#### 页表的定义和作用

在分页系统中，允许将进程的各个页离散地存储在内存的任一物理块中，为保证进程仍然能够正确地运行，即能在内存中找到每个页面所对应的物理块，系统又为每个进程建立了一张页面映像表，简称页表。

页表的作用是实现从页号到物理块号的地址映射。

#### 分段存储管理方式和分页的区别和联系

(分段以段为单位，各段可以离散存放，但是每一段要连续，分页是以页为单位)

相似点：

采用离散分配方式，通过地址映射机构实现地址变换

不同点：

页是信息的物理单位，分页是为了满足系统的需要；段是信息的逻辑单位，含有一组意义相对完整的信息，分段是为了满足用户的需要。

页的大小固定且由系统确定，由系统把逻辑地址分为页号和页内地址，由机器硬件实现；段的长度不固定，取决于用户程序，编译程序对源程序编译时根据信息的性质划分。

分页的作业地址空间是一维的；分段的作业地址空间是二维的。

#### 内存管理方式的内容

1 一次装入

1. 连续分配

单一连续分配、固定分区分配、动态分区分配、动态可重定位分区分配

1. 离散分配

(基本)分页存储管理方式

(基本)分段存储管理方式

短页式存储管理方式

2 虚拟内存

#### 什么是动态分区分配

1. 原理：

动态分区分配是根据进程的实际需要，动态地为之分配内存空间。作业装入内存时，把可用内存分出一个连续区域给作业，且分区的大小正好适合作业大小的需要。分区的大小和个数依装入作业的需要而定。

2. 实现

在实现过程中涉及如下问题：

分区分配中的数据结构

分区分配算法

分区分配及回收操作

1） 分区分配中的数据结构

（1）空闲分区表示

空闲分区表：记录每个空闲分区的情况。每个空闲分区占一个表目。表目中包括：分区序号、分区始址、分区的大小、等。

空闲分区链：在每个分区的起始部分，设置一些用于控制分区分配的信息，以及用于链接各分区所用的前向指针；在分区尾部则设置一后向指针，在分区末尾重复设置状态位和分区大小表目。

（2）已占分区说明表

结构：作业号；起始地址；大小

2）分区分配算法

为把一个新作业装入内存，需按照一定的分配算法，从空闲分区表或空闲分区链中选出一分区分配给该作业。

常用的分配算法：

(1) 首次适应算法FF

(2) 循环首次适应算法

(3) 最佳适应算法

#### 重定位

多道作业装入内存，逻辑地址与实际的内存物理地址不同

#### 虚拟存储器为何引入、定义、特征、实现方法

引入： (作业太大，作业太多，假象🡪扩大)计算机系统中内存的大小是有限的，当用户的作用地址空间大于内存可用空间时，用户使用计算机就不太方便了。为了给其他地址空间超过内存可用空间的大作业得用户提供方便，使他们摆脱对内存和外存的分配和管理，由操作系统把多级存储器统一管理起来，实现对内存的自动“扩大”。给用户造成一个假象：就是系统中有一个很大的内存供用户使用，实际上这个“很大的内存”是不存在的，是虚的（当然它有一定的物质基础来支持），所以引入虚拟存储器的概念。

定义：虚拟存储器是指，具有请求调入功能和置换功能，在逻辑上对内存容量加以扩充的一种存储器系统。

特征：多次性、对换性、虚拟性。

实现方法：

分页请求系统

硬件：

请求分页的页表机制

缺页中断机构

地址变换机构

软件：

实现请求调页的软件和实现页面置换的软件。

请求分段系统

硬件：

请求分段的页表机制

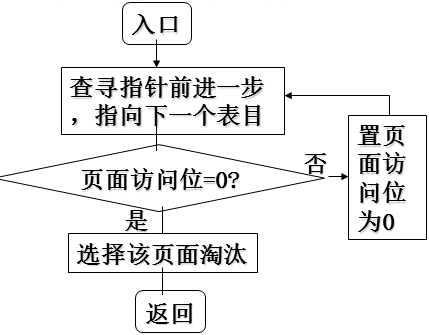
缺段中断机构

地址变换机构

软件：

实现请求调段的软件和实现断置换的软件。

#### Clock算法（最近未使用）





#### 请求分段存储管理方式的定义

以分段为单位进行换入、换出。

硬件支持：

请求段表机制

段名

段长

段基址 指示本段在内存中的起始地址

存取方式：由于应用程序中的段是信息的逻辑单位，可根据该信息的属性对它实施保护，故在短表中增加存取方式字段，如果该字段为两位，则存取属性是只执行、只读和允许读/写。

