操作系统原理第四章作业

姓名：马福泉　学号：23336179　截止日期：2025 年 5 月 15 日

完成日期：2025 年 5 月 9 日

Question 1：8.1 试说明内部碎片与外部碎片的区别。

Answer 1：

(1) 内碎片：是已经被分配出去（能明确指出属于哪个进程）却不能被利用的内存空间，占有这些区域或页面的进程并不使用这个存储块。而在进程占有这块存储块时，系统无法利用它。直到进程释放它，或进程结束时，系统才有可能利用这个存储块

(2) 外碎片：是还没有被分配出去（不属于任何进程），但由于太小了无法分配给申请内存空间的新进程的内存空闲区域，这些存储块的总和可以满足当前申请的长度要求，但是由于它们的地址不连续或其他原因，使得系统无法满足当前申请

Question 2：8.2 考虑下面生成二进制的过程。编译器被用来生成单个单元的目标代码，链接器被用来将多个目标单元合并为一个程序二进制。链接器如何改变指令和数据到内存地址的绑定?需要将什么信息从编译器传递给链接器，以协助完成链接器的内存绑定任务?

Answer 2：

(1) 链接器通过修改目标代码中的地址引用，将指令和数据绑定到最终的内存地址。链接器合并各目标文件的代码段（.text）和数据段（.data），为它们分配运行时内存地址，并根据重定位表修正指令和数据中的符号引用，如函数调用、全局变量地址，替换为最终地址，以确保程序在运行时能够正确访问指令和数据。

(2)

符号表　（函数/变量名、地址、类型）；

重定位表　（需修正的指令位置及关联符号）；

段信息　（大小、对齐要求）

Question 3：8.3 如果有内存块 100KB、500KB、200KB、300KB 和 600KB(按顺序)，首次适应算法、最佳适应算法、最差适应算法各自将怎样放置大小分别为 212KB、417KB、112KB 和 426KB(按顺序)的进程?哪一种算法的内存利用率最高?

Answer 3：

(1) 首次适应算法会从头开始搜索内存块，找到第一个足够大的内存块来放置进程。

212KB 的进程：找到第一个足够大的内存块是 500KB，放置后剩余 288KB。

417KB 的进程：找到第一个足够大的内存块是 600KB，放置后剩余 183KB。

112KB 的进程：找到第一个足够大的内存块是 200KB，放置后剩余 88KB。

426KB 的进程：找到第一个足够大的内存块是 300KB，但不够大，继续找下一个，发现没有足够大的内存块，所以无法放置。

(2) 最佳适应算法会找到最接近进程大小的内存块来放置进程。

212KB 的进程：找到最接近的内存块是 300KB，放置后剩余 88KB。

417KB 的进程：找到最接近的内存块是 600KB，放置后剩余 183KB。

112KB 的进程：找到最接近的内存块是 200KB，放置后剩余 88KB。

426KB 的进程：找到最接近的内存块是 500KB，放置后剩余 74KB。

(3) 最差适应算法会找到最大的内存块来放置进程。

212KB 的进程：找到最大的内存块是 600KB，放置后剩余 388KB。

417KB 的进程：找到最大的内存块是 500KB，放置后剩余 83KB。

112KB 的进程：找到最大的内存块是 300KB，放置后剩余 188KB。

426KB 的进程：找到最大的内存块是 200KB，但不够大，继续找下一个，发现没有足够大的内存块，所以无法放置。

(4) 内存利用率

首次适应算法：成功放置了 3 个进程，剩余内存为 288KB+183KB+88KB=559KB，总内存为 1700KB，利用率为(1700-559)/1700=67.12%。

最佳适应算法:成功放置了 4 个进程,剩余内存为 88KB+183KB+88KB+74KB=433KB，总内存为 1700KB，利用率为(1700-433)/1700=74.53%。

最差适应算法：成功放置了 2 个进程，剩余内存为 388KB+83KB+188KB=659KB，总内存为 1700KB，利用率为(1700-659)/1700=61.24%。

综上，最佳适应算法的内存利用率最高。

Question 4：8.8 许多系统中，程序二进制常见的结构如下:代码从一个小的固定虚拟地址(如 0)开始保存。代码段后是用来保存程序变量的数据段，当程序开始执行时，在虚拟地址空间的另一端被分配，并被允许向更低的虚拟地址方向递增。在下列方案中，采用上述结构有何意义?

a.连续内存分配

b.纯分段

c.纯分页

Answer 4：

(1) 连续内存分配

便于操作系统预留连续空间，动态分配内存，减少外部碎片。

程序加载时只需分配两块连续区域（代码+数据），管理简单。

(2) 纯分段

代码段和数据段作为独立段，可设置不同权限。

高地址数据段与低地址代码段隔离，防止越界访问，如缓冲区溢出难以覆盖代码。

(3) 纯分页

固定代码段起始地址简化页表项计算，减少 TLB 的失效。

数据段向低地址增长时，页分配仍可保持局部性，减少缺页中断频率。

Question 5：8.9 假设一个将页表存放在内存的分页系统:

a. 如果一次内存访问需 200ns，访问一页内存要用多长时间?

b.如果加入 TLB，并且 75%的页表引用发生在 TLB,内存有效访问时间是多少?(假设在 TLB 中查找页表项占用零时间，如果页表项在其中。)

Answer 5：

a. 访问一页内存的时间

页表存放在内存，访问页表需要 1 次内存访问 （查页表）。

访问实际数据需要另 1 次内存访问 （访问目标页）。

总时间：2 × 200ns = 400ns。

b. 加入 TLB 后的有效访问时间

TLB 命中率 = 75%（TLB 命中时无需访问页表，直接访问数据）。

TLB 未命中率 = 25%（仍需查页表 + 访问数据）。

有效访问时间：(0.75×200ns)+(0.25×400ns)=150ns+100ns=250ns

Question 6：

8.12 假设有下面的段表：

段	基地址	长度
0	219	600
1	2300	14
2	90	100
3	1327	580
4	1952	96

下面逻辑地址的物理地址是多少？

a. 0430
b. 110
c. 2500
d. 3400
e. 4122

Answer 6：

a. 0430：段号为 0，段内偏移为 430。基地址为 219，长度为 600。
物理地址 = 基地址 + 段内偏移 = 219 + 430 = 649
b. 110：段号为 1，段内偏移为 10。基地址为 2300，长度为 14。
物理地址 = 基地址 + 段内偏移 = 2300 + 10 = 2310
c. 2500：段号为 2，段内偏移为 500。基地址为 90，长度为 100。
段内偏移超出长度，无效地址。
d. 3400：段号为 3，段内偏移为 400。基地址为 1327 长度为 580
物理地址 = 基地址 + 段内偏移 = 1327 + 400 = 1727
e. 4122：段号为 4，段内偏移为 122。基地址为 1952，长度为 96。|
段内偏移超出长度，无效地址