索，而没有采用Hash等有效算法。
2. 文件保护简单使用了三位保护码，对应于允许读，允许写和允许执行，如下所示：

1 1 1

允许读 允许写 允许执行

如对应位为0，则不允许。

1. 程序中使用的主要数据结构如下：
   1. 主文件目录（MFD）和用户文件目录（UFD）

MFD UFD

用户名1

文件目录指针1

用户名2

文件目录指针2

用户名3

文件目录指针3

┋

┋

文件名

┋

文件长度

保护码

文件名

┋

文件长度

保护码

图4.1主文件目录（MFD）和用户文件目录（UFD）

② 打开文件目录（AFD）

打开文件号

打开保护码

读写指针

图4.2 打开文件目录（AFD）

1. 程序框图4.3所示：

输入用户名

开始

在MFD中找到该用户？

报告无此用户文件

“try again”

否

是

结束

显示该用户目录表UFD中所有文件

初始化运行

文件表AFD

保存当前文件目录

Say good bye

打印当前文件目录

输入操作命令

是什么命令？

建立文件处理程序

删除文件处理程序

打开文件处理程序

建立文件处理程序

关闭文件处理程序

建立文件处理程序

create delete open bye close read write

图4.3 文件系统各个函数算法实现流程图

**三、参考书目**

[1] 郭浩志主编，计算机软件实践教程，西北电讯工程学院出版社，1995年（第二版）

[2] 庞丽萍，李胜利编，操作系统原理，华中科技大学出版社，2000年（第三版）

[3] 汤小丹编著，计算机操作系统，西安电子科技大学出版社（第四版）

# 

# 附录 实验报告参考规范

使用学院统一的实验报告封面并正确给出课程名称、课程号、专业、班级、学好、姓名、指导教师和完成日期。

报告内容及格式如下：  
**1．实验目的**

给出本实验要求达到的目的。  
**2．实验内容**

给出本实验要求完成的实验任务。  
**3 实验步骤**

1. 任务分析：以无歧义的陈述说明实验任务，强调的是要做什么？并明确规定：  
   　　(1) 输入的形式和输入值的范围；  
   　　(2) 输出的形式；  
   　　(3) 程序所能达到的功能；  
   　　(4) 测试数据：包括正确的输入及其输出结果和含有错误的输入及其输出结果。
2. 概要设计：说明本程序中用到的所有抽象数据类型的定义、主程序的流程以及各程序模块之间的层次(调用)关系。
3. 详细设计　　实现概要设计中定义的所有数据类型，对每个操作只需要写出伪码算法；对主程序和其他模块也都需要写出伪码算法(伪码算法达到的详细程度建议为：按照伪码算法可以在计算机键盘直接输入高级程序设计语言程序)；画出函数和过程的调用关系图。
4. 调试分析：  
   　　a．调试过程中遇到的问题是如何解决的以及对设计与实现的回顾讨论和分析；  
   　　b．算法的时空分析(包括基本操作和其他算法的时间复杂度和空间复杂度的分析)和  
   　　 改进设想；  
   　　c．经验和体会等。
5. 测试结果：列出你的测试结果，包括输入和输出。这里的测试数据应该完整和严格，最好多于需求分析中所列。
6. 使用说明：说明如何使用你编写的程序，详细列出每一步的操作步骤。

**4**．**实验总结**

**5．附录**

带注释的程序清单。