访问字段A：访问频繁程度

修改位M: 进入内存后是否已被修改过

存在位P: 指示该段是否已调入内存

增补位: 分段中特有的。是否做过动态增长

外存始址： 指示本段在外存中的起始地址

缺段中断机构

地址变换机构

页号 块号 物理地址对应的是块号加块内偏移

#### 系统抖动的原因

置换算法选择不当

#### OS定义

一组能有效地组织和管理计算机硬件和软件资源，合理地对各类作业进行调度，以及方便用户使用的程序的集合。

#### OS特性

并发性（最基本） 是指两个或多个事件在同一时间间隔内发生

共享性（最基本） 是指系统中的资源可供内存中多个并发执行的进程共同使用 互斥的共享方式和同时访问方式

虚拟性 通过某种技术奖一个物理实体变为若干个逻辑上的对应物的功能

异步性 各个进程以不可预知的速度执行

#### 设备分配的过程（P203约定）（分配设备、分配设备控制器、分配通道）

分配步骤：

1 设备分配

2 分配控制器

3 分配通道

#### 设备分配时应考虑的因素

1 设备的固有属性 (独占设备的分配策略/ 共享设备的分配策略/ 虚拟设备的分配策略)

2 设备分配算法（先来先服务/ 优先级高者优先）

3 设备分配中的安全性

#### 死锁 定义 产生的原因 必要条件 处理方法

死锁是指多个进程因竞争资源而造成的一种僵局，若无外力作用，这些进程都将无法向前推进。

原因：竞争可消耗性资源、竞争不可抢占性资源、进程推进顺序不合法。

必要条件：互斥条件、请求和保持、不可抢占、循环等待。

处理方法：预防（破坏死锁的3个必要条件，除互斥外）

避免（银行家算法：目的：避免死锁，工作过程、安全性的判断P110）

检测（资源分配图的简化）

解除（终止进程等等）

#### SPOOLING技术 定义，分成哪几个部分、特点

SPOOLING技术是一种虚拟设备技术，是在联机情况下实现的同时外围操作的技术。它可以把一台设备改造为虚拟设备，在进程所需的物理设备不存在或被占用的情况下，使用该设备。SPOOLING技术是对脱机输入、输出系统的模拟，所以又称为假脱机操作。

SPOOLING系统主要由四部分组成：输入井和输出井（这是在磁盘上开辟出来的两个存储区域。输入井模拟脱机输入时的磁盘，用于收容I/O设备输入的数据。输出井模拟脱机输出时的磁盘，用于收容用户程序的输出数据。输入输出井中的数据一般以文件的形式组织管理(井文件)。一个文件仅存放某一个进程的输入（或者输出）数据，所有进程的数据输入（或输出）文件链接成一个输入（或输出）队列）、输入缓冲区和输出缓冲区（用于缓和CPU和磁盘之间速度不匹配的矛盾。输入缓冲区暂存由输入设备传送的数据，之后再传送到输入井。输出缓冲区用于暂存从输出井传送的数据，之后再传送到输出设备）、输入进程和输出进程（内存、井、设备、缓冲区）、井管理程序（用于控制作业与磁盘井之间信息的交换）。

