第四章 存储器管理

一、教学目的要求：

1．掌握存储管理的几项基本功能

2．掌握虚拟存储器的概念

3．掌握地址变换的概念和实现

4．掌握内存分配和回收的主要功能

5．掌握信息保护的几种方法

6．掌握分区管理的实现及特点

7．掌握覆盖和交换技术的实现

二、内容分析：

1．概述：本部分内容从整体上讲存储管理所要实现的功能，从分区到页式、段式、段页式的存储管理在内存利用率、内存扩充、内存共享与保护等方面的优化。

2．教学重点：

1）存储管理的基本功能

2）内存扩充的几种方式

3）分区存储管理的实现

4）页式存储管理的实现

5）段式存储管理的实现

6）段页式存储管理的实现

3．教学难点：

1）分区分配算法

2）地址映射

存储管理的基本功能

1．内存的分配和回收

内存分配按分配时机的不同，可分为两种方式。(1）静态存储分配：指内存分配是在作业运行之前各目标模块连接后，把整个作业一次性全部装入内存，并在作业的整个运行过程中，不允许作业再申请其他内存，或在内存中移动位置。也就是说，内存分配是在作业运行前一次性完成的。(2）动态存储分配：作业要求的基本内存空间是在目标模块装入内存时分配的，但在作业运行过程中，允许作业申请附加的内存空间，或是在内存中移动，即分配工作可以在作业运行前及运行过程中逐步完成。

2．地址重定位

（1） 内存空间（或物理空间）

内存是由若干个存储单元组成的，每个存储单元有一个编号，这种编号可唯一标识一个存储单元，称为内存地址（或物理地址）。

（2）逻辑空间

源程序经过汇编或编译后，形成目标程序，每个目标程序都是以 0 为基址顺序进行编址的，原来用符号名访问的单元用具体的数据——单元号取代。这样生成的目标程序占据一定的地址空间，称为作业

的逻辑地址空间，简称逻辑空间。在逻辑空间中每条指令的地址和指令中要访问的操作数地址统称为逻辑地址。

	名空间	0b	逻辑地址空间	1000b	物理空间
		100b		1100b	
	Mov R1,[data]		Mov R1,[200]		Mov R1,[200]
		200b		1200b	
data:	6817		6817		6817
		299b		1299b	

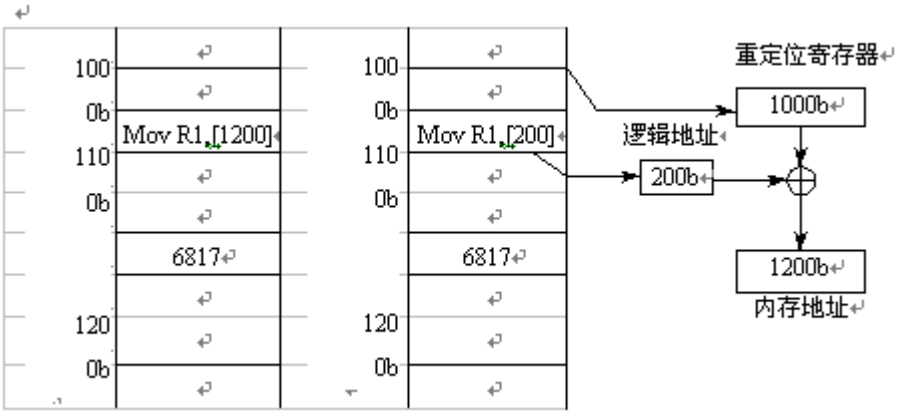
（3）地址重定位

① 静态地址重定位

静态地址重定位是在程序执行之前由操作系统的重定位装入程序完成的。

② 动态地址重定位

动态地址重定位是在程序执行期间进行的。

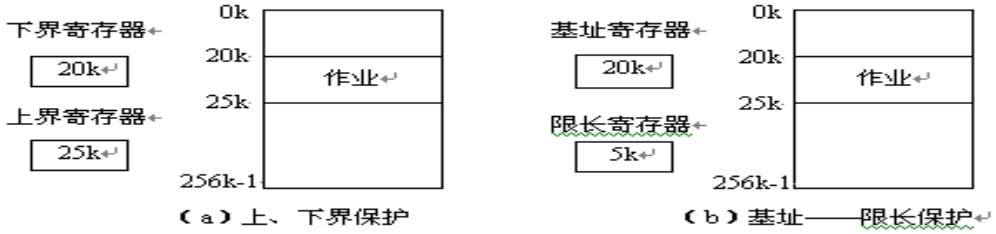


(a) 静态地址重定位　　(b) 动态地址重定位

3．存储保护

（1）上、下界存储保护：上、下界保护是一种简单的存储保护技术。系统可为每个作业设置一对上、下界寄存器，分别用来存放当前运行作业在内存空间的上、下边界地址，用它们来限制用户程序的活动范围。

