WEEK 9 题目

1. 课后quiz

Question 2

Incorrect

Mark 0.00 out of 1.00

Flag question

Consider a disk queue with requests for I/O to blocks on cylinders

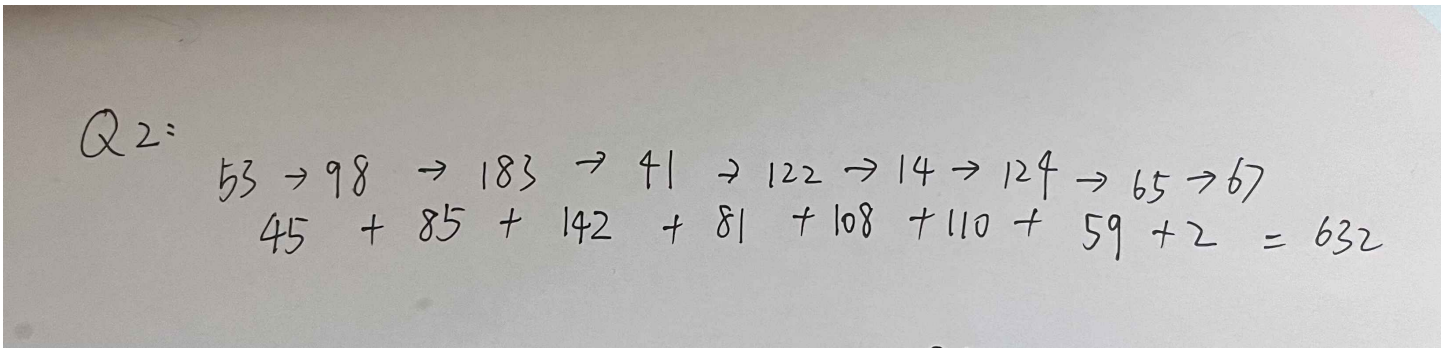
98, 183, 41, 122, 14, 124, 65, 67.

The **First Come First Served scheduling algorithm** is used. The head is initially at cylinder number **53** moving towards larger cylinder numbers on its servicing pass. The cylinders are numbered from 0 to 199.

The total head movement (in number of cylinders) incurred while servicing these requests is:

