窦轶////yi.dou@njupt.edu.cn

Operating Systems

操作系统

存储管理基本概念

Linux
Android
Linux
OpenStack
Mac OS
Windows

存储管理基本概念

本讲内容

1.计算机中的存储体系

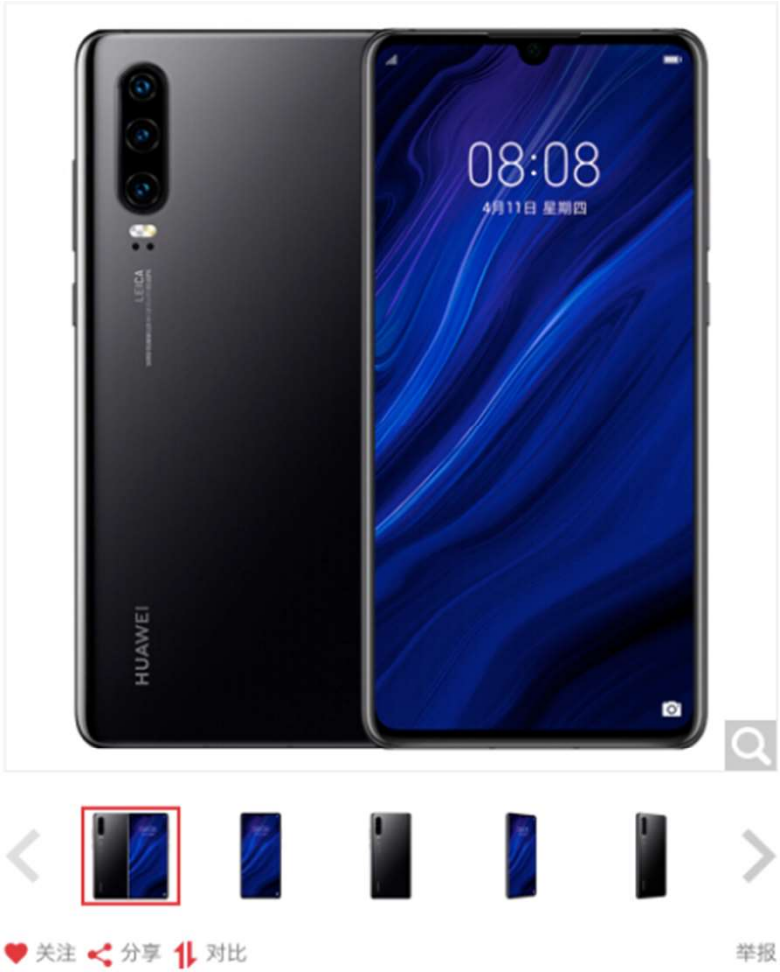
2. 存储管理目标及任务

3. 连续存储区管理方案

4. 分区存储的管理方案

5. 存储覆盖与交换技术

什么是内存？有何作用？



关注 分享 对比 举报

新品 华为 HUAWEI P30 超感光徕卡三摄麒麟980AI智能芯片全面屏屏内指纹版手机8GB+64GB亮黑色全网通双4G手机双

现在购机享受价保至6月18日，618提前购，安心购买！屏内指纹，感光徕卡三摄mate20优惠200、到手价3299起、还有赠品

618 全球年中购物节

京东价 ¥3988.00 降价通知

累计评价 14万+

促 销 满送 满100元即赠热销商品，赠完即止

满额返券 购买此商品满10元返配件品类优惠券（送完为止） 详情 >>

增值业务 高价回收-卖了换钱 套 3元1G

配送至 北京海淀区三环以内 有货

由 京东 发货，并提供售后服务。23:00前下单，预计明天(05月28日)送达

重 量 0.48kg

服务支持 自营放心购 免举证退换货 原厂维修

京尊达 99元免基础运费(20kg内) 京准达 自提

选择颜色 天空之境 亮黑色 极光色 赤茶橘 珠光贝母

选择版本 8GB+64GB 8GB+128GB 8GB+256GB

选择版本 标准版 碎屏险套装版 京享无忧版 特惠版

套 装 优惠套装1 优惠套装2 优惠套装3 优惠套装4 优惠套装5 优惠套装6

什么是内存？有何作用？

电脑、办公 > 电脑整机 > 游戏本 > 外星人（Alienware） > 外星人Alienware 17

自营 外星人京东自营旗舰店 联系客服 关注店铺

关注 分享 对比 举报

企业购更优惠

外星人17.3英寸机皇4K游戏笔记本电脑(i9-8950HK 32G 1T固态X2 1T GTX1080 8G独显 UHD)

【全尺寸机皇新高度】i9+GTX1080+4K超清屏,全方位无死角,双1TBSSD+1TBHDD,兼顾速度与容量。

618 全球年中购物节

京东价 ¥32999.00 降价通知

累计评价 1500+

增值业务 高价回收，享补贴

配送至 北京海淀区三环以内 有货 支持 京尊达 99元免基础运费(20kg内)

由 京东 发货，并提供售后服务. 有货（外地跨区调货），暂免调货服务费. 18:00前下单，预计**05月29日(周三)**送达

重 量 7.8kg

选择颜色

- i9-8950HK 16G RTX2080MQ 8G 红
- i7-8750H 16G RTX2080MQ 8G 银
- i7-8750H 16G RTX2070MQ 8G 银
- i7-8750H 16G RTX2060 6G 银
- i7-8750H 16G RTX2060 OC 6G独显
- i7-8750H 16G GTX1660Ti 6G独显
- i9-8950HK 32G GTX1080 8G独显
- i9-8950HK 32G GTX1080 8G UHD
- i7-7820HK 16G GTX1080 8G独显
- i9-8950HK 16G GTX1070 8G独显
- i7-8750H 16G GTX1060 6G独显 黑
- i7-8750H 16G GTX1070 8G独显
- i7-8750H 16G GTX1070 8G QHD
- i7-8750H 16G GTX1060 6G QHD

什么是内存？有何作用？

关注 分享 对比 举报

京东物流 三星（SAMSUNG） DDR4 2400 2133 4G 8G 四代笔记本内存条 三星原厂正品 DDR4 2133 4GB

品牌机原厂内存供应商 买就买个真的的内存 京东自营仓直发 原厂正品内存 不烧机 吃鸡不蓝屏

京东价 ¥355.00 降价通知

累计评 3700

促 销 赠品 ×1 ×1 （赠完即止）

增值业务 以旧换新，卖了换钱

配送至 北京海淀区五环到六环之间 有货 支持 货到付款 免运费

由 京东 发货，本尚网来数码专营店 提供售后服务. 11:10前下单,预计今天(08月03日)送达

选择颜色


DDR4 2400 4GB


DDR4 2400 8GB


DDR4 2133 16GB


DDR4 2400 16GB


DDR4 2133 4GB


DDR4 2133 8GB

增值保障 全保换2年 ¥29 电脑组装保 ¥99 服务送鼠标 ¥70

1 + -

加入购物车

什么是内存？有何作用？

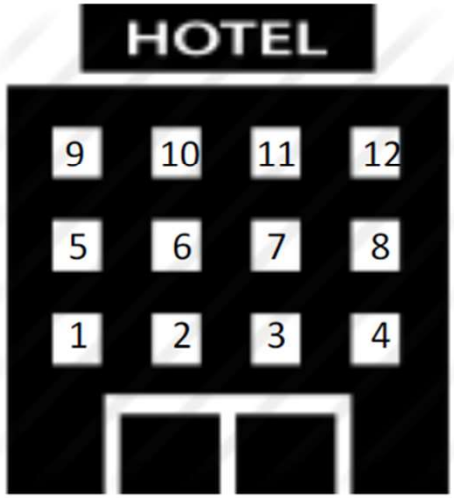
内存可存放数据。程序执行前需要先放到内存中才能被CPU处理——缓和CPU与硬盘之间的速度矛盾

思考：在多道程序环境下，系统中会有多个程序并发执行，也就是说会有多个程序的数据需要同时放到内存中。那么，如何区分各个程序的数据是放在什么地方的呢？

什么是内存？有何作用？

内存可存放数据。程序执行前需要先放到内存中才能被CPU处理——缓和CPU与硬盘之间的速度矛盾

思考：在多道程序环境下，系统中会有多个程序并发执行，也就是说会有多个程序的数据需要同时放到内存中。那么，如何区分各个程序的数据是放在什么地方的呢？



内存

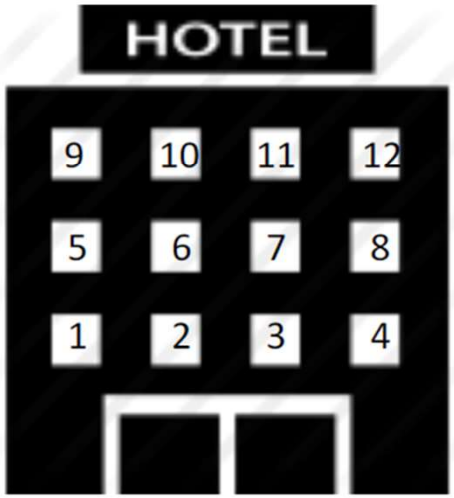
内存中也有一个一个的“小房间”，每个小房间就是一个“存储单元”

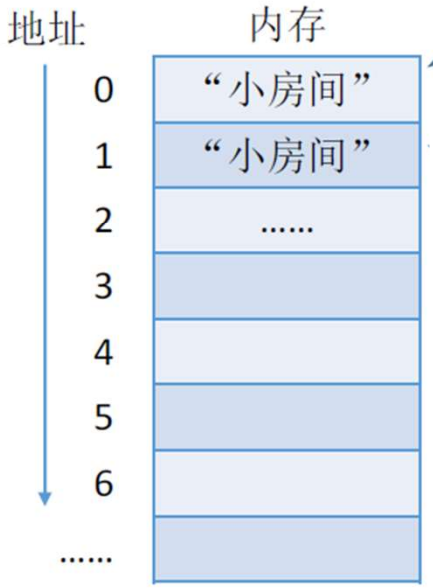
什么是内存？有何作用？

内存可存放数据。程序执行前需要先放到内存中才能被CPU处理——缓和CPU与硬盘之间的速度矛盾

思考：在多道程序环境下，系统中会有多个程序并发执行，也就是说会有多个程序的数据需要同时放到内存中。那么，如何区分各个程序的数据是放在什么地方的呢？

方案：给内存的存储单元编地址





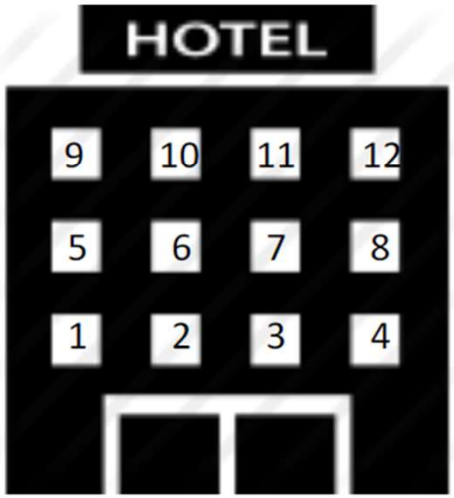
内存中也有一个一个的“小房间”，每个小房间就是一个“存储单元”

什么是内存？有何作用？

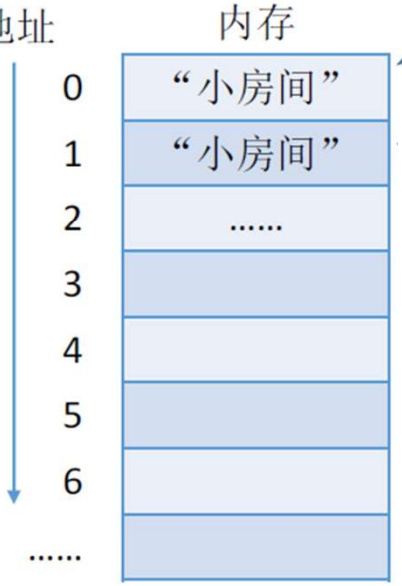
内存可存放数据。程序执行前需要先放到内存中才能被CPU处理——缓和CPU与硬盘之间的速度矛盾