（2）基址—限长存储保护：上、下界保护的一个变种是采用基址—限长存储保护。



（a）上、下界保护　　（b）基址——限长保护

4．虚拟存储器

对内存进行逻辑上的扩充，现在普遍采用虚拟存储管理技术。

虚拟存储技术的基本思想是把有限的内存空间与大容量的外存统一管理起来，构成一个远大于实际内存

的、虚拟的存储器。此时，外存是作为内存的直接延伸，用户并不会感觉到内、外存的区别，即把两级存储器当作一级存储器来看待。一个作业运行时，其全部信息装入虚存，实际上可能只有当前运行的必需一部分信息存入内存，其他则存于外存，当所访问的信息不在内存时，系统自动将其从外存调入内存。

5．内外存数据传输的控制

（1）覆盖

（2）交换

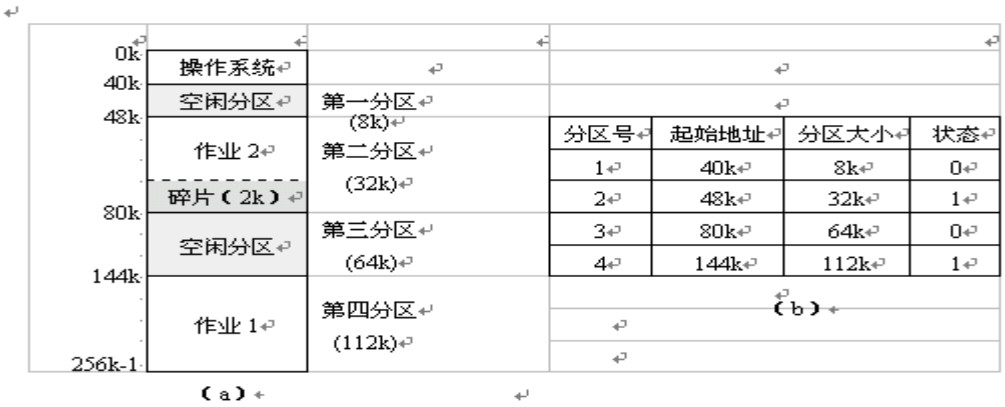
（3）请求调入和预调入方式

分区存储管理

把内存划分成若干分区，每个分区里容纳一个作业。按照分区的划分方式，可归为两种常见的分配方法：固定分区法和动态（可变）分区法。

1. 固定分区法

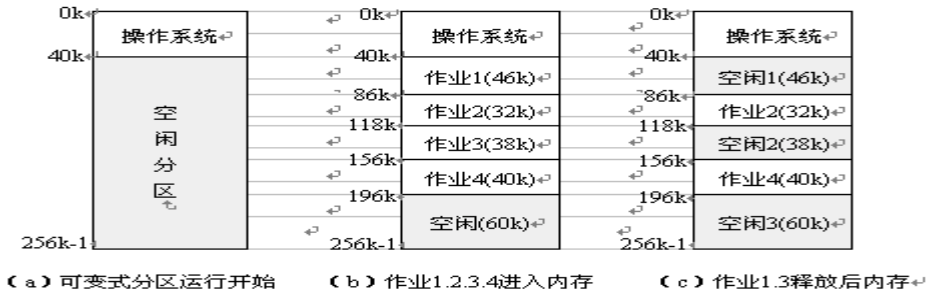
固定分区存储管理是实现多道程序设计的最简单的一种存储管理技术。其基本思想是，在作业未进入内存之前，就由操作员或操作系统把内存可用空间划分成若干个固定大小的存储区，除操作系统占用一个区域外，其余区域为系统中多个用户共享，因为在系统运行期间，分区大小、数目都不变，所以固定式分区也称为静态分区。用户程序装入内存时：①提出申请；②查找空区；③分配空区；④释放已使用区。过程：申请→查找→分配→使用→释放。