特点：

提高了I/O的速度

将独占设备改造为共享设备

实现了虚拟设备功能

#### 四种I/O控制方式

1.使用轮询的可编程I/O方式

2.使用中断的可编程I/O方式

3.直接存储器访问方式DMA

4.I/O通道控制方式----CPU干预最少的方式

#### 文件存储空间的管理（位示图、成组链接法）描述，如何分配、回收

空闲表法和空闲链表法

位示图法

用0，1表示是否分配

二维数组map[m,n] –> 位示图

分配：

1 顺序扫描位示图，从中找出一个或一组其值为”0”的二进制位（“0”表示空闲时）

2 将所找到的一个或一组二进制位转换成与之对应的盘块号。b=列数n\*（i-1）+j

3 修改位示图Map[i,j]=1

回收：

1 将回收盘块的盘块号转换成位示图中的行号和列号。转换公式为：

i=(b-1)DIV n+1

j=(b-1)MOD n+1

2 修改位示图 Map[i,j]=0

成组链接法

分配：

当系统要为用户分配文件所需的盘块时，须调用盘块分配过程来完成。

该过程首先检查空闲盘块号栈是否上锁，如未上锁，便从栈顶取出一空闲盘块号，将与之对应的盘块分配给用户，然后将栈顶指针下移一格。

若该盘块已是栈底，即S.free(0), 这是当前栈中最后一个可分配的盘块号。由于在该盘块号所对应的盘块中记有下一组可用的盘块号，因此，须调用磁盘读过程将栈底盘块号所对应盘块的内容读入栈中，作为新的盘块号栈的内容，并把原栈底对应的盘块分配出去(其中的有用数据已读入栈中)。然后，再分配一相应的缓冲区（作为该盘块的缓冲区）。最后，把栈中的空闲盘块数减1并返回。

回收：

在系统回收空闲盘块时，须调用盘块回收过程进行回收。它是将回收盘块的盘块号记入空闲盘块号栈的顶部，并执行空闲盘块数加1操作。当栈中空闲盘块号数目已达100时，表示栈已满，便将现有栈中的100个盘块号记入新回收的盘块中，再将其盘块号作为新栈底。

#### 空闲表、空闲链表 与内存动态分区分配相似

#### 引入进程的原因

为了描述程序在计算机内的执行情况

#### I/O定义

在该系统中包括有用于实现信息输入、输出和存储功能的设备和相应的设备控制器，在有的大中型机中，还有I/O通道或I/O处理机。

（通道是硬件 将控制通道的程序放在主存）

#### I/O分类

设备按其使用特性分类：存储设备、输入＼输出设备

按其共享属性分类：独占、共享、虚拟

按信息交换的单位分类：块设备、字符设备

按传输速率分类：低速、中速、高速

#### 引入缓冲的原因

（为了解决速度差问题）

引入缓冲区的主要原因归结为以下几点：

缓和CPU与I/O设备间速度不匹配的矛盾。

减少对CPU的中断频率，放宽对CPU中断响应时间的限制

提高CPU和I/O设备之间的并行性。

#### 文件目录的类型

单级目录（不允许两个文件重名）

两级目录

多级目录（树形目录）

#### 文件的外存分配方式

连续分配

链接分配

索引分配

#### 目录管理的要求

实现按名存取--最基本功能，也是引用文件系统的主要目的

提高对目录的检索速度

文件共享

允许文件重名

#### 文件的逻辑结构

有结构文件（记录式文件）

无结构文件（流式文件）

#### 外存的组织方式

连续组织方式

链接组织方式

索引组织方式

#### 进程的同步分为什么

进程同步

互斥：临界区需要互斥的访问

同步：为完成同一任务相互合作

#### 磁盘调度算法的原理和计算 根据磁道访问序列 计算平均寻道长度 并评价算法

先来先服务：

公平、简单，没有长期不满足。

寻道未优化，平均长，适用请求磁盘IO进程少的。

最短寻道时间优先：

老进程饥饿

不能保证平均最短

SCAN:

考虑距离和磁头移动方向

寻道性能好，防止饥饿，广泛用于大中小型机和网络中磁盘调度。

刚移动过某一磁道，恰有进程请求访问， 扫描完后才处理，严重推迟

CSCAN:

减少请求进程的延迟

N-Step-SCAN算法：

将磁盘请求队列分成若干个长度为N的子队列

按FCFS算法依次处理

FSCAN算法：

只将磁盘请求访问队列分成两个子队列

当前所有请求磁盘I/O的进程 按SCAN算法进行处理

扫描期间，新出现 新请求都将被推迟到下一次扫描时处理

SCAN算法不仅考虑欲访问的磁道与当前磁道的距离，更优先考虑磁头的当前移动方向。在磁头正在自里向外移动时，所选择的下一个访问对象应是其欲访问的磁道既在当前磁道之外，又是距离最近的。这样自里向外地访问，直至再无更外的磁道需要访问时，才将磁臂换向，自外向里移动。这时，每次选择要访问的磁道，在当前磁道之内且距离最近者这样的进程来调度。

