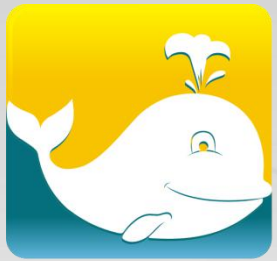
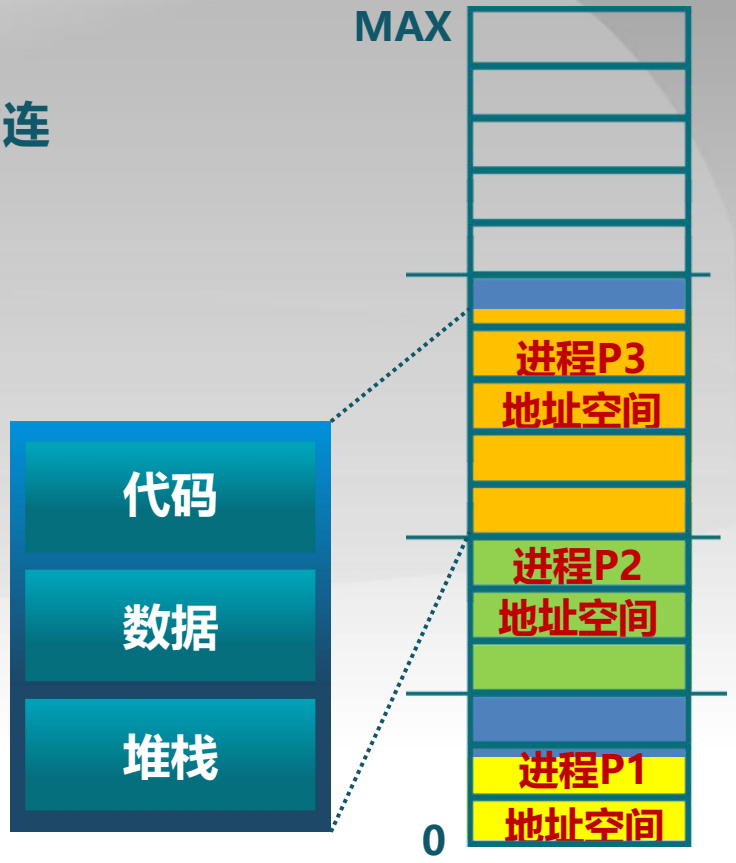


操作系统

Operating System

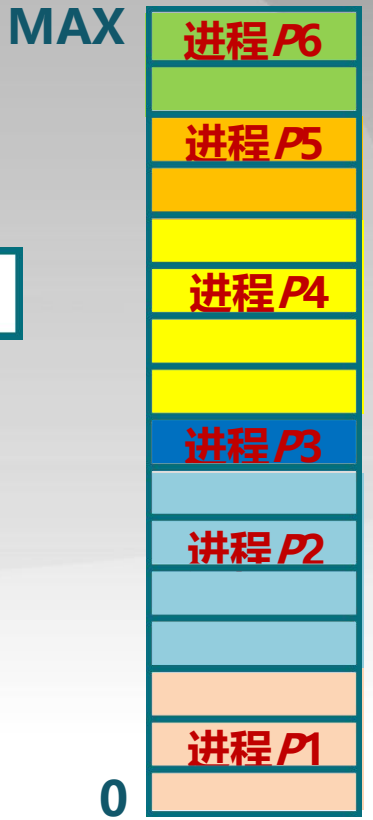
连续内存分配和内存碎片

- 连续内存分配
  - 给进程分配一块不小于指定大小的连续的物理内存区域
- 内存碎片
  - 空闲内存不能被利用
- 外部碎片
  - 分配单元之间的未被使用内存
- 内部碎片
  - 分配单元内部的未被使用内存
  - 取决于分配单元大小是否要取整



连续内存分配：动态分区分配

- 动态分区分配
  - 当程序被加载执行时，分配一个进程指定大小可变的分区(块、内存块)
  - 分区的地址是连续的
- 操作系统需要维护的数据结构
  - 所有进程的已分配分区
  - 空闲分区(Empty-blocks)
- 动态分区分配策略
  - 最先匹配(First-fit)
  - 最佳匹配(Best-fit)
  - 最差匹配(Worst-fit)