2. 动态（可变）分区法

各个分区是在相应作业要进入内存时才建立的，使其大小恰好适应作业的大小，这种技术称为动态分区法。过程：准备→申请→查找→分配→使用→释放。

（1）空闲内存的组织形式



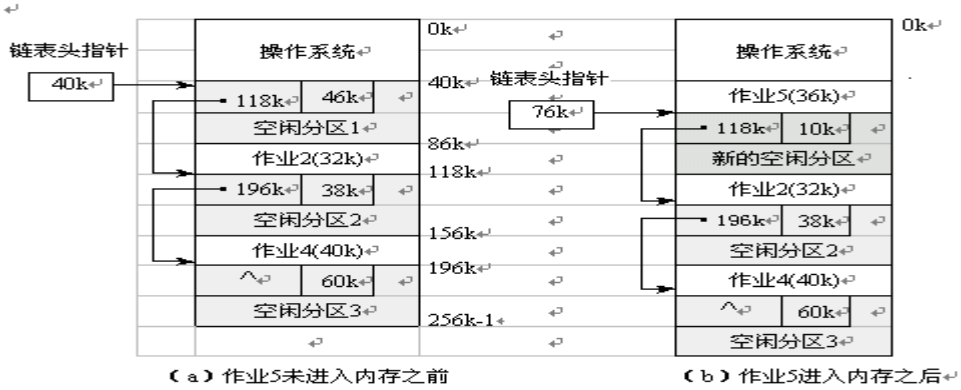
空闲分区链表的组织是这样的：在每个空闲分区的起始部分开辟出一个单元，存放一个链表指针和该分区的大小，链表指针指向下一个空闲分区。系统中用一个固定单元作为空闲分区链表的链表头指针，指向第一块空闲分区首地址，最后一块空闲分区的链表指针存放链尾标志。如图 4.7（a）所示 。

1． 内存的分配和回收

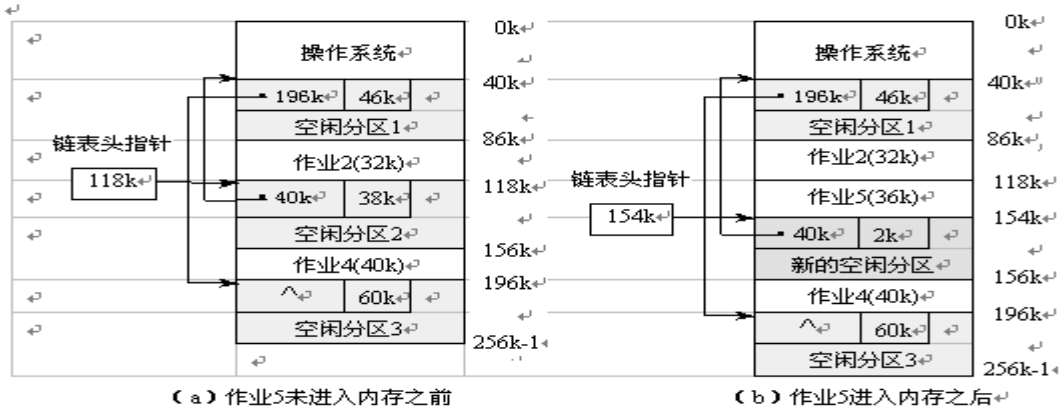
当某一个用户作业完成释放所占分区时，系统应进行回收。在可变式分区中，应该检查回收区与内存中前后空闲区是否相邻，若相邻，则应进行合并，形成一个较大的空闲区，并对相应的链表指针进行修改；若不相邻，应将空闲区插入到空闲区链表的适当位置。