SCAN算法中磁头移动的规律似电梯的运行，又称为电梯调度算法。算法既能获得较好的寻道性能，又能防止进程饥饿，被广泛用于大、中、小型机和网络中的磁盘调度。

存在的问题：当磁头刚从里向外移动过某一磁道时，恰有一进程请求访问此磁道，这时该进程必须等待，待磁头从里向外，然后再从外向里扫描完所有要访问的磁道后，才处理该进程的请求，致使该进程的请求被严重地推迟。

**先来先服务（FCFS）**

根据进程请求访问磁盘的先后次序进行调度。优点：公平、简单，且每个进程的请求都能依次得到处理，不会出现某一进程的请求长期得不到满足的情况。

缺点：未对寻道进行优化，致使平均寻道时间可能较长。仅适用于请求磁盘I/O的进程数目较少的场合。

**最短寻道时间优先（SSTF）**

算法选择要求访问的磁道与当前磁头所在的磁道距离最近的进程，以使每次的寻道时问最短。

存在的问题：可能导致某些进程发生“饥饿”。因为只要不断有所要访问的磁道与磁头当前所在磁道的距离较近的新进程到达，就会出现“老进程饥饿”现象。这种调度算法不能保证平均寻道时间最短。

循环扫描 CSCAN

为了减少请求进程的延迟，CSCAN算法规定磁头单向移动。若规定只自里向外移动，当磁头移到最外的被访问磁道时，磁头立即返回到最里的欲访磁道，即将最小磁道号紧接着最大磁道号构成循环，进行扫描。

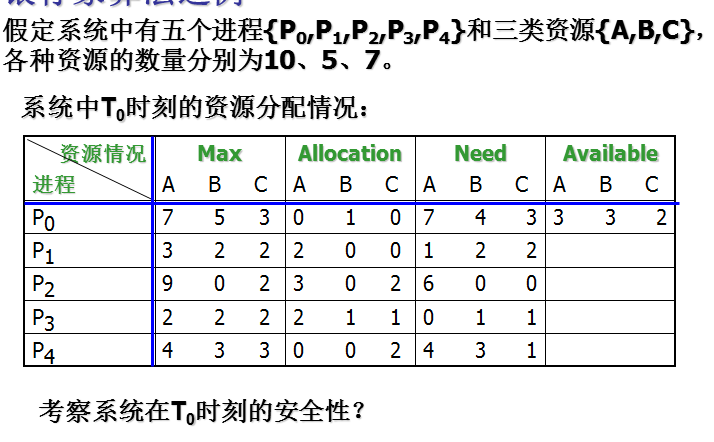
采用循环扫描方式后，上述请求进程的请求延迟，将从原来的2T减为T+Smax，其中，T为由里向外（或相反）扫描完所有要访问的磁道所需的寻道时间，而Smax是将磁头从最外面被访问的磁道直接移到最里边欲访问的磁道所需的寻道时间。