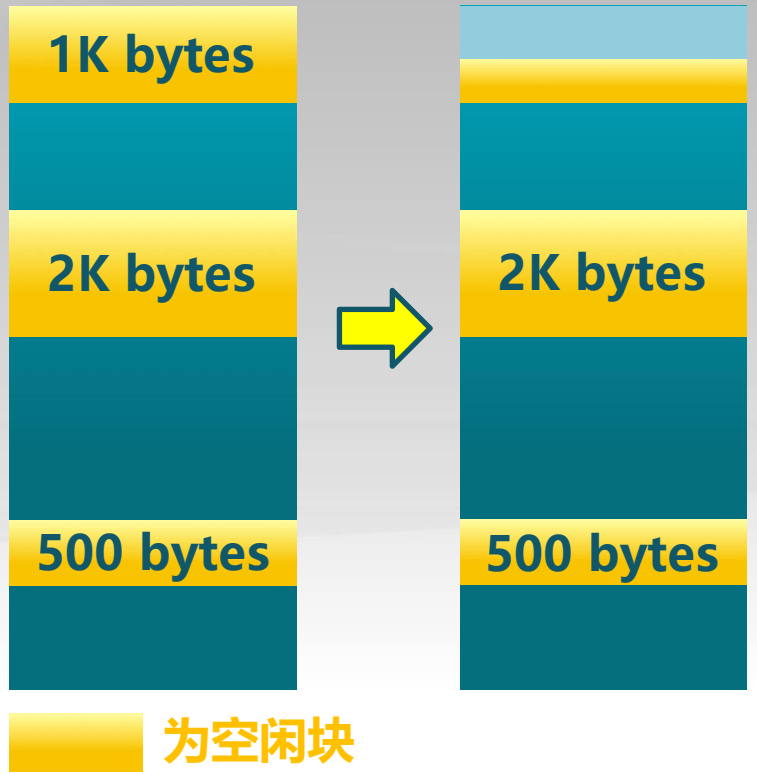


最先匹配(First Fit Allocation)策略

思路：

分配n个字节，使用第一个可用的空间比n大的空闲块。

示例：
分配400字节，使用第一个1KB的空闲块。



最先匹配(First Fit Allocation)策略

- 原理 & 实现
  - 空闲分区列表按地址顺序排序
  - 分配过程时，搜索一个合适的分区
  - 释放分区时，检查是否可与临近的空闲分区合并
- 优点
  - 简单
  - 在高地址空间有大块的空闲分区
- 缺点
  - 外部碎片
  - 分配大块时较慢