Answer: 98, 183, 41, 122, 14, 124, 65, 67 ✖

The correct answer is: 632



Question 3

Incorrect

Mark 0.00 out of 1.00

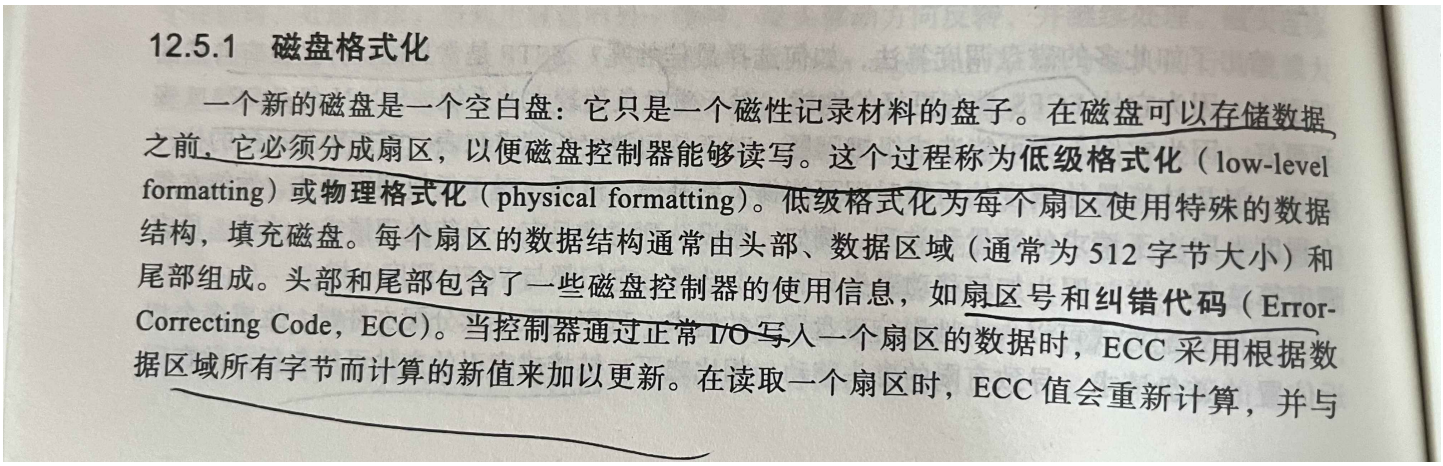
Flag question

The process of dividing a disk into sectors that the disk controller can read and write, before a disk can store data is known as _____

- low-level formatting
- swap space creation
- none of the mentioned
- partitioning ✖

Your answer is incorrect.

The correct answer is:
low-level formatting



12.5.1 磁盘格式化

一个新的磁盘是一个空白盘：它只是一个磁性记录材料的盘子。在磁盘可以存储数据之前，它必须分成扇区，以便磁盘控制器能够读写。这个过程称为**低级格式化**（low-level formatting）或**物理格式化**（physical formatting）。低级格式化为每个扇区使用特殊的数据结构，填充磁盘。每个扇区的数据结构通常由头部、数据区域（通常为 512 字节大小）和尾部组成。头部和尾部包含了一些磁盘控制器的使用信息，如扇区号和**纠错代码**（Error-Correcting Code，ECC）。当控制器通过正常 I/O 写入一个扇区的数据时，ECC 采用根据数据区域所有字节而计算的新值来加以更新。在读取一个扇区时，ECC 值会重新计算，并与

Question 13

Correct

Mark 1.00 out of 1.00

Flag question

The two steps the operating system takes to use a disk to hold its files are _____ and _____

- caching & logical formatting
- partitioning & logical formatting ✔
- swap space creation & caching
- logical formatting & swap space creation

Your answer is correct.

The correct answer is:
partitioning & logical formatting

Question 15

Incorrect

Mark 0.00 out of 1.00

Flag question

Consider a disk queue with requests for I/O to blocks on cylinders

98, 183, 41, 122, 14, 124, 65, 67.

The **SSTF scheduling algorithm** is used. The head is initially at cylinder number **53** moving towards larger cylinder numbers on its servicing pass. The cylinders are numbered from 0 to 199.

The total head movement (in number of cylinders) incurred while servicing these requests is 41,65,67,14,9 ✖ .

The distance from 53 to 65 and to 41 is the same 12. So, there are 2 ways:

1. **53-65-67-41-14-98-122-124-183** , the total head movement = 236
2. **53-41-65-67-98-122-124-183-14**, the total head movement = 323

The first option provides less seek time.