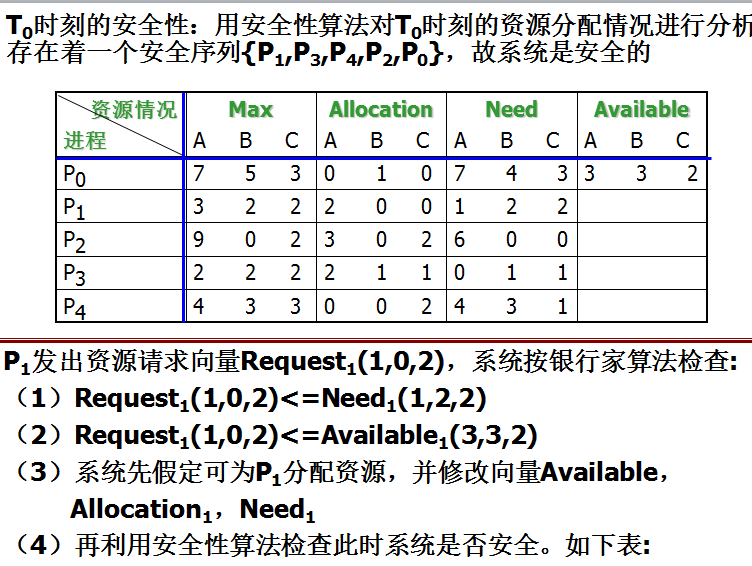
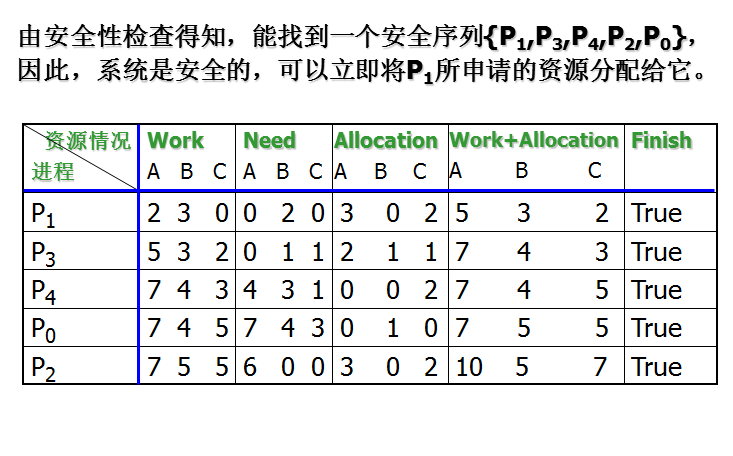
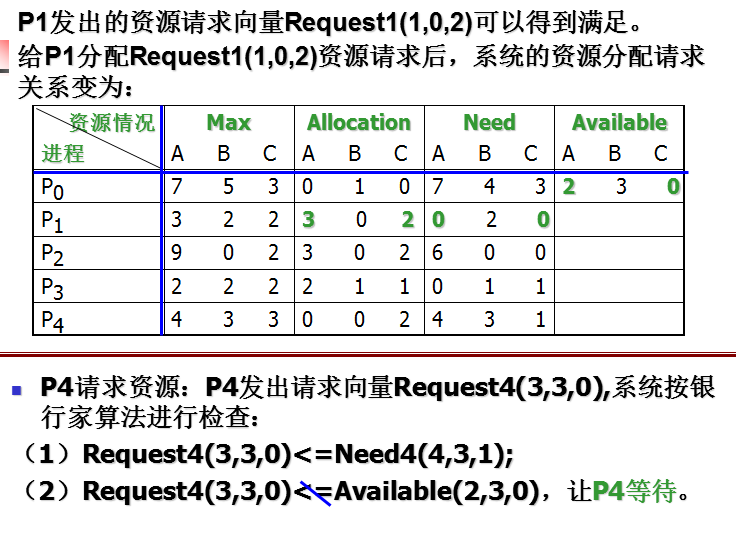
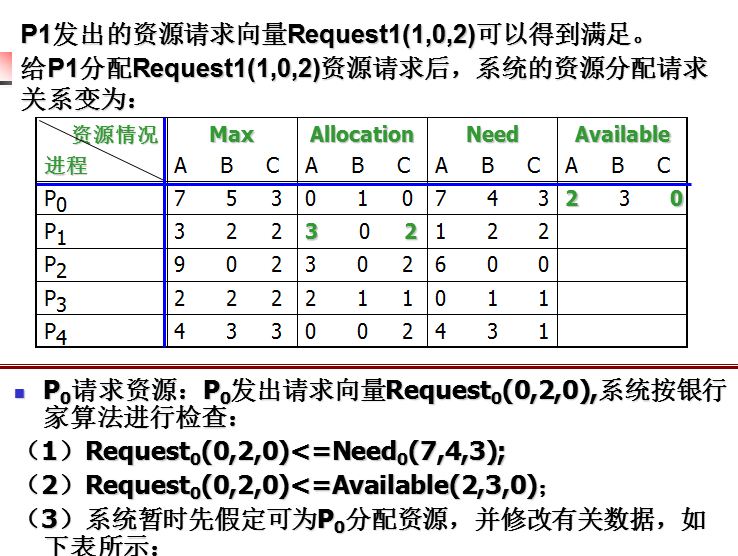
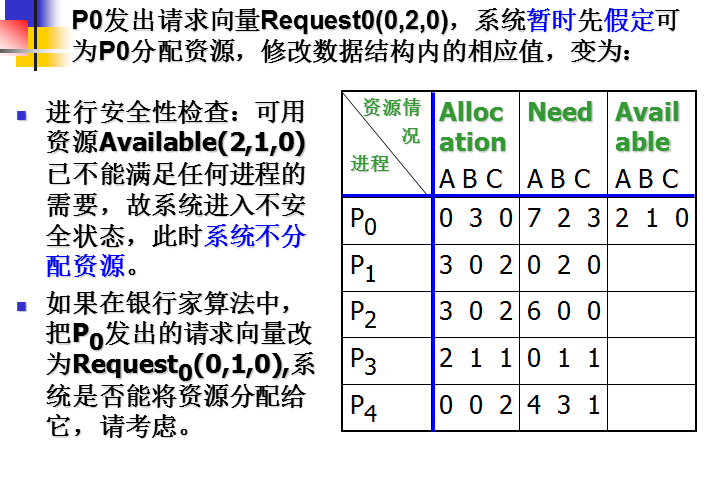
N-Step-SCAN算法：SSTF、SCAN、CSCAN几种调度算法都可能出现磁臂停留在某处不动的情况，称为磁臂粘着。在高密度盘上更容易出现此情况。N-Step-SCAN算法将磁盘请求队列分成若干个长度为N的子队列。磁盘调度将按FCFS算法依次处理这些子队列，而每处理一个队列时，又是按SCAN算法。这样就可避免出现粘着现象。N值取得很大时，其性能接近SCAN算法；N=1时，则退化为FCFS算法。