分配算法：

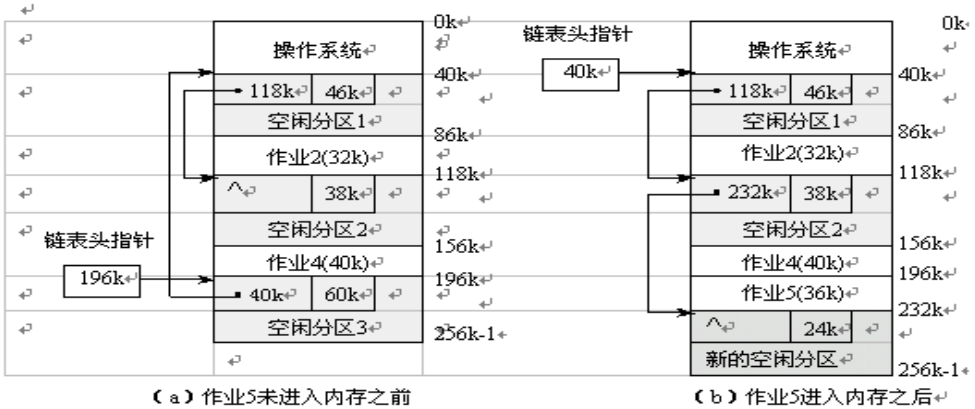
（1）首次适应算法



（2）最佳适应算法



（3）最坏适应算法



回收：

在固定分区法和动态分区法中，用户程序须装入一个连续的内存空间中，由于各作业申请和释放内存的结果，在内存中经常可能出现大量的分散的小空闲区，内存中这种容量太小，无法被利用的小分区称为“碎片”或“零头”。为充分利用这些较小的分散的空闲区，最简单的办法是定时或在分配时把所有的碎片合并为一个连续区（参见教材 P108 图 4-11），实现的方法是移动某些已分配的内容，使所有作业的分区紧挨在一起。而把空闲区留在另一端。这种技术称为紧缩（或叫拼凑）。

动态重定位法：作业在内存中的位置可以移动。

通过紧缩消除碎片，但需耗费大量 CPU 时间。

覆盖技术

思想：把程序划分为若干个功能上相对独立的程序段，按照程序的逻辑结构让那些不会同时执行的程序段共享同一块内存区。

交换技术

早期的交换技术用于单用户系统。这种交换技术是利用外存解决内存不足的问题，但效率很低，因为在执行作业的换入/换出时，CPU 是空闲的。

交换技术：作业在内存和磁盘之间更换。

页式存储管理

1．分页的概念

问题的提出：多个班的学生去礼堂听报告，为便于管理，如果一个班必须坐在一起，有两种处理办法：一是预留好各班的座位，这相当于“固定分区”，但若人未到齐，可能有空位子；二是让其他人挪一下位子，空出足够的座位，让某班人坐在一起，这相当于“可重定位分区”，但要花费一些时间。如果并非必须坐在一起，有座位就随便坐，那既省时又省空间，只是相互间要知道彼此的座号即可，这就是分页储存管理技术。

2．分页存储管理的基本概念

分页存储管理的基本方法是：

（1） 逻辑空间分页：将一个进程的逻辑地址空间划分成若干大小相等的部分，每一部分称做页面或页，每页都有一个编号，叫做页号。

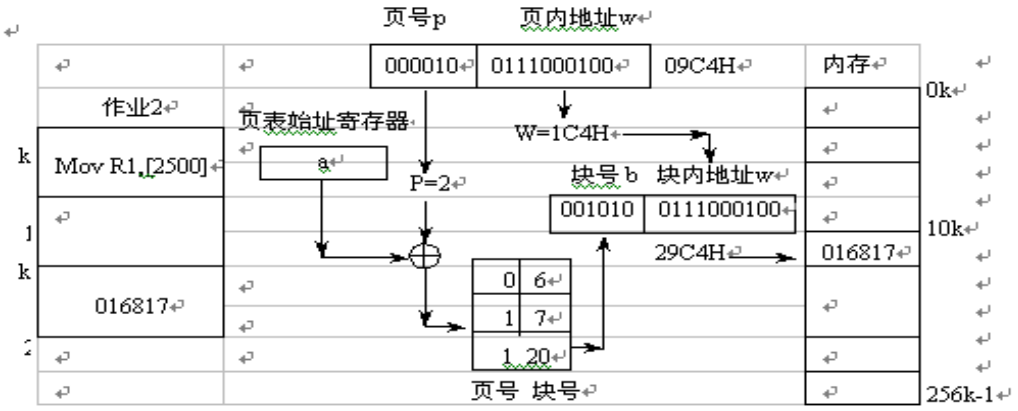
（2） 内存空间分块：把内存也划分成与页面相同大小的若干个存储块，称做内存块或页框。

（3） 逻辑地址表示：在分页存储管理方式中，表示地址的结构：页号+页内相对地址

（4） 内存分配原则：在分页情况下，系统以块为单位把内存分给作业或进程，并且一个进程可装入物理上不相邻的内存块中。

（5）页表，为解决作业或进程离散地分布在内存块中，系统为每个进程设立一张页面映象表，简称页表。其作用是实现从页号到内存块的地址映射，如同邮政编码与单位对照明细表。