思考：在多道程序环境下，系统中会有多个程序并发执行，也就是说会有多个程序的数据需要同时放到内存中。那么，如何区分各个程序的数据是放在什么地方的呢？

方案：给内存的存储单元编地址



内存地址从0开始，每个地址对应一个存储单元



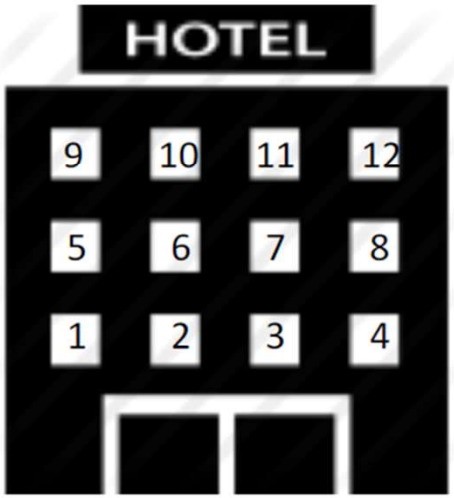
内存中也有一个一个的“小房间”，每个小房间就是一个“存储单元”

什么是内存？有何作用？

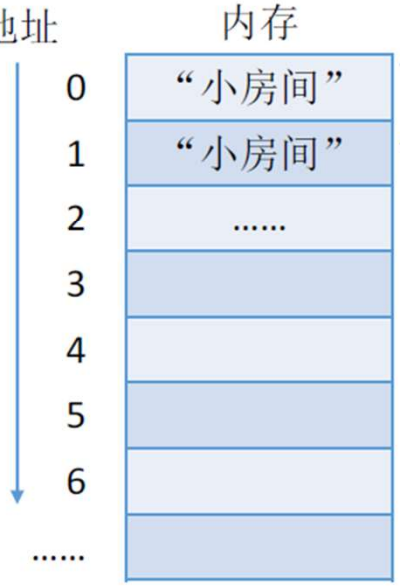
内存可存放数据。程序执行前需要先放到内存中才能被CPU处理——缓和CPU与硬盘之间的速度矛盾

思考：在多道程序环境下，系统中会有多个程序并发执行，也就是说会有多个程序的数据需要同时放到内存中。那么，如何区分各个程序的数据是放在什么地方的呢？

方案：给内存的存储单元编地址



内存地址从0开始，每个地址对应一个存储单元



内存中也有一个一个的“小房间”，每个小房间就是一个“存储单元”

如果计算机“按字节编址”，则每个存储单元大小为1字节，即1B，即8个二进制位

计算机中的存储体系

1 金字塔体系

- 越往上，存储介质的访问速度越快，价格也越高。
- 寄存器、高速缓存、主存储器和磁盘缓存均属于存储管理的管辖范畴，掉电后它们存储的信息不再存在。
- 固定磁盘和可移动存储介质属于设备管理的管辖范畴，它们存储的信息将被长期保存。
- 而磁盘缓存本身并不是一种实际存在的存储介质，它依托于固定磁盘，提供对主存储空间的扩充。

计算机中的存储体系

2 主存储器（内存）

- 内存空间是由存储单元组成的一维连续的地址空间
- 存放代码及数据等信息

存储管理基本概念

本讲内容

1.计算机中的存储体系

2. 存储管理目标及任务

3. 连续存储区管理方案

4. 分区存储的管理方案

5. 存储覆盖与交换技术

存储管理目标及任务

1 管理目标

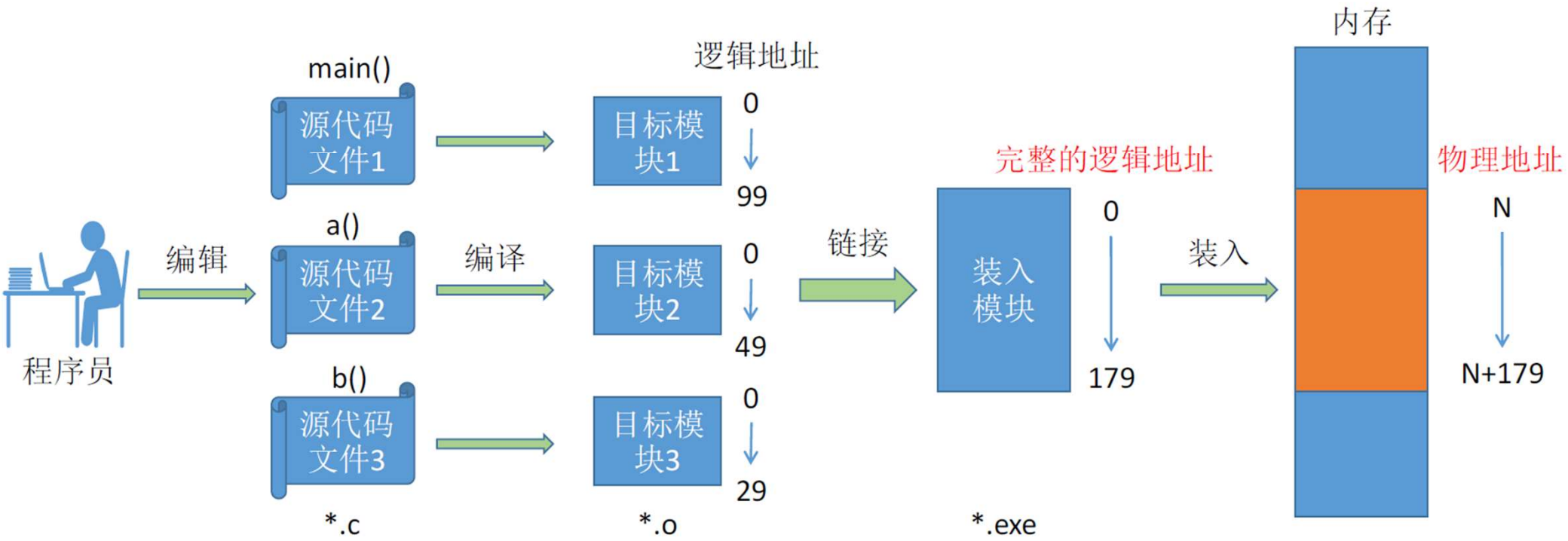
- 充分利用内存空间，为进程的并发执行提供存储支持
- 方便用户使用，自动装入程序、数据，用户不必考虑硬件细节

存储管理目标及任务

2 功能任务

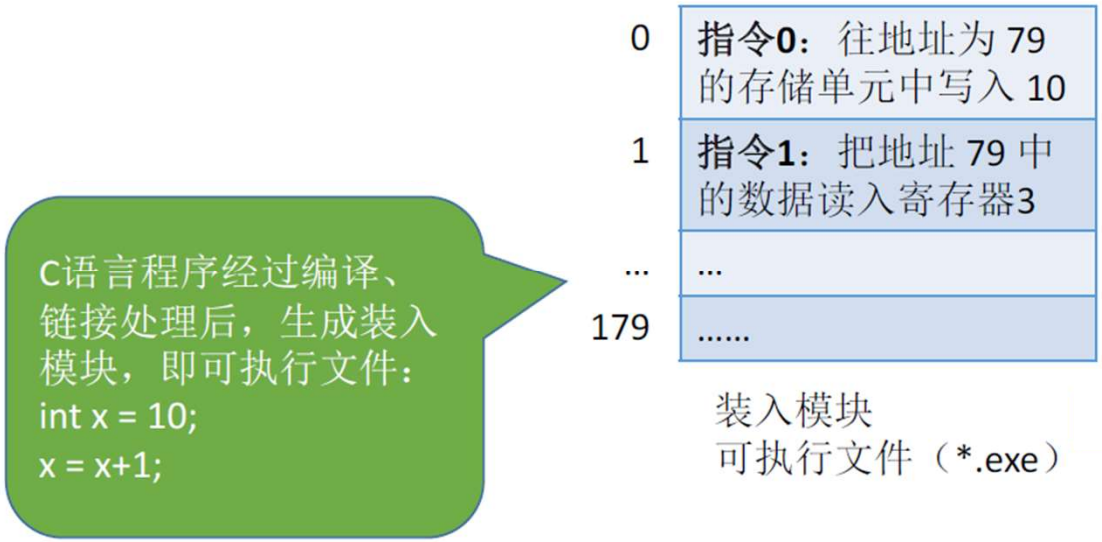
- 实现内存空间的管理、分配与回收
- 实现逻辑地址到物理地址的自动转换
- 实现多个进程间的信息共享与通信
- 实现多个进程隔离和信息保护
- 实现内存和辅存空间的协同使用

从写程序到程序运行



编译：由编译程序将用户源代码编译成若干个目标模块（编译就是把高级语言翻译为机器语言）
链接：由链接程序将编译后形成的一组目标模块，以及所需库函数链接在一起，形成一个完整的装入模块
装入（装载）：由装入程序将装入模块装入内存运行