FSCAN算法：本算法是N-Step-SCAN算法的简化。它只将磁盘请求访问队列分成两个子队列。 一是当前所有请求磁盘I/O的进程形成的队列，由磁盘调度按SCAN算法进行处理；另一个则是在扫描期间，新出现的所有请求磁盘I/O进程组成的等待处理的请求队列。从而使所有的新请求都将被推迟到下一次扫描时处理。

大题预测：

#### 银行家算法



#### 平均周转时间

1、在一个批处理系统中，采用非抢占式**短作业优先**的作业调度算法当一个作业进入系统后就可以开始调度，假定作业都仅是计算，忽略调度花费的时间。现有4个作业，进入系统的时间和需要计算的时间如下表所示：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **作业** | **进入系统时间** | **需要计算时间** | **开始时间** | **完成时间** | **周转时间** |
| **1** | 10.0 | 2.0 | 10.0 | 12.0 | 2.0 |
| **2** | 10.2 | 1.0 | 12.8 | 13.8 | 3.6 |
| **3** | 10.4 | 0.5 | 12.3 | 12.8 | 2.4 |
| **4** | 10.5 | 0.3 | 12.0 | 12.3 | 1.8 |

* 1. 试计算出每个作业的开始时间、完成时间及周转时间并填入表中。
  2. 计算4个作业的平均周转时间和平均带权周转时间为多少。（11分）

(2)平均周转时间：(2+3.6+2.4+1.8)/4=2.45

（1+3.6+4.8+6）/4=3.85

#### 缺页率

1. 在一个请求分页存储管理系统中，假定系统分配给一个作业的物理块数为3，并且此作业的页面走向为2、3、2、1、5、2、4、5、3、2、5、2。试用FIFO（先进先出淘汰算法）和LRU（最近最久未使用淘汰算法）两种算法分别计算出程序访问过程中所发生的缺页次数及缺页率（假设开始执行时主存中没有页面）。（12分）
2. 根据所给页面走向，使用先进先出页面淘汰算法时，页面置换情况如下：

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **走向** | **2** | **3** | **2** | **1** | **5** | **2** | **4** | **5** | **3** | **2** | **5** | **2** |
| **块1** | 2 | 2 |  | 2 | 5 | 5 | 5 |  | 3 |  | 3 | 3 |
| **块2** |  | 3 |  | 3 | 3 | 2 | 2 |  | 2 |  | 5 | 5 |
| **块3** |  |  |  | 1 | 1 | 1 | 4 |  | 4 |  | 4 | 2 |
| **缺页** | 缺 | 缺 |  | 缺 | 缺 | 缺 | 缺 |  | 缺 |  | 缺 | 缺 |

物理块数为3时，采用先进先出淘汰算法，缺页数是，9**（3分）**；缺页率=9/12 **（3分）**

（2）根据所给页面走向，使用最近最久未使用页面淘汰算法时，页面置换情况如下：

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **走向** | **2** | **3** | **2** | **1** | **5** | **2** | **4** | **5** | **3** | **2** | **5** | **2** |
| **块1** | 2 | 2 |  | 2 | 2 |  | 2 |  | 3 | 3 |  |  |
| **块2** |  | 3 |  | 3 | 5 |  | 5 |  | 5 | 5 |  |  |
| **块3** |  |  |  | 1 | 1 |  | 4 |  | 4 | 2 |  |  |
| **缺页** | 缺 | 缺 |  | 缺 | 缺 |  | 缺 |  | 缺 | 缺 |  |  |