2． 分页系统中的地址映射



分段存储管理技术

问题的提出：前面介绍的各种存储管理技术中，提供给用户的逻辑地址空间是一维的线性空间。这与内存的物理组织基本相同。但用户编写的程序的逻辑结构却不是这样的。通常，我们写的程序模块和

数据模块组成，编程者希望程序的地址单元单独占用一片内存空间。这样，程序在内存中的存放情况就与我们的程序的逻辑结构对应起来。为了满足用户在编程和使用等方面的需求，引入分段存储管理技术。

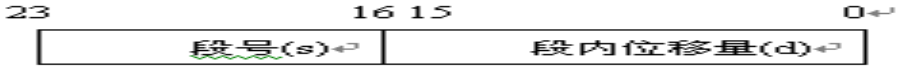
1．分段存储管理的基本概念

分段是一组逻辑信息的集合

每个段都有自己的名字和长度，系统常常为每一段规定一个内部段名，内部段名实际上是一个编号，称为段号。每个段都从 0 开始编址，并采用一段连续的地址空间。段的长度由该段所包含的逻辑信息的长度决定，因而各段长度不等。

2. 程序的地址结构

由于整个作业的地址空间分成多个段，所以，逻辑地址要用两个成分来表示，段号 S 和段内地址，就是说，在分段存储情况下，作业的逻辑地址空间是二维的，就像外地电话号码须拨区号+区内电话号码一样。



在图所示地址结构中，允许一个作业最多有 64K 个段，每段的最大长度为 64KB。

3. 内存分配

在分段存储管理中，内存以段为单位进行分配，每个段单独占用一块连续的内存分区，各分区的大小由对应段的大小决定，这有些类似于动态分区分配方式。但二者有区别：在分段存储管理系统中，一个作业或进程有多个段，这段可以离散地放入内存的不同的分区中。

4.分页和分段的主要区别

相拟之处：二者在内存中都不是整体连续的，都要通过地址映射机构将逻辑地址映射到物理内存中。

主要区别：

（1）页是信息的物理单位，它不考虑一页中是否包含完整的函数，甚至一条指令就可能跨两个页面。因此，用户本身并不需要把程序分页，完全是系统管理上的要求。

（2）段是信息的逻辑单位，每一段在逻辑上是相对完整的一组信息，分段是为了更好地满足用户的需要。

（3）页的大小是由系统固定的，在一个系统中所有页的大小是一样的，只能有一种大小。

（4）段的长度因段而异，取决于用户所编写的程序

（5）分页的作业地址空间是一维的，只需用一个地址编号。分段的作业地址空间是二维的，包括段号，段内地址。

5．分段存储管理的基本原理

在分段基础上实现的虚拟存储器是以段为单位进行换入、换出的，与请求分页存储管理相似，在程序运行之前不要调入所有的分段，只需把当前程序运行时所需的几个分段装入内存（即程序是部分装入内存）就启动程序运行。当程序运行过程中需要访问一个新段，而该段尚未装入内存时，便产生缺段中断，由操作系统把所缺的段调入内存。

在分段存储管理方式中还提供了段的动态连接，即：在程序运行过程中要调用哪段时，才将那段连接上。

6．段的共享和保护

分段管理的一个优点是提供了对代码或数据进行有效地共享。为了共享某个段，只需在各个作业的段表中登记一项，使它们的基址都指向同一物理单元。共享是在段一级出现的。这样，任何共享的信息就可单独成为一段。