知识滚雪球：指令的工作原理



Tip：为了简化理解，本节中我们默认操作系统给进程分配的是一片连续的内存空间

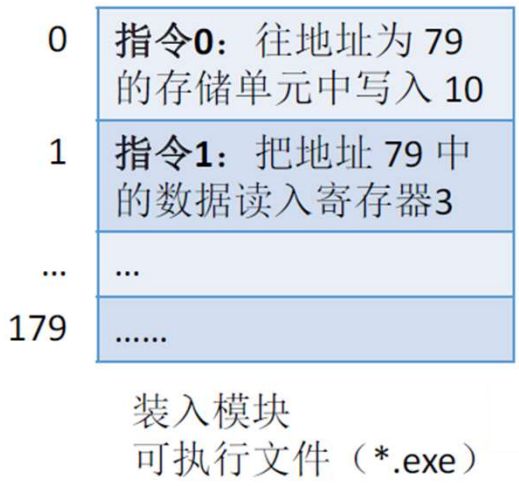
int x 这个变量的地址是79，并且初试值为x=10

知识滚雪球：指令的工作原理

程序经过编译、链接后生成的指令中指明的是逻辑地址（相对地址），即：相对于进程的起始地址而言的地址

C语言程序经过编译、链接处理后，生成装入模块，即可执行文件：

```
int x = 10;
x = x+1;
```



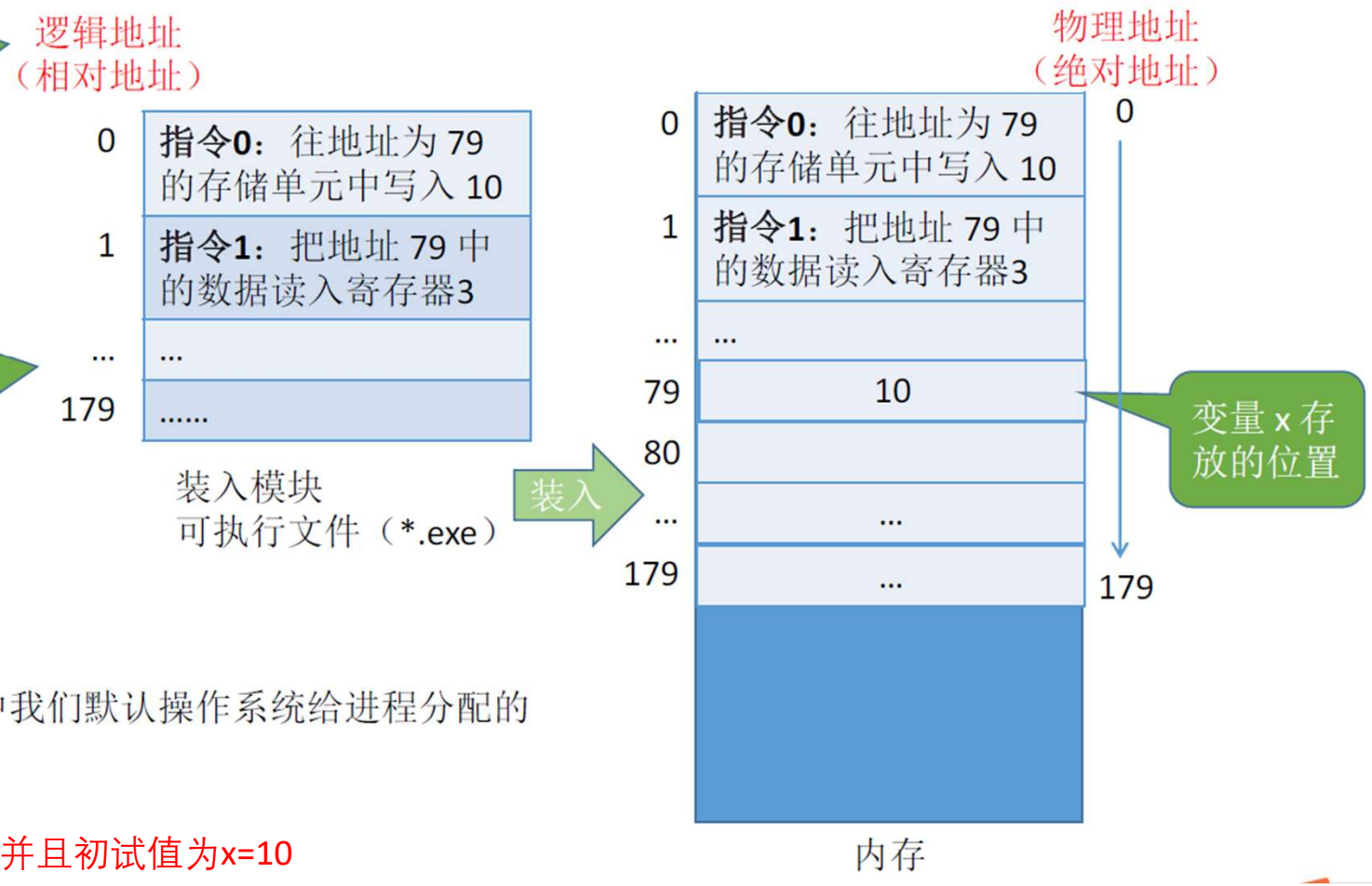
Tip：为了简化理解，本节中我们默认操作系统给进程分配的是一片连续的内存空间

int x 这个变量的地址是79，并且初试值为x=10

知识滚雪球：指令的工作原理

程序经过编译、链接后生成的指令中指明的是逻辑地址（相对地址），即：相对于进程的起始地址而言的地址

C语言程序经过编译、链接处理后，生成装入模块，即可执行文件：
int x = 10;
x = x+1;