物理块数为3时，采用最近最久未使用淘汰算法，缺页数是：7**（3分）**；缺页率=7/12**（3分）**

1．在一个请求分页系统中，采用LRU页面置换算法，例如一个作页的页面走向为4，3，2，1，4，3，5，4，3，2，1，5，当分配给该作业的物理块数M分别为3和4时，试计算访问过程中所发生的缺页次数和缺页率？（注明：有内存块最初都是空的），并比较所得结果。

解：（1）当M=3时，

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 页面走向 | 4 | 3 | 2 | 1 | 4 | 3 | 5 | 4 | 3 | 2 | 1 | 5 |
| 缺页标记 | \* | \* | \* | \* | \* | \* | \* |  |  | \* | \* | \* |
| M1 | 4 | 4 | 4 | 1 | 1 | 1 | 5 | 5 | 5 | 2 | 2 | 2 |
| M2 |  | 3 | 3 | 3 | 4 | 4 | 4 | 4 | 4 | 4 | 1 | 1 |
| M3 |  |  | 2 | 2 | 2 | 3 | 3 | 3 | 3 | 3 | 3 | 5 |

缺页次数=10 缺页率=缺页次数/总页数\*100%=10/12\*100%=83.3%

（2）当M=4时

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 页面走向 | 4 | 3 | 2 | 1 | 4 | 3 | 5 | 4 | 3 | 2 | 1 | 5 |
| 缺页标记 | \* | \* | \* | \* |  |  | \* |  |  | \* | \* | \* |
| M1 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 5 |
| M2 |  | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 |
| M3 |  |  | 2 | 2 | 2 | 2 | 5 | 5 | 5 | 5 | 1 | 1 |
| M4 |  |  |  | 1 | 1 | 1 | 1 | 1 | 1 | 2 | 2 | 2 |

缺页次数=8 缺页率=8/12\*100%=67%

#### PV操作

3、 有三个进程PA、PB和PC合作解决文件打印问题：PA将文件记录从磁盘读入主存的缓冲区1，每执行一次读一个记录；PB将缓冲区1的内容复制到缓冲区2，每执行一次复制一个记录；PC将缓冲区2的内容打印出来，每执行一次打印一个记录。缓冲区1和2都只能存放一个记录。请用P、V操作来保证文件的正确打印。（12分）

Var empty1，empty2，full1，full2：semphore：=1，1，0，0；

//empty1：缓冲区1是否为空，初值为1；

//empyt2：缓冲区2是否为空，初值为1；

//full1：缓冲区1是否有记录可供处理，初值为0；

//full2：缓冲区2是否有记录可供处理，初值为0；

PA

{

从键盘读一个记录；

P(empty1)；

将记录存入缓冲区1；

V(full1)；

}

PB

{

P(full1)；

从缓冲区1中取出记录；

V(empty1)；

P(empty2)；

将记录存入缓冲区2；

V(full2)；

}

PC

{

P(full2)；

从缓冲区2中取出记录；

V(empty2)；

打印记录；

}

2. 假定一个阅览室可供50个人同时阅读。读者进入和离开阅览室时都必须在阅览室入口处的一个登记表上登记，阅览室有50个座位，规定每次只允许一个人登记或注销登记。要求：（1）用PV操作描述读者进程的实现算法（可用流程图表示，登记、注销可用自然语言描述）；

（2）指出算法中所用信号量的名称、作用及初值。

解

S1:阅览室可供使用的空座位，其初值为50

S: 是否可通过阅览室，其初值为1

Process READ\_in（i=1…50）

{到达阅览室入口处;

P(S1);P(S);

在入口处登记座位号;

V(s);

进入座位并阅读;

}

Process READ\_out（j=1…50）

{结束阅读到达阅览室入口处;

P(S);

在入口处注销座位号;

V(S1);V(S)

离开入口处;

#### 磁盘调度

3、假设一个可移动磁头的磁盘具有200个磁道，其编号为0~199，当前它刚刚结束了125道的存取，正在处理149道的服务请求，假设系统当前磁盘请求序列为：88, 147, 95, 177, 94, 150, 102, 175, 138。试问对以下的磁盘调度算法而言，满足以上请求序列，磁头将如何移动？并计算总的磁道移动数。