The correct answer is: 236

Question 4

Correct

Mark 1.00 out of 1.00

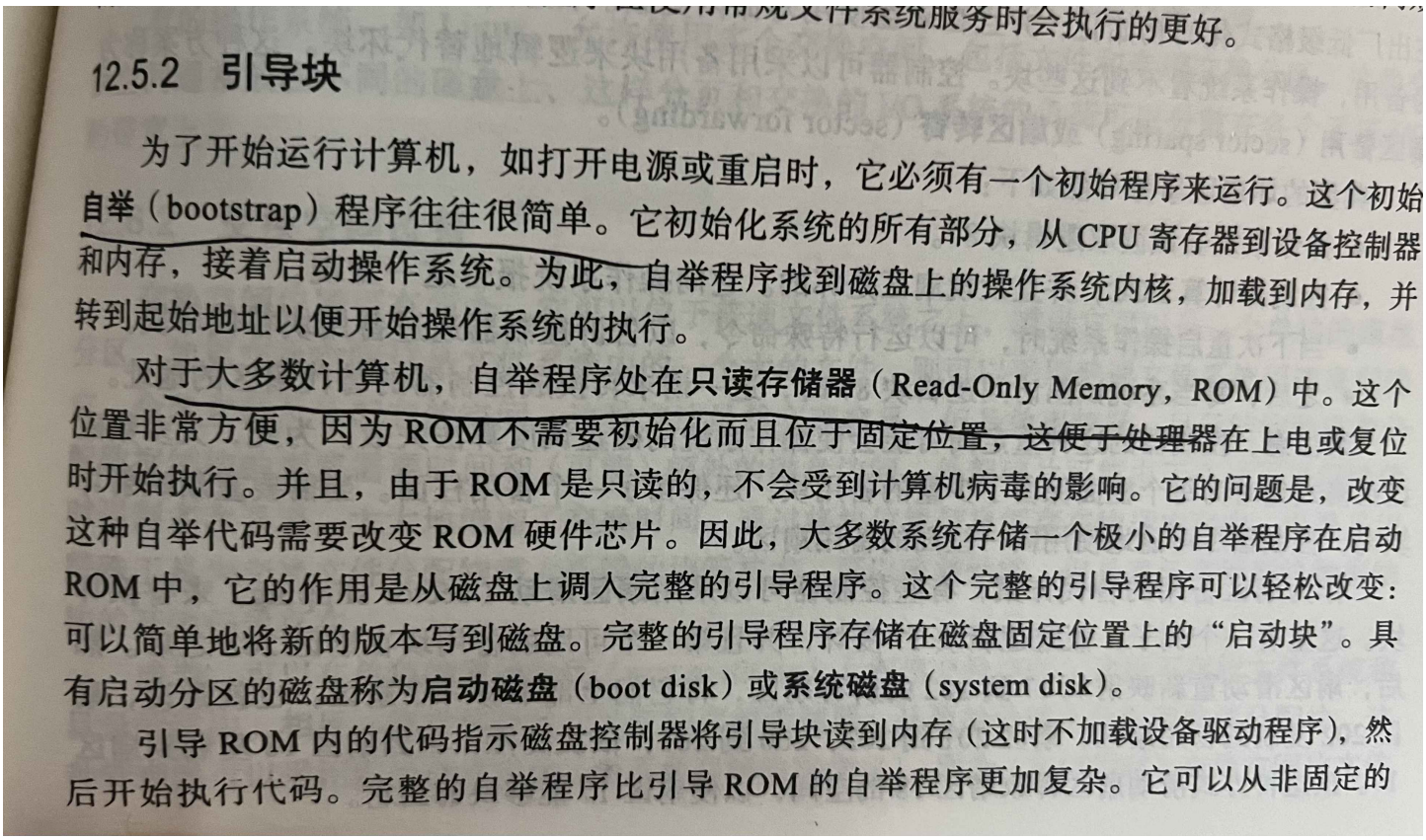
Flag question

For most computers, the bootstrap is stored in _____. A bootstrap is the program that initializes the operating system (OS) during startup.

- ROM ✔ The bootstrap routine is stored in ROM and is capable of reading from backing store the complete operating system, which is loaded into the empty memory.
- RAM
- Tertiary storage
- Cache

Your answer is correct.

The correct answer is:
ROM



12.5.2 引导块

为了开始运行计算机，如打开电源或重启时，它必须有一个初始程序来运行。这个初始自举（bootstrap）程序往往很简单。它初始化系统的所有部分，从 CPU 寄存器到设备控制器和内存，接着启动操作系统。为此，自举程序找到磁盘上的操作系统内核，加载到内存，并转到起始地址以便开始操作系统的执行。