Tip：为了简化理解，本节中我们默认操作系统给进程分配的是一片连续的内存空间

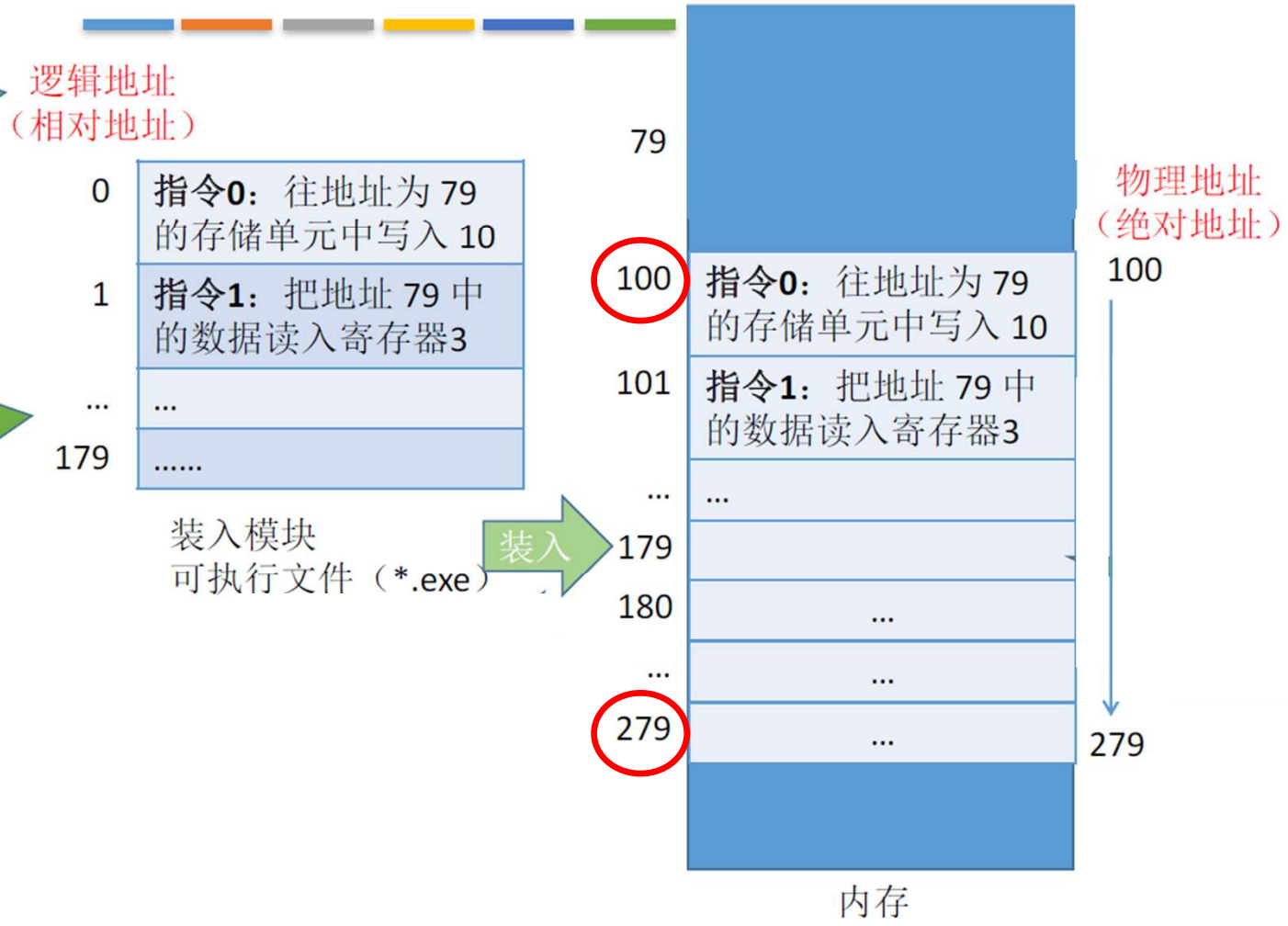
int x 这个变量的地址是79，并且初试值为x=10

知识滚雪球：指令的工作原理

程序经过编译、链接后生成的指令中指明的是逻辑地址（相对地址），即：相对于进程的起始地址而言的地址

C语言程序经过编译、链接处理后，生成装入模块，即可执行文件：

```
int x = 10;
x = x+1;
```

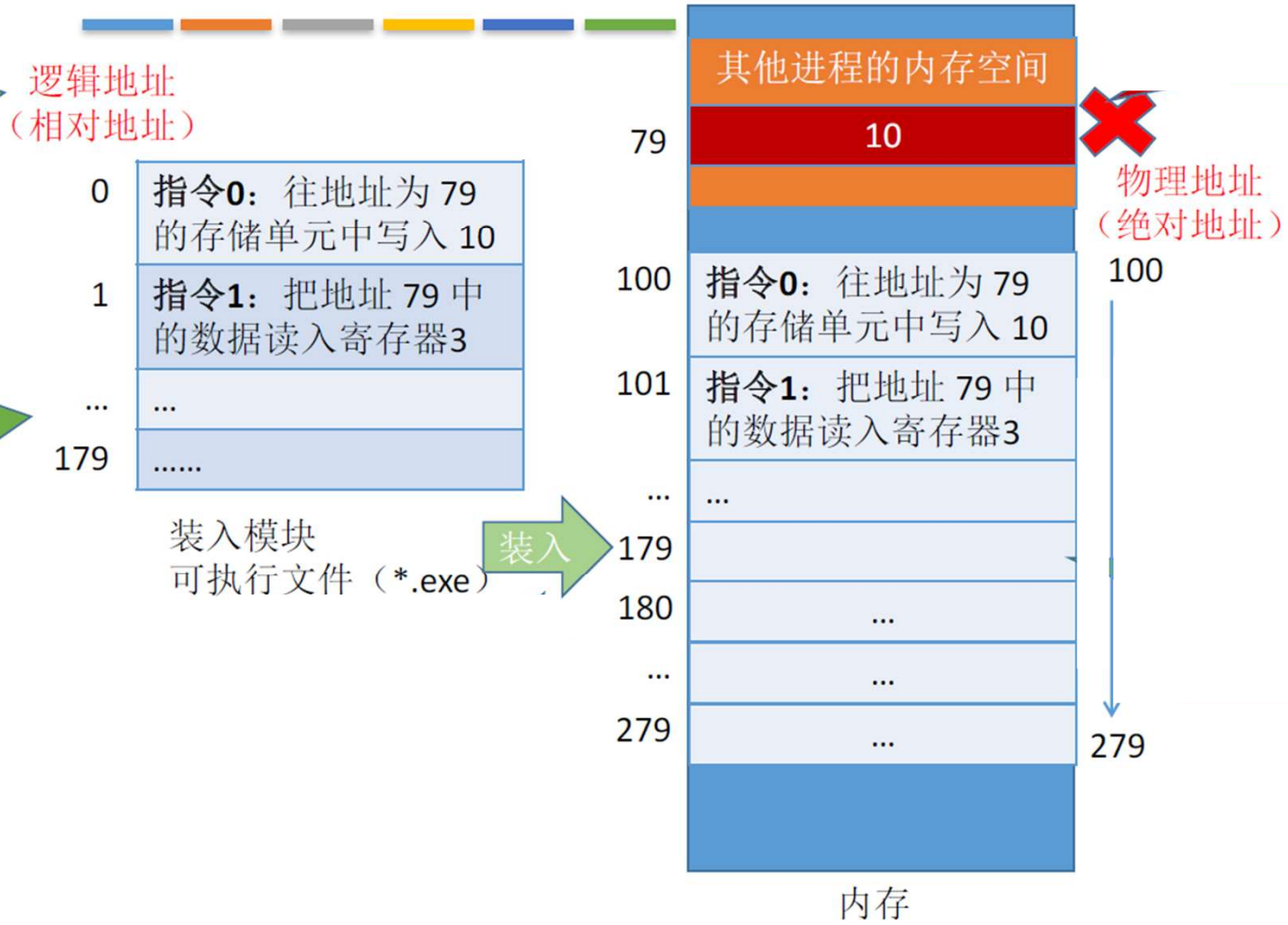


知识滚雪球：指令的工作原理

程序经过编译、链接后生成的指令中指明的是逻辑地址（相对地址），即：相对于进程的起始地址而言的地址

C语言程序经过编译、链接处理后，生成装入模块，即可执行文件：

```
int x = 10;
x = x+1;
```

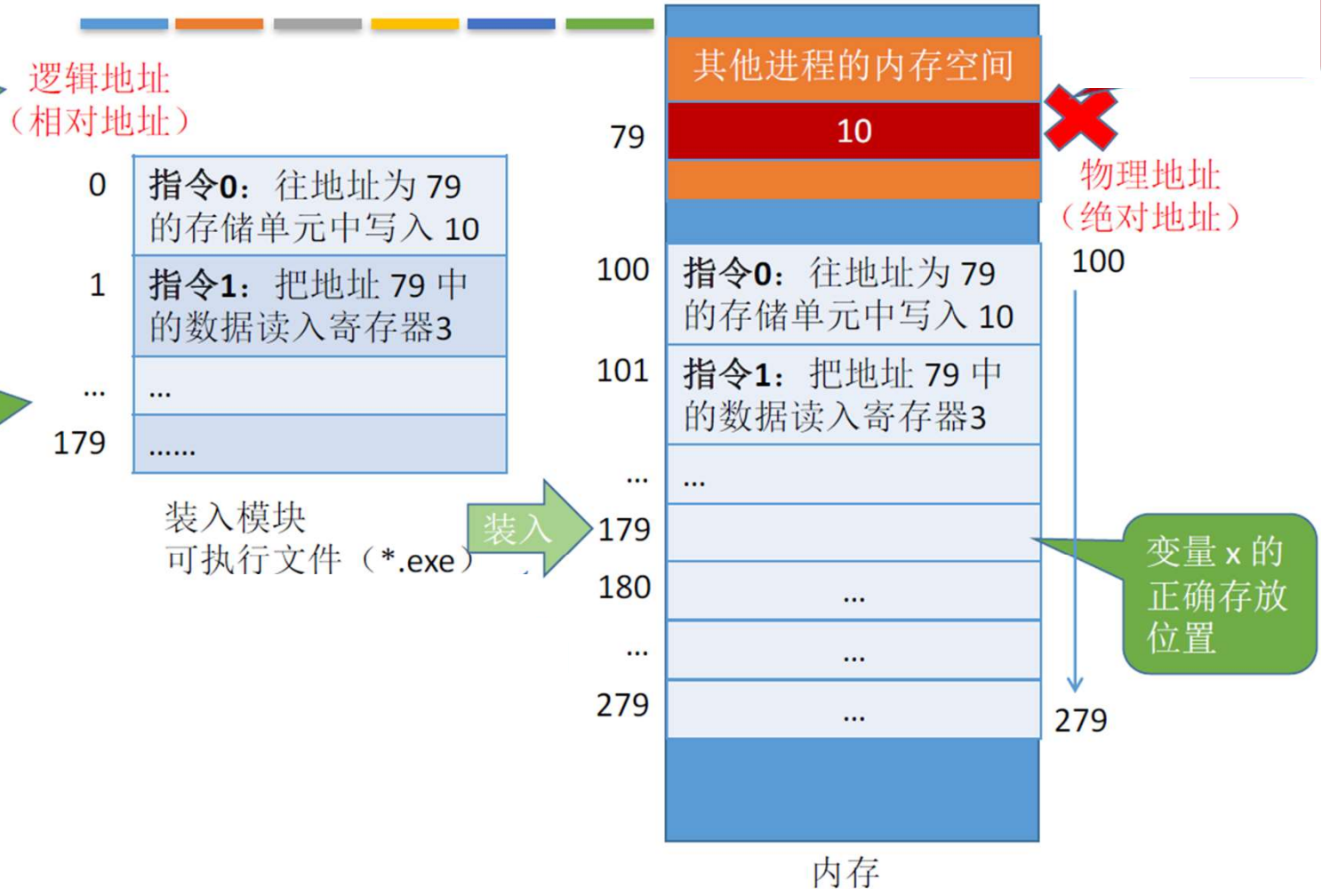


知识滚雪球：指令的工作原理

程序经过编译、链接后生成的指令中指明的是逻辑地址（相对地址），即：相对于进程的起始地址而言的地址

C语言程序经过编译、链接处理后，生成装入模块，即可执行文件：

```
int x = 10;
x = x+1;
```

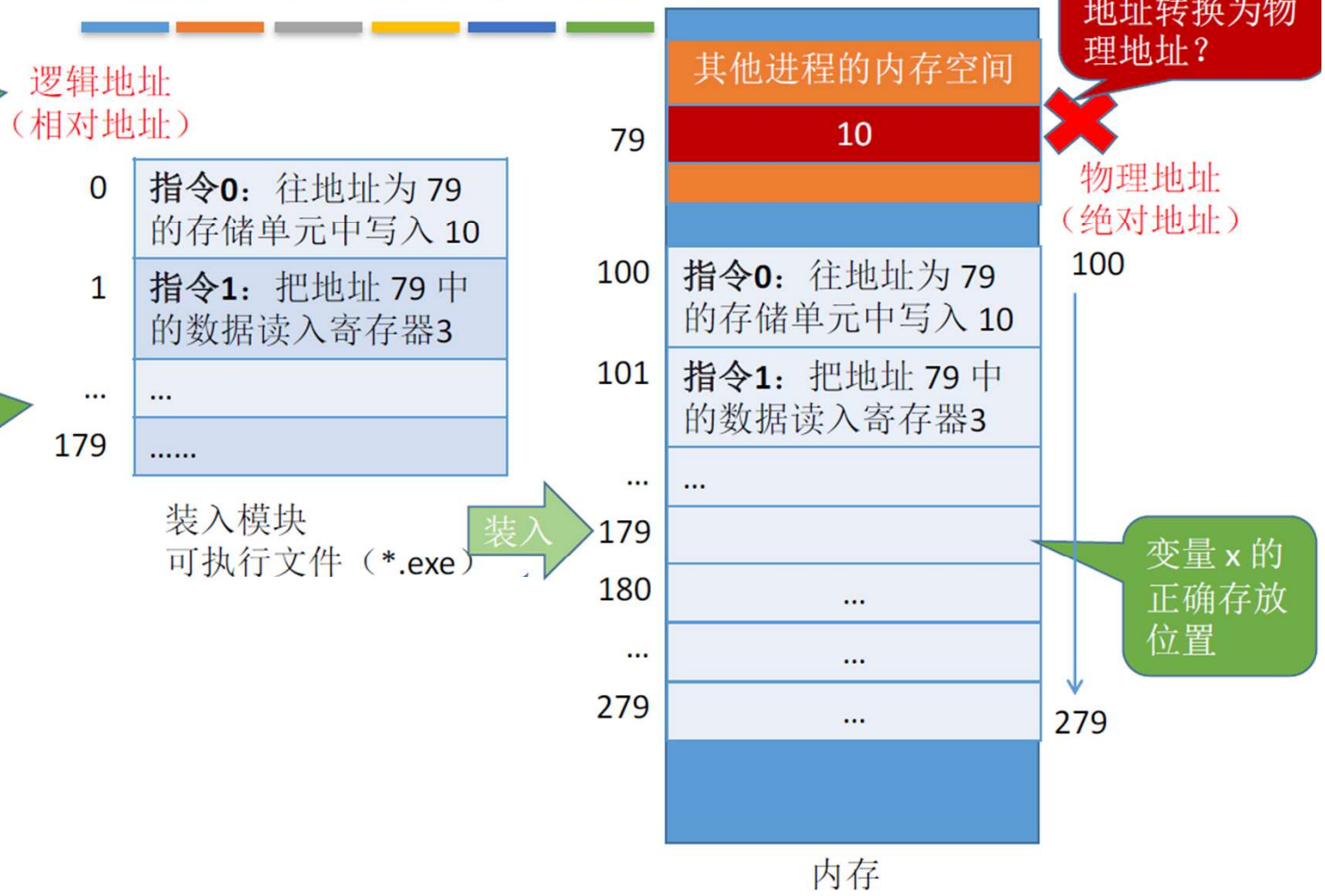


知识滚雪球：指令的工作原理

程序经过编译、链接后生成的指令中指明的是逻辑地址（相对地址），即：相对于进程的起始地址而言的地址

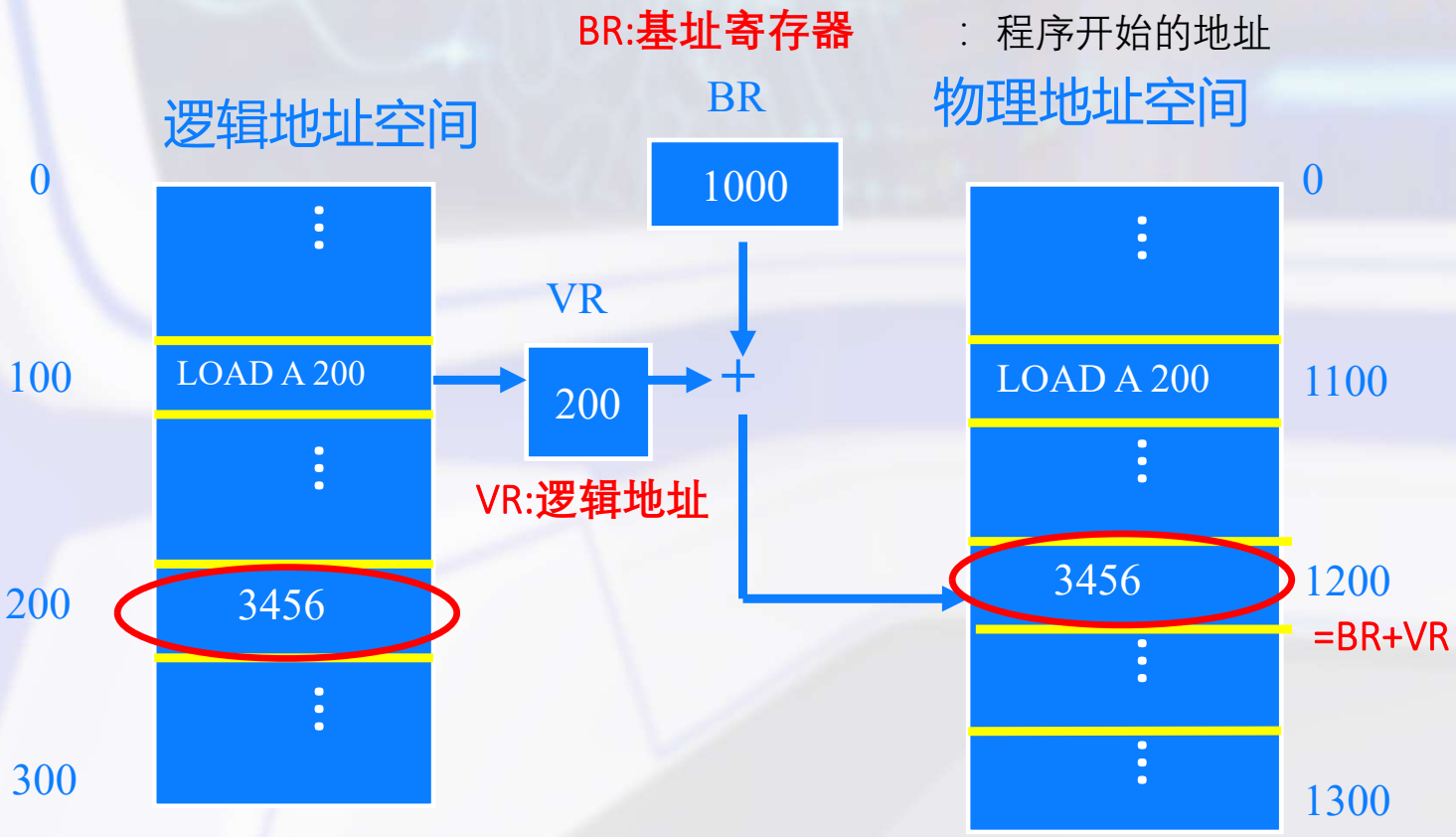
C语言程序经过编译、链接处理后，生成装入模块，即可执行文件：

```
int x = 10;
x = x+1;
```