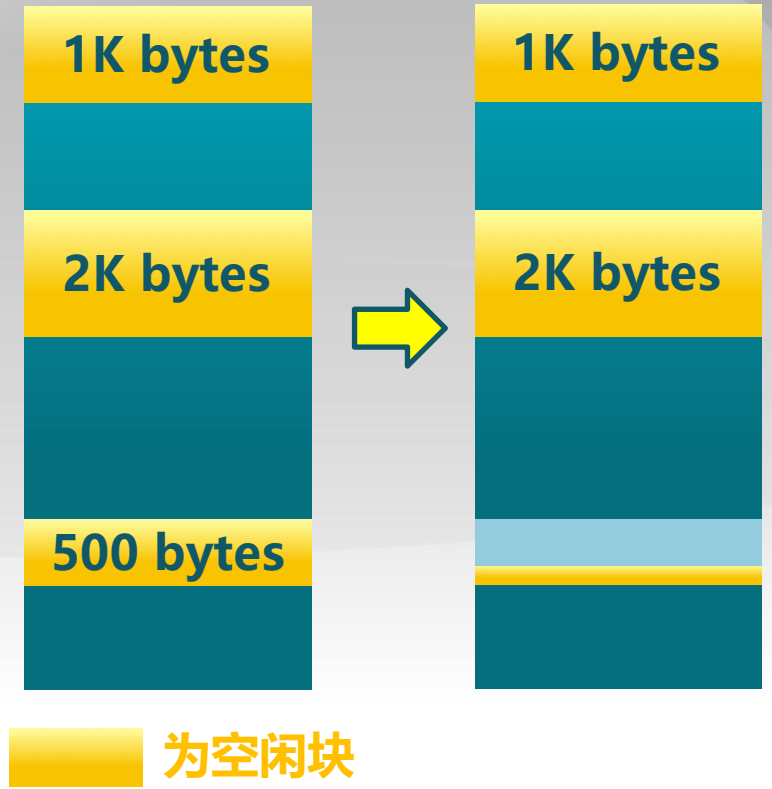
最佳匹配(Best Fit Allocation)策略

思路：

分配n字节分区时， 查找并使用不小于n的最小空闲分区

示例：

分配400字节， 使用第3个空闲块(最小)



最佳匹配(Best Fit Allocation)策略

- 原理 & 实现
  - 空闲分区列表按照大小排序
  - 分配时，查找一个合适的分区
  - 释放时，查找并且合并临近的空闲分区（如果找到）
- 优点
  - 大部分分配的尺寸较小时，效果很好
    - 可避免大的空闲分区被拆分
    - 可减小外部碎片的大小
    - 相对简单
- 缺点
  - 外部碎片
  - 释放分区较慢
  - 容易产生很多无用的小碎片

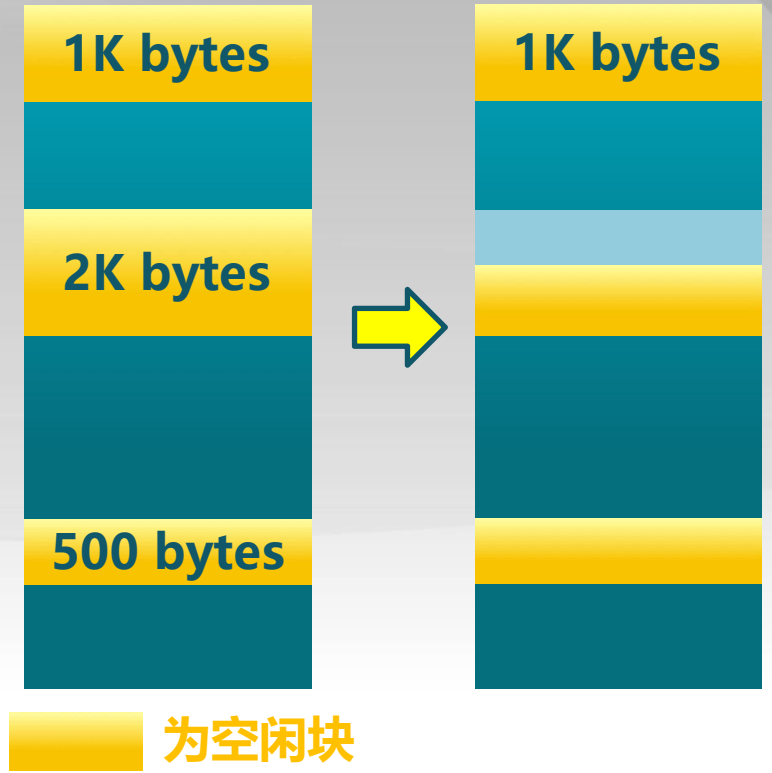
最差匹配(Worst Fit Allocation)策略

思路：

分配n字节，使用尺寸不小于n的最大空闲分区

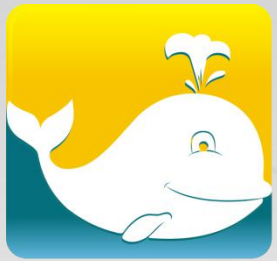
示例：

分配400字节，使用第2个空闲块（最大）



最差匹配(Worst Fit Allocation)策略

- 原理 & 实现
  - 空闲分区列表按由大到小排序
  - 分配时，选最大的分区
  - 释放时，检查是否可与临近的空闲分区合并，进行可能的合并，并调整空闲分区列表顺序
- 优点
  - 中等大小的分配较多时，效果最好
  - 避免出现太多的小碎片
- 缺点
  - 释放分区较慢
  - 外部碎片
  - 容易破坏大的空闲分区，因此后续难以分配大的分区



操作系统

Operating System