分段管理的另一个突出优点是便于各段保护，段的保护措施包括以下几种：

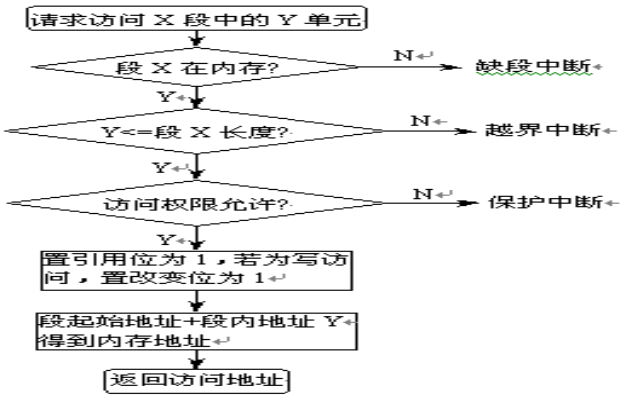
（1）存取控制；

（2）段表本身可起保护作用；

（3）保护环，每个层次的分段有一个保护环，环号越小，级别越高。

在环保护机制下，程序的访问和调用遵循的规则是：一个环内的段可以访问同环内或环号更大的环中的数据段；一个环内的段可以调用同环内或环号更小的几种服务。

7．地址重定位



段页式存储管理技术

分页存储管理能有效地提高内存利用率，而分段存储管理能很好地满足用户需要。把这两种管理技术有机地结合起来“各取所长”，就形成新的存储管理系统——段页式存储管理系统。

1．基本原理

段页式存储管理的技术要点：

（1）等分内存：把整个内存分成大小相等的内存块；

（2）作业或进程的地址空间采用分段的方式；

（3）段内分页：把每一段划分成若干页；

（4）逻辑地址结构，由三部分组成：段号（S）、页号（P）、页内地址（D）记作 V=（S，P，D）；

（5）内存分配：内存的分配单元是内存块；

（6）段表、页表和段表地址寄存器。

总起来说，用户程序逻辑划分成若干段，每段又分成若干页面。内存划分成对应大小的块。进程映象对换是以页为单位进行的，从而使逻辑上连续的段有效的分散的内存块中。

UNIX 中的存储管理技术

UNIX S-V 采用了请求分页存储管理技术和对换技术。

1. 对换

(1) 对换空间的分配

(2) 进程的换出和换入

2. 请求分页

UNIX S—V 中采用请求分页存储管理方式。其作用原理与 4.3.2 节所讲的知识基本相同。大致可分为：数据结构；页面淘汰进程；缺页。

三、本章小结：

存储管理在操作系统中占有重要地位。存储器种类很多，按照其容量，存取速度以及在系统中的作用，可分为三级存储器：高速缓存、内存和外存。

用户程序在计算机上从进入系统到运行要经历编辑、编译、连接、装入和运行等阶段，其中与内存分配有密切关系的阶段是连接阶段和装入阶段。

用于多道程序操作系统的存储管理算法是从最简单的分区方法到复杂的段页式。在一个特定系统中所用策略的决定因素取决于硬件提供的支持。

存储管理算法在很多方面存在着差别，在比较时，应重点考虑以下几点：

（1）硬件支持；

（2）性能；

（3）碎片；

（4）重定位；

（5）对换；

（6）共享；

（7）保护。

虚拟存储技术，允许把大的逻辑地址空间映射到较小的物理内存上，这样就提高了多道程序并发执行的程序，增加 CPU 的利用率，虚拟存储器具有一系列新的特征，包括：虚拟扩充、部分装入、离散分配和多次对换等。

请求分页式存储管理是根据实际程序执行的顺序，动态申请存储块的，并不是把所有页面都放入内存。

当内存的需求量超出实际内存量时，为释放内存块给新的页面，需要进行页面置换。有多种方法，FIFO 是最容易实现的，但性能不是很好。最佳算法需要未来知识，仅有理论价值。LRU 是 OPT 的近拟算法，但实现时要有硬件的支持和软件开销。多数页面置换算法，如最近未使用置换法等都是 LRU 的近似算法。

分段技术能很好地满足用户需要。分段是一组信息逻辑单位，提供了用户可见的三维地址空间。

段页式虚拟存储管理是在分段管理基础上加虚存技术，一个作业的各个模块可根据调用需要进行动态装入。

在大型通用系统中，往往把段页式存储管理和虚存技术结合起来，形成带虚存的段页式系统，它兼顾了分段在逻辑上的优点和请求分页式在存储管理方面的长处，是最通用、最灵活的系统。

UNIX S—S 采用对换和请求分页存储管理技术，页面淘汰采用 LRU 算法。为实现对换和请求分页，系统设立了很多数据结构、便于各分区的共享和保护。

本章重点把握下列概念和方法：

请求分页的基本思想；

地址空间分页，内存分块、页与块大小相同；

作业部分装入内存；

作业所占的各块不连续；

硬件通过页表生成访内地址；

若缺页，进行缺页中断处理，换入内存；

利用快表可加速地址转换。