存储管理目标及任务

3 地址重定位



存储管理基本概念

本讲内容

1.计算机中的存储体系

2. 存储管理目标及任务

3. 连续存储区管理方案

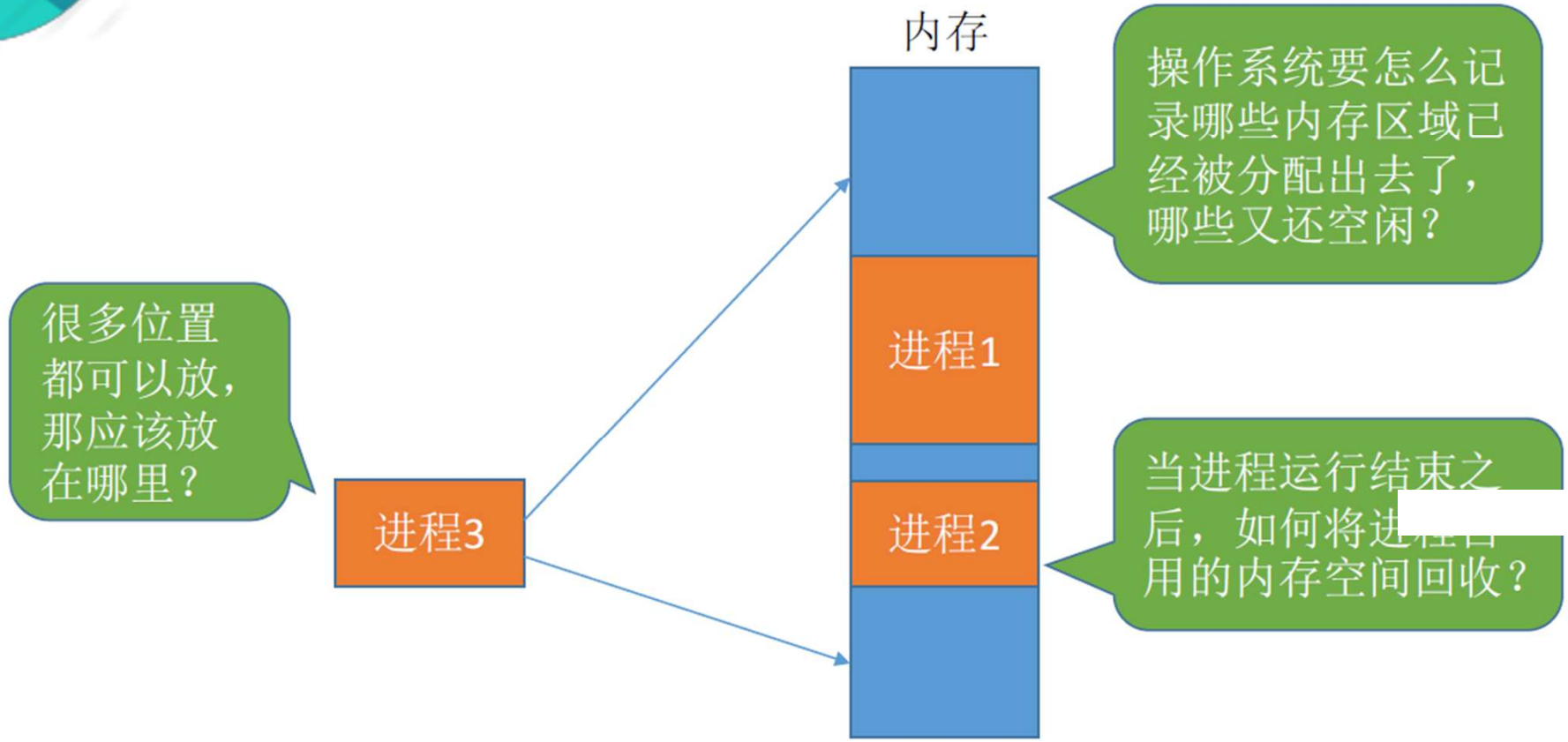
4. 分区存储的管理方案

5. 存储覆盖与交换技术

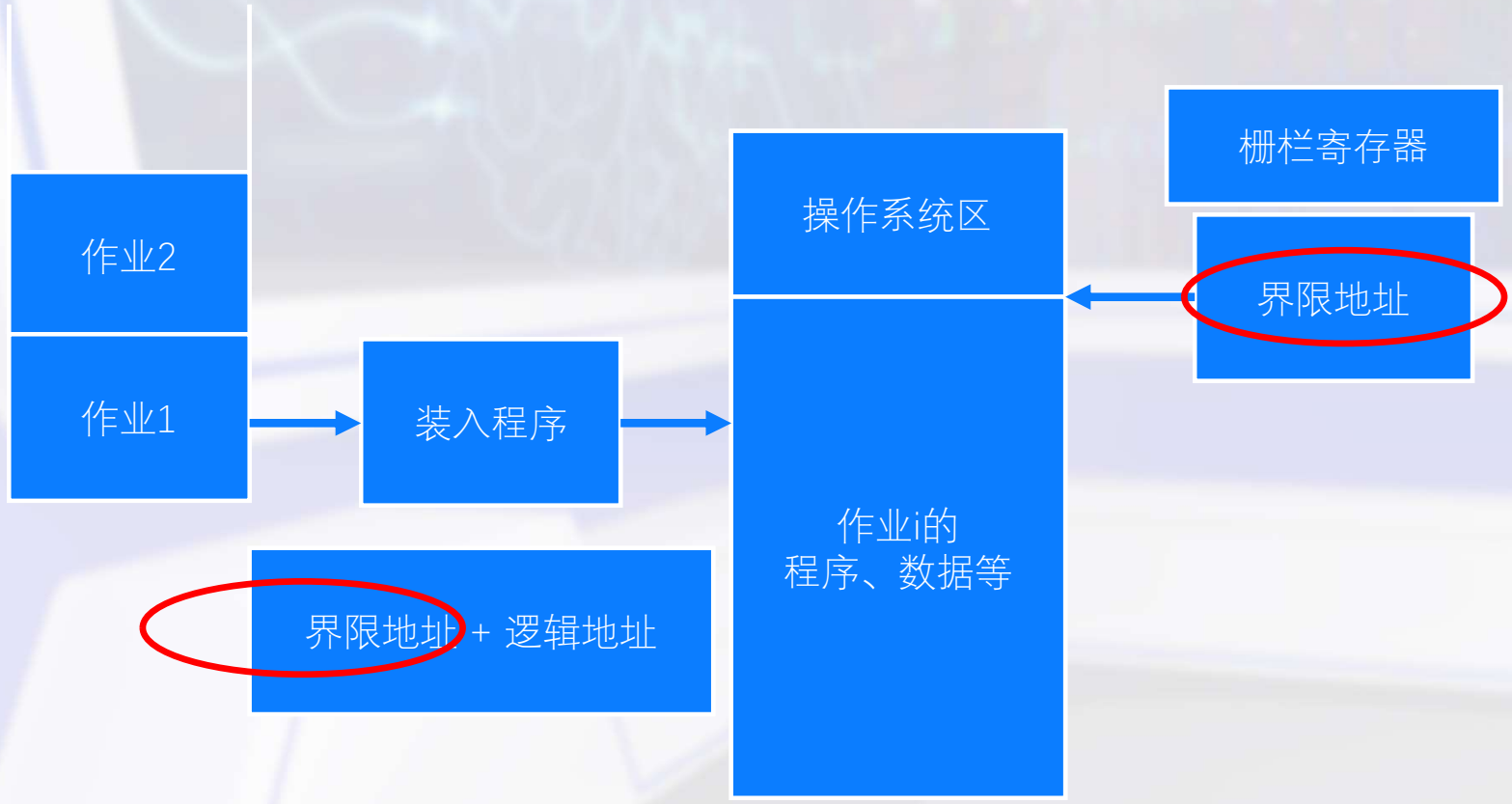
内存空间的分配与回收

操作系统作为系统资源的管理者，当然也需要对内存进行管理，要管些什么呢？

1. 操作系统负责*内存空间的分配与回收*

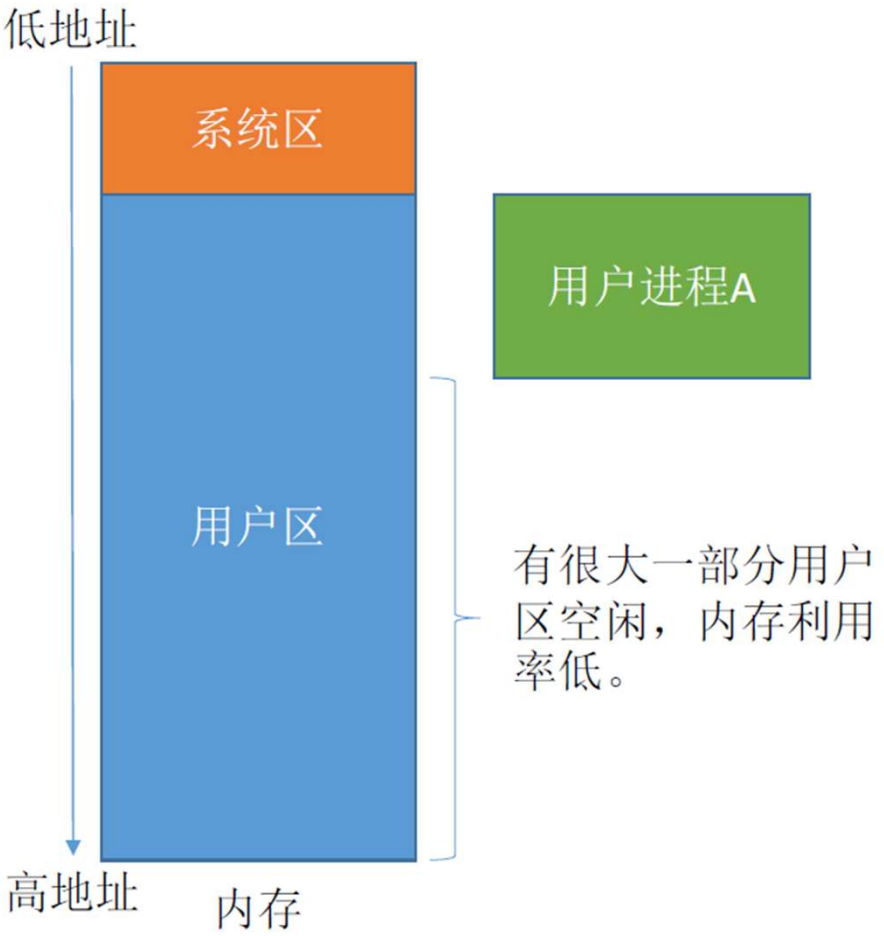


连续存储区管理方案



单一连续分配

在单一连续分配方式中，内存被分为系统区和用户区。
系统区通常位于内存的低地址部分，用于存放操作系统相关数据；用户区用于存放用户进程相关数据。
内存中只能有一道用户程序，用户程序独占整个用户区空间。

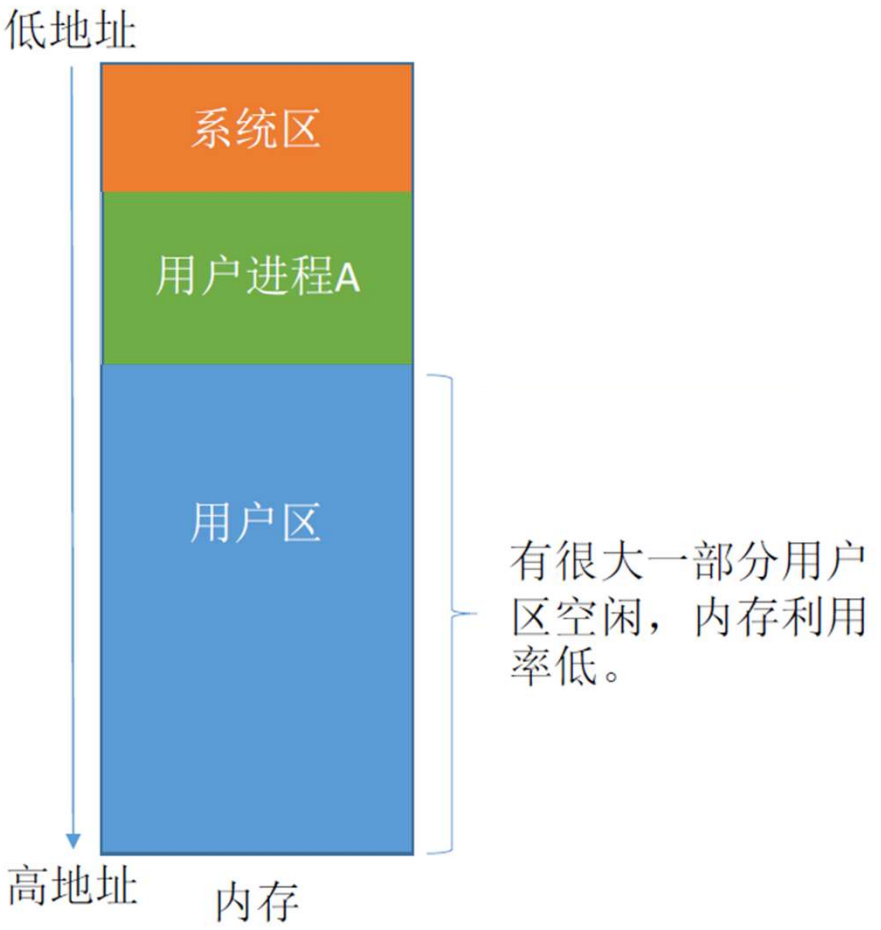


单一连续分配

在单一连续分配方式中，内存被分为系统区和用户区。系统区通常位于内存的低地址部分，用于存放操作系统相关数据；用户区用于存放用户进程相关数据。
内存中只能有一道用户程序，用户程序独占整个用户区空间。