（1）先来先服务策略

（2）最短寻道时间优先策略

（3）扫描策略

(1)FCFS算法： 5分

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 当前149 | 下一磁道 | 88 | 147 | 95 | 177 | 94 | 150 | 102 | 175 | 138 |
|  | 移动距离 | 61 | 59 | 52 | 82 | 83 | 56 | 48 | 73 | 37 |

总的磁道移动数为：61+59+52+82+83+56+48+73+37=551

(2)SSTF算法： 5分

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 当前  149 | 下一磁  道 | 150 | 147 | 138 | 102 | 95 | 94 | 88 | 175 | 177 |
|  | 移动距离 | 1 | 3 | 9 | 36 | 7 | 1 | 6 | 87 | 2 |

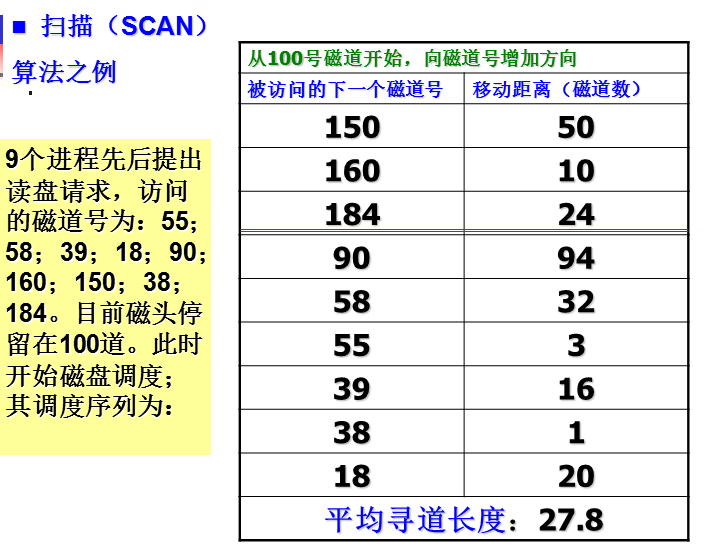
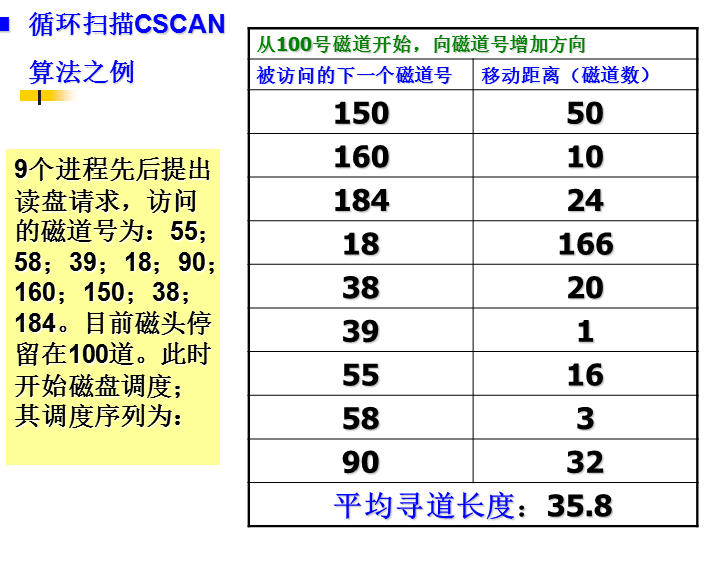
总的磁道移动数为：1+3+9+36+7+1+6+87+2=152

(3)SCAN算法： 5分

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 当前  149 | 下一磁道  (按增加磁  道数方向) | 150 | 175 | 177 | 147 | 138 | 102 | 95 | 94 | 88 |
|  | 移动距  离 | 1 | 25 | 2 | 30 | 9 | 36 | 7 | 1 | 6 |

总的磁道移动数为：1+25+2+30+9+36+7+1+6=117



#### 求物理地址

1. 设页面大小为1KB，将逻辑地址3BADH划分为页号和页内偏移量两部分。用16进制表示。

例2. 设页面大小为2KB，将逻辑地址3BADH划分为页号和页内偏移量两部分。用16进制表示。