优点：实现简单；无外部碎片；可以采用覆盖技术扩充内存；不一定需要采取内存保护（eg：早期的 PC 操作系统 MS-DOS）。
缺点：只能用于单用户、单任务的操作系统中；有内部碎片；存储器利用率极低。

分配给某进程的内存区域中，如果有些部分没有用上，就是“内部碎片”



存储管理基本概念

本讲内容

1.计算机中的存储体系

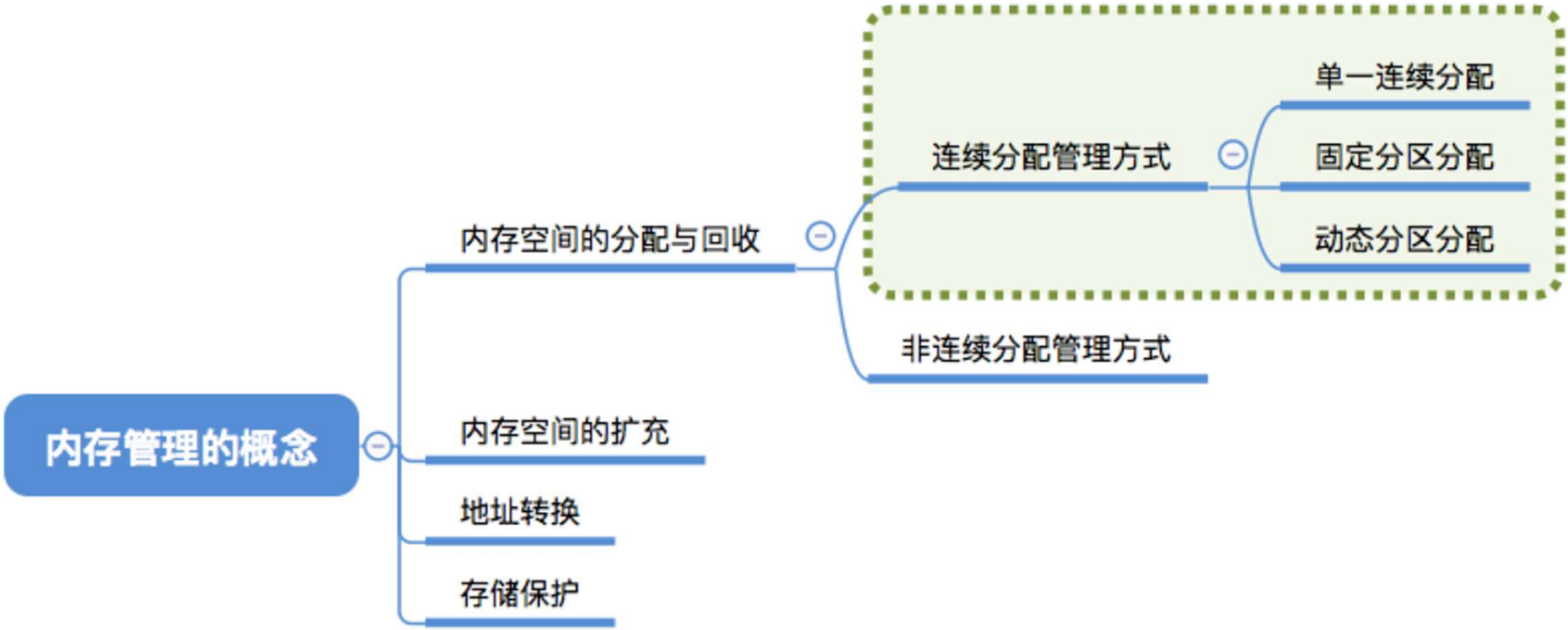
2. 存储管理目标及任务

3. 连续存储区管理方案

4. 分区存储的管理方案

5. 存储覆盖与交换技术

知识总览



连续分配：指为用户进程分配的必须是一个连续的内存空间。

分区存储的管理方案

1 存储分区

- 系统把内存用户区划分为若干分区
- 分区大小可以相等，也可以不等
- 一个进程占据一个分区

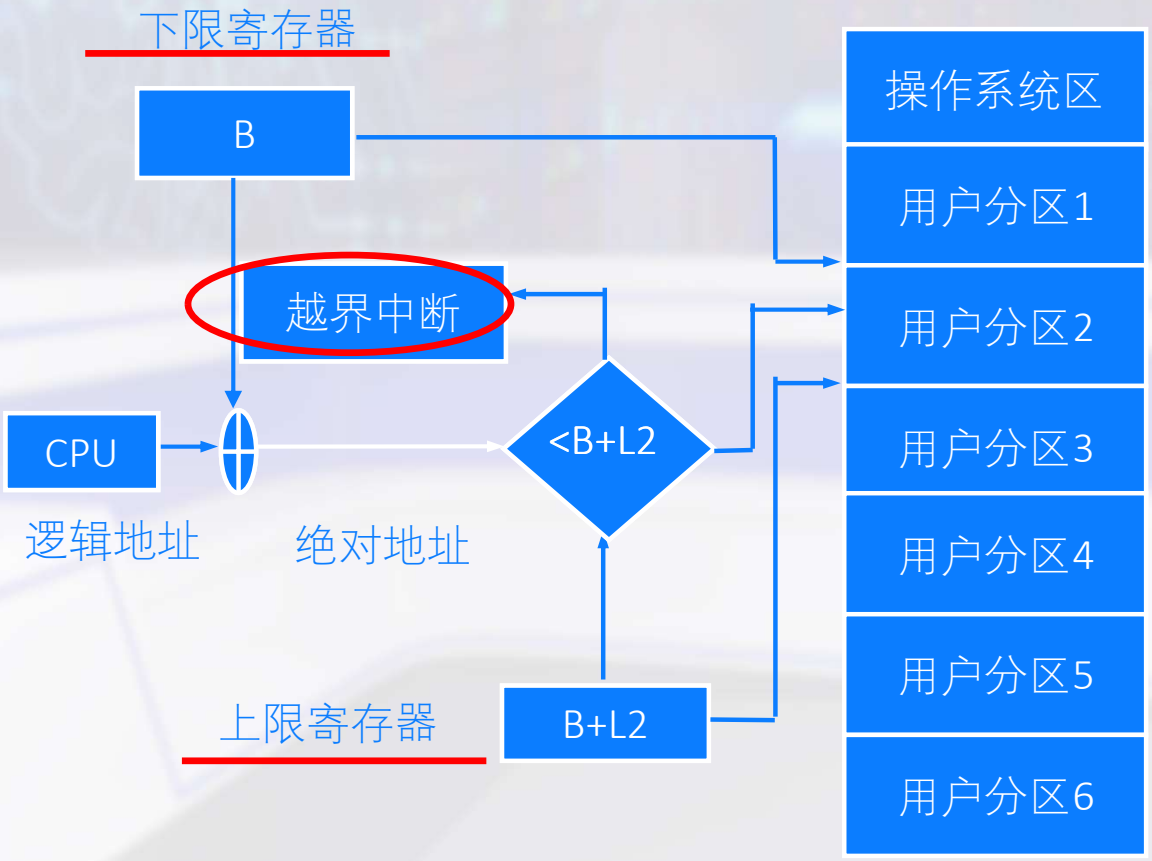
固定分区存储

可变分区存储

分区存储的管理方案

2 固定分区

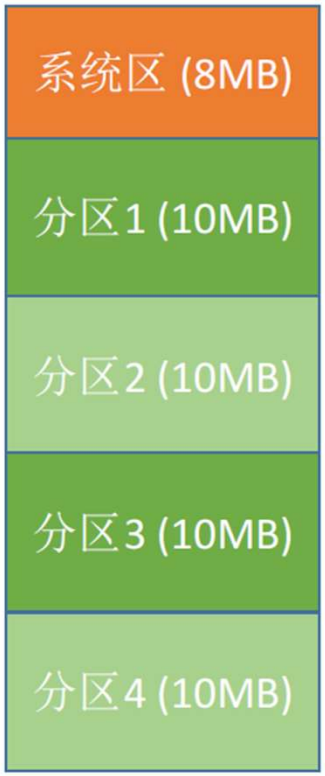
- 预先将内存分割成若干个连续区域
- 如果有足够大空闲区，则分配给创建的进程



固定分区分配

分区大小相等：缺乏灵活性，但是很适合用于用一台计算机控制多个相同对象的场合（比如：钢铁厂有n个相同的炼钢炉，就可把内存分为n个大小相等的区域存放n个炼钢炉控制程序）

分区大小不等：增加了灵活性，可以满足不同大小的进程需求。根据常在系统中运行的作业大小情况进行划分（比如：划分多个小分区、适量中等分区、少量大分区）

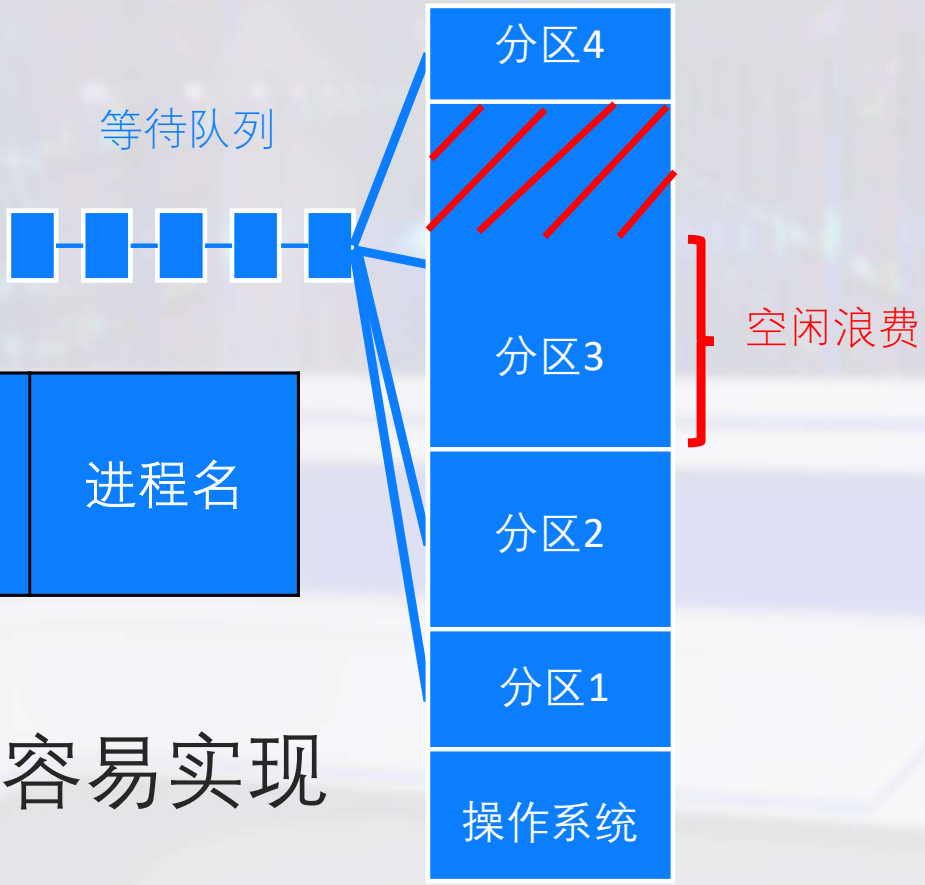


内存（分区大小相等）

内存（分区大小不等）

分区存储的管理方案

2 固定分区



分区号	起始地址	长度	状态	进程名

- 优点：内存管理简单，容易实现
- 缺点：内存利用率低，不够灵活

固定分区分配

操作系统需要建立一个数据结构——分区说明表，来实现各个分区的分配与回收。每个表项对应一个分区，通常按分区大小排列。每个表项包括对应分区的大小、起始地址、状态（是否已分配）。

分区号	大小（MB）	起始地址（M）	状态
1	2	8	未分配
2	2	10	未分配
3	4	12	已分配
……	……	……	……

用数据结构的数组（或链表）即可表示这个表

内存（分区大小不等）

固定分区分配

操作系统需要建立一个数据结构——分区说明表，来实现各个分区的分配与回收。每个表项对应一个分区，通常按分区大小排列。每个表项包括对应分区的大小、起始地址、状态（是否已分配）。

分区号	大小（MB）	起始地址（M）	状态
1	2	8	未分配
2	2	10	未分配
3	4	12	已分配
……	……	……	……

用数据结构的数组（或链表）即可表示这个表

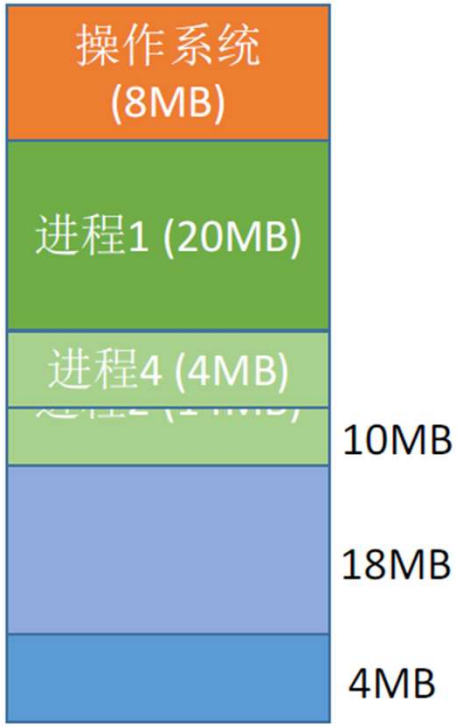
当某用户程序要装入内存时，由操作系统内核程序根据用户程序大小检索该表，从中找到一个能满足大小的、未分配的分区，将之分配给该程序，然后修改状态为“已分配”。

优点：实现简单，无外部碎片。
缺点：a. 当用户程序太大时，可能所有的分区都不能满足需求，此时不得不采用覆盖技术来解决，但这又会降低性能；b. 会产生内部碎片，内存利用率低。

内存（分区大小不等）

动态分区分配

动态分区分配又称为可变分区分配。这种分配方式不会预先划分内存分区，而是在进程装入内存时，根据进程的大小动态地建立分区，并使分区的大小正好适合进程的需要。因此系统分区的大小和数目是可变的。（eg：假设某计算机内存大小为 64MB，系统区 8MB，用户区共 56 MB…）

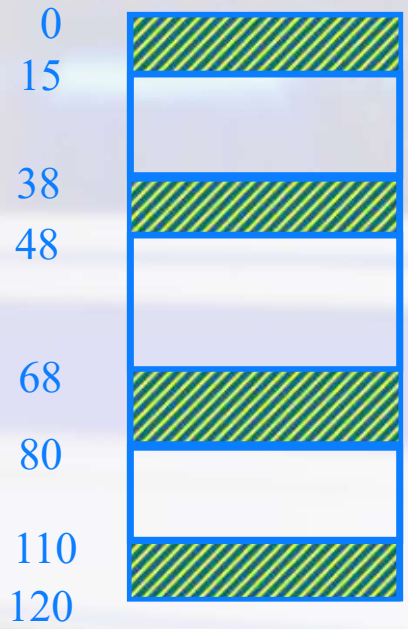


1. 系统要用什么样的数据结构记录内存的使用情况？
2. 当很多个空闲分区都能满足需求时，应该选择哪个分区进行分配？
3. 如何进行分区的分配与回收操作？

分区存储的管理方案

3 可变分区

- 内存不预先划分好
- 根据需求和内存空间来分配



分区存储的管理方案

3 可变分区

空闲区表

始址	长度	标志
15	23	未分配
48	20	未分配
80	30	未分配

已分配区表

始址	长度	标志
0	15	J1
38	10	J2
68	12	J3
110	10	J4
		空

空闲分区表

分区存储的管理方案

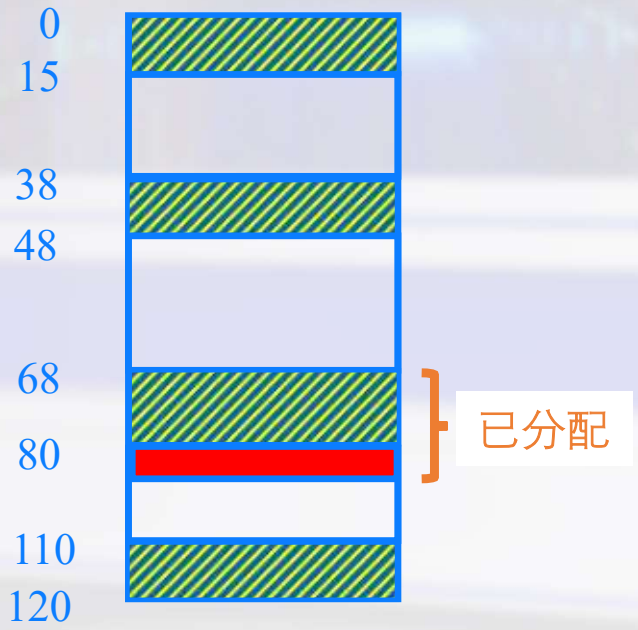
3 可变分区

空闲区表

始址	长度	标志
15	23	未分配
48	20	未分配
85	25	未分配

已分配区表

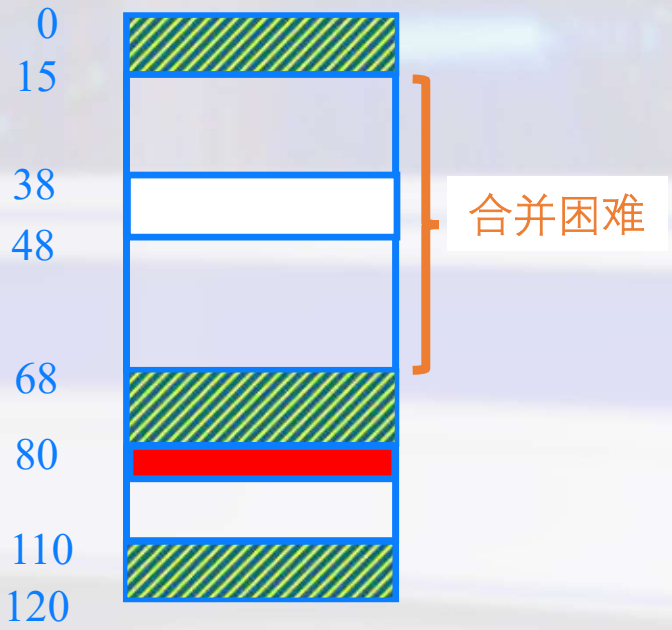
始址	长度	标志
0	15	J1
38	10	J2
68	12	J3
110	10	J4
80	5	J5



分区存储的管理方案

3 可变分区

- 内存回收： 空间合并，修改管理表格
- 内存碎片： 反复分配回收后，产生很多不连续的小空闲块
- 系统中总的空闲空间能够容纳进程，但是每一个空闲块没法容纳进程，进程要被阻塞

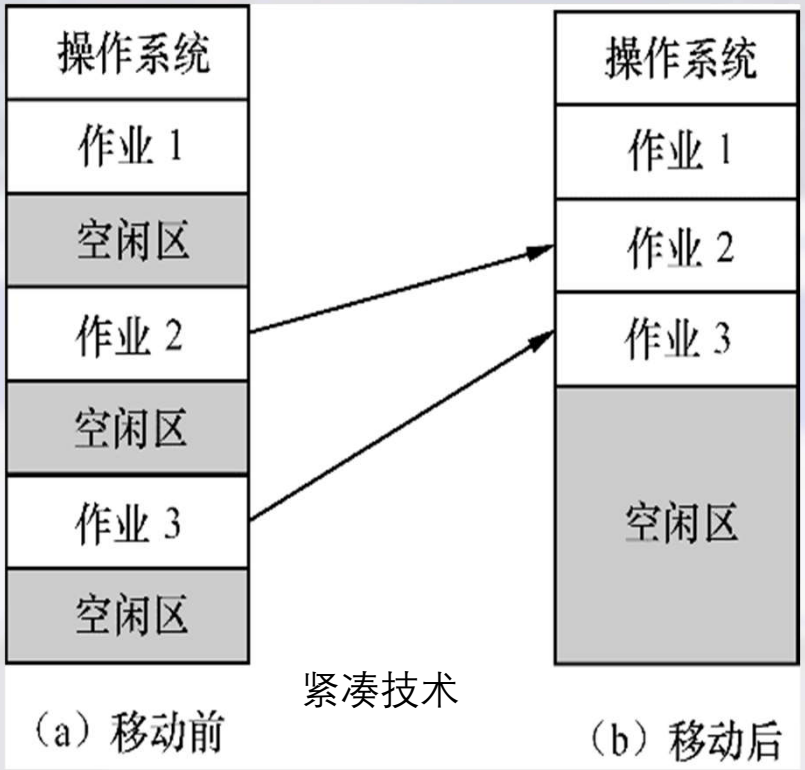


分区存储的管理方案

3 可变分区

把不连续的空闲块变成连续的空闲块

- 内存回收： 空间合并，修改管理表格
- 内存碎片：反复分配回收后，产生很多不连续的小空闲块



存储管理基本概念

本讲内容

1.计算机中的存储体系

2. 存储管理目标及任务

3. 连续存储区管理方案

4. 分区存储的管理方案

5. 存储覆盖与交换技术

存储覆盖与交换技术

1 引入原因

内存不够用

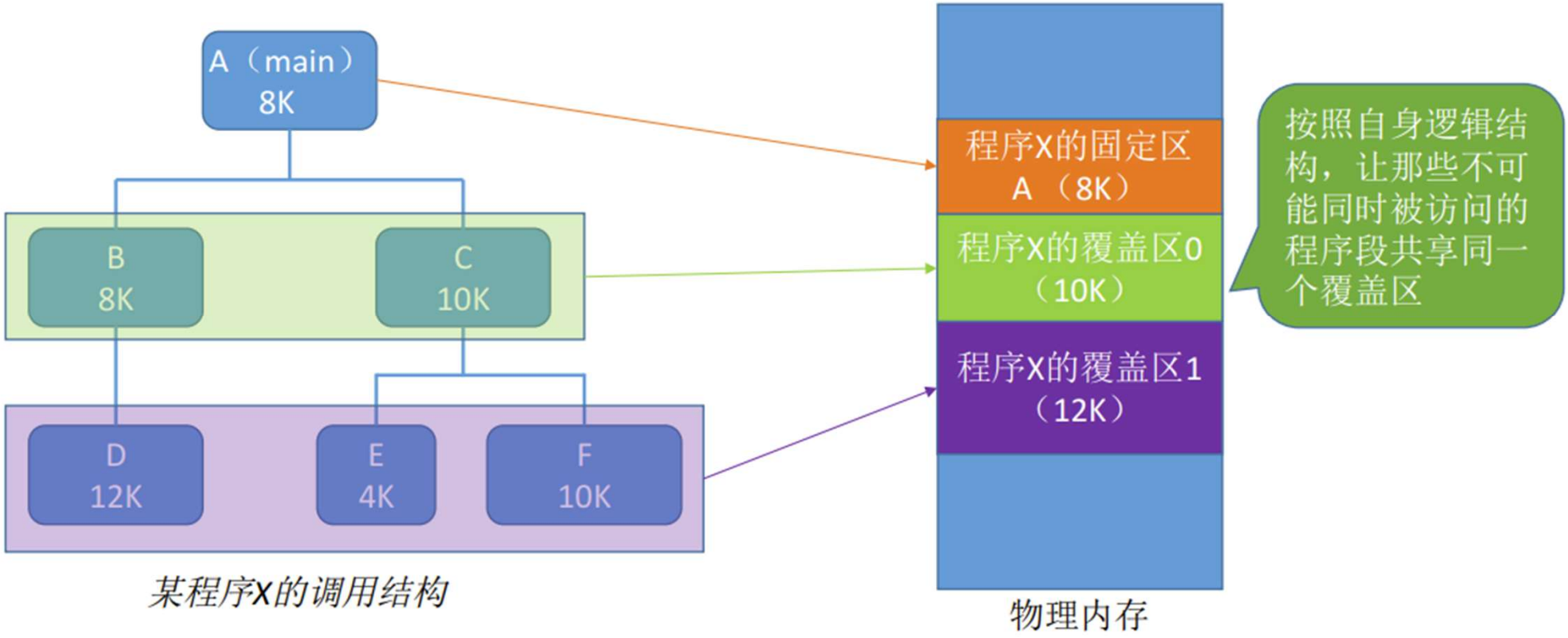
- 多道环境下扩充内存，解决在较小的存储空间中运行较大、较多进程时的矛盾
- 进程的程序和数据主要放在外存，需要执行的部分放在内存，内外存之间进行信息交换

存储覆盖与交换技术

2 覆盖技术

- **同一个进程内部不同程序段，分时共享并覆盖前面的数据**
- 进程若干程序段、数据段等共享同一存储空间
- 要求模块之间有明确调用结构，并向系统指明

覆盖技术



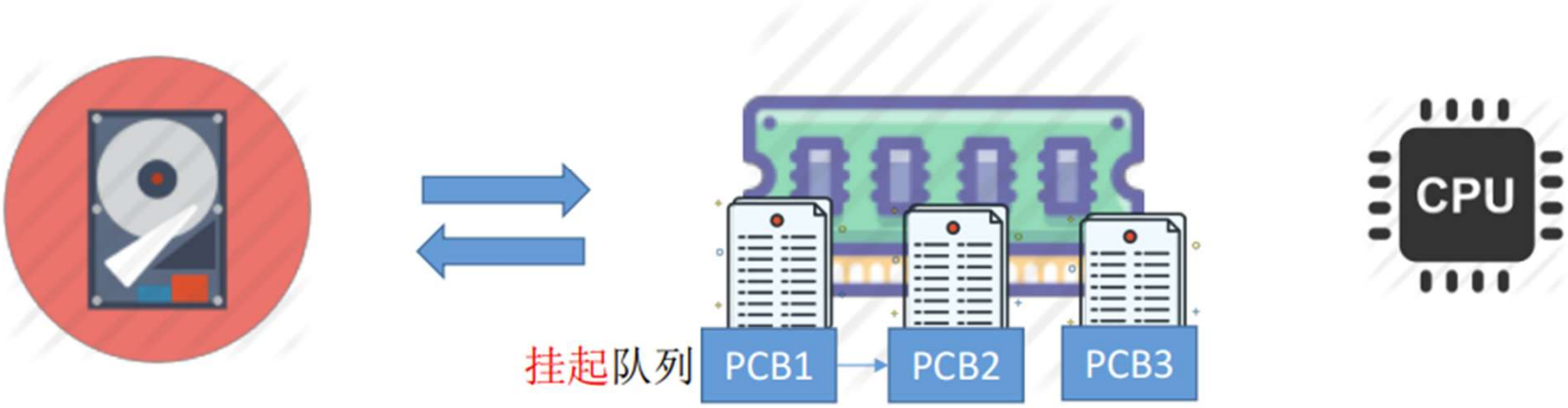
必须由程序员声明覆盖结构，操作系统完成自动覆盖。缺点：对用户不透明，增加了用户编程负担。
覆盖技术只用于早期的操作系统中，现在已成为历史。

---

8 + 10 + 12 → 8 + 8 + 10 + 12 + 4 + 10

交换技术

交换（对换）技术的设计思想：内存空间紧张时，系统将内存中某些进程暂时换出外存，把外存中某些已具备运行条件的进程换入内存（进程在内存与磁盘间动态调度）

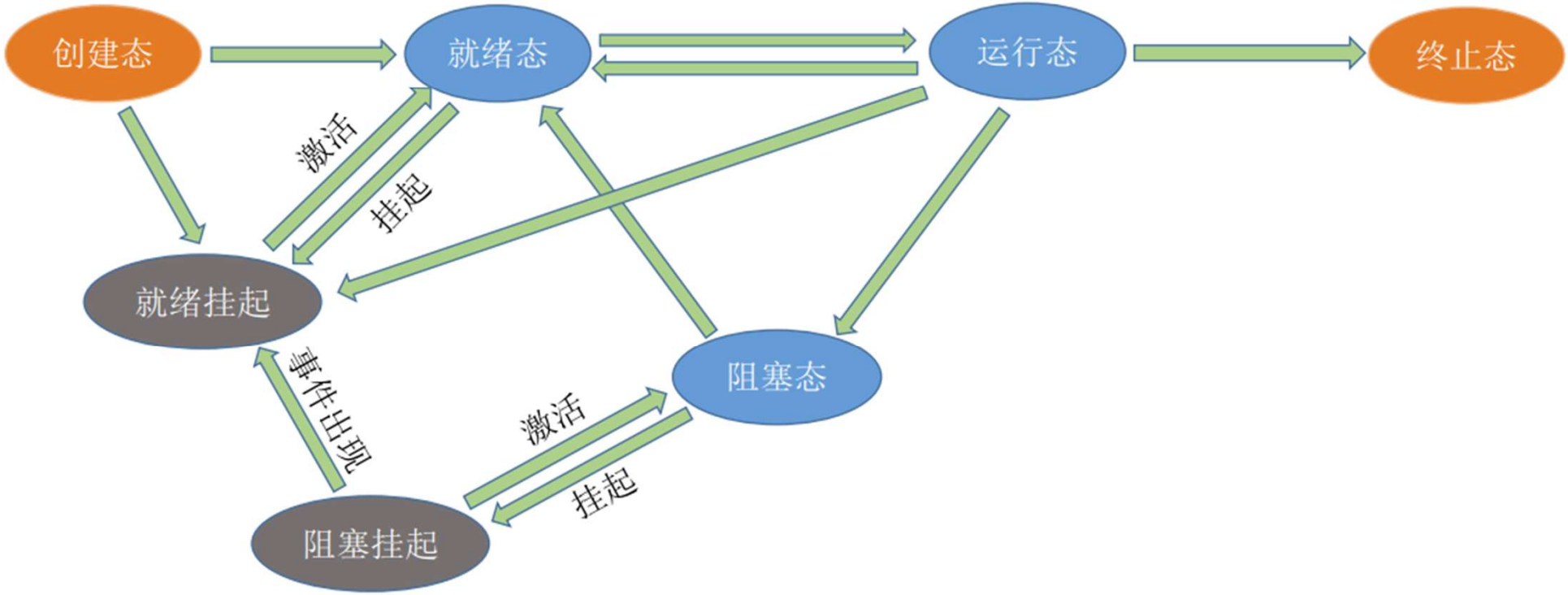


中级调度（内存调度），就是要决定将哪个处于挂起状态的进程重新调入内存。

交换技术

交换（对换）技术的设计思想：内存空间紧张时，系统将内存中某些进程暂时换出外存，把外存中某些已具备运行条件的进程换入内存（进程在内存与磁盘间动态调度）

暂时换出外存等待的进程状态为挂起状态（挂起态，suspend）
挂起态又可以进一步细分为就绪挂起、阻塞挂起两种状态



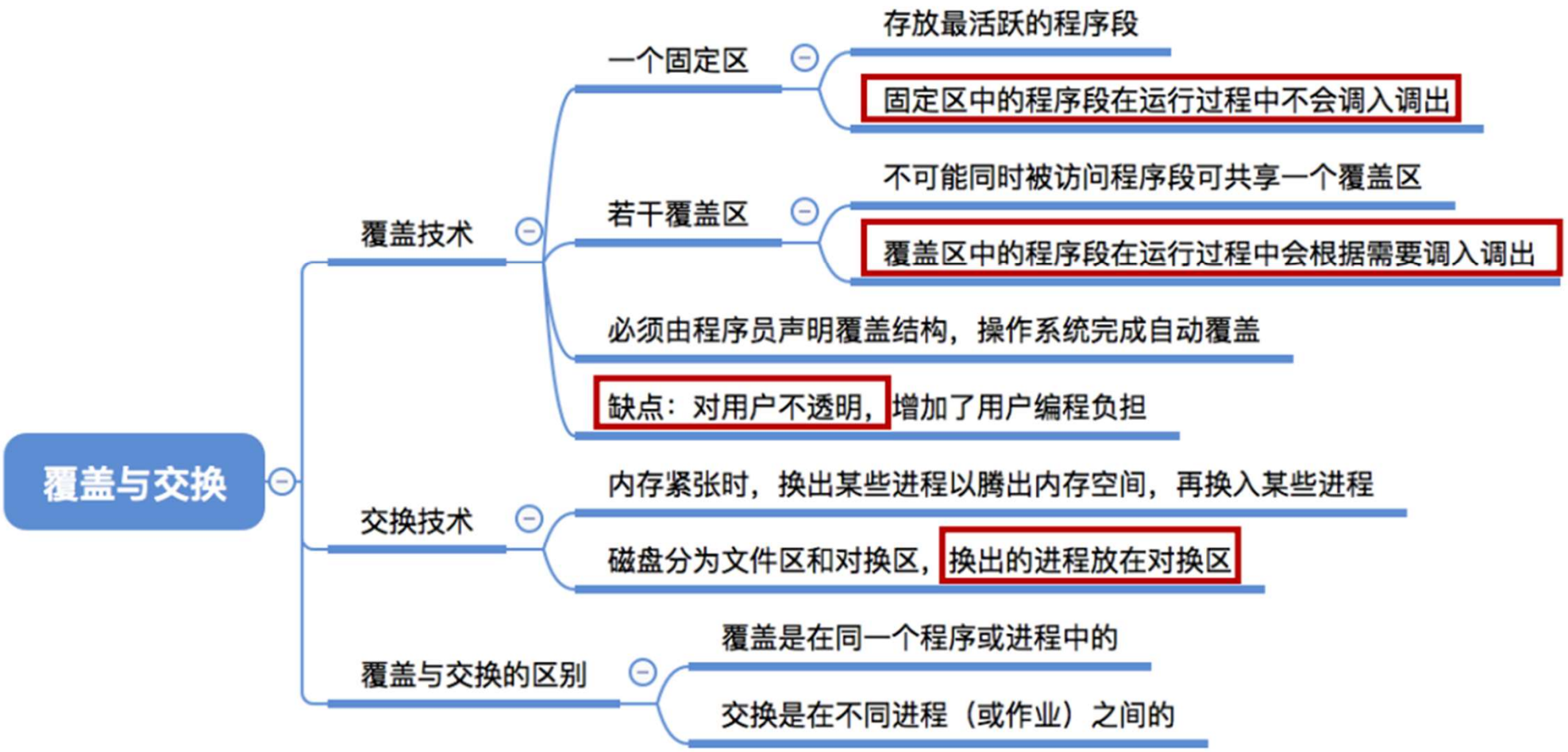
存储覆盖与交换技术

3 交换技术

- 多个进程之间分时共享内存储空间
- 以进程为单位，在内、外存之间动态调度
- 需要在辅存设置一个盘交换区 (swap分区)

用一个固定区域来临时存放进程，只有当进程执行的时候再调入内存

知识回顾与重要考点


覆盖与交换
覆盖技术
一个固定区
存放最活跃的程序段
固定区中的程序段在运行过程中不会调入调出
若干覆盖区
不可能同时被访问程序段可共享一个覆盖区
覆盖区中的程序段在运行过程中会根据需要调入调出
必须由程序员声明覆盖结构，操作系统完成自动覆盖
缺点：对用户不透明，增加了用户编程负担
交换技术
内存紧张时，换出某些进程以腾出内存空间，再换入某些进程
磁盘分为文件区和对换区，换出的进程放在对换区
覆盖与交换的区别
覆盖是在同一个程序或进程中的
交换是在不同进程（或作业）之间的

存储管理基本概念

Linux
Android
Linux
OpenStack
Mac OS
Windows