对于大多数计算机，自举程序处在**只读存储器**（Read-Only Memory，ROM）中。这个位置非常方便，因为 ROM 不需要初始化而且位于固定位置，这便于处理器在上电或复位时开始执行。并且，由于 ROM 是只读的，不会受到计算机病毒的影响。它的问题是，改变这种自举代码需要改变 ROM 硬件芯片。因此，大多数系统存储一个极小的自举程序在启动 ROM 中，它的作用是从磁盘上调入完整的引导程序。这个完整的引导程序可以轻松改变：可以简单地将新的版本写到磁盘。完整的引导程序存储在磁盘固定位置上的“启动块”。具有启动分区的磁盘称为**启动磁盘**（boot disk）或**系统磁盘**（system disk）。

引导 ROM 内的代码指示磁盘控制器将引导块读到内存（这时不加载设备驱动程序），然后开始执行代码。完整的自举程序比引导 ROM 的自举程序更加复杂。它可以从非固定的

Problem

Consider a system using multilevel paging scheme. The page size is 1 MB. The memory is byte addressable and virtual address is 64 bits long. The page table entry size is 4 bytes. Find:

1. How many levels of page table will be required?
2. Give the divided physical address and virtual address.

页表项大小是由物理地址的位数决定的，因为页表中的每个页表项存放的都是一个物理地址指示对应页在内存中的位置。所以页表项大小为4B = 4 x 8bit = 32bit

32位表物理地址 32-20=12 20位→页面大小 1MB = 2^{20}B 20位页内偏移量

前面12位表示页号 因此物理内存被分为 2^{12} 个帧

逻地：64bits 页面/帧大小：1MB=2^{20}B 页表项大小 4B

内存层页表中只有帧号

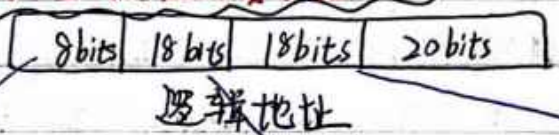
1. 帧数位：页表项大小 4B=4×8bit=32bits 所以帧数为32位（帧号）
2. 主存中帧的数量为 2^{32} frames 2^{32}个帧
3. 主存大小：总帧数×帧的大小 $=2^{32}\times 2^{20}=2^{52}$B 物理地址位数52位
4. 页内偏移量位数：页面大小：1MB=2^{20}B → 20位 页内偏移量
5. 进程大小 逻地64位 所以进程大小为 2^{64}B
6. 进程被分为多页：$2^{64}/2^{20}=2^{44}$页
7. 内页表大小：进程被分的页数×页表项大小 $=2^{44}\times 4B=2^{46}$B
8. 2^{46}B > 2^{20}B 页表要被分页
9. 内层表被分页数 $2^{46}/2^{20}=2^{26}$页
10. 内页表中每一页的页表项数（~~页表被分的块数~~）页面大小/页表项大小 = 1MB/4B = 2^{11}
11. 在内页表的一页中搜索一个条目所需位数：2^{18}页就需要18位
12. 外层页表-1：用于跟踪存储内层页表页的帧 外层页表-1的大小=外页表1的页数×页表项大小 = 内层页表被分页数×页表项大小 $=2^{26}\times 4B=2^{28}$B

13. 2^{28}B = 256MB > 1MB 外层页表-1的大小大于1帧，因此外层页表-1要被再分页
14. 外层页表-1被分页数 256MB/1MB = 256页 → 此时外层页表-1被分为256页存在不同的帧中
15. 外层页表-1的每一页页表项数 = 页面大小/页表项大小 = 1MB/4B = 2^{18}页
16. 在外层页表-1的一页中搜索一个条目所需位数：外层页表-1包含中的一页包含2^{18}页 因此要18位
17. 外层页表-2的大小 = 外层页表2的页数×页表项大小 = 外层页表-1被分的页数×表项大小 = 256×4B = 2^{10}B = 1KB < 1MB 不用再分层

PTBR 页表基寄存器指向外层页表-2的基址

18. 在外层页表2中搜索一个条目所需位数：外层表2包含256项（外层页表1被映射）

256 = 2^8项 因此要8位



逻辑地址

前8位用于搜索外层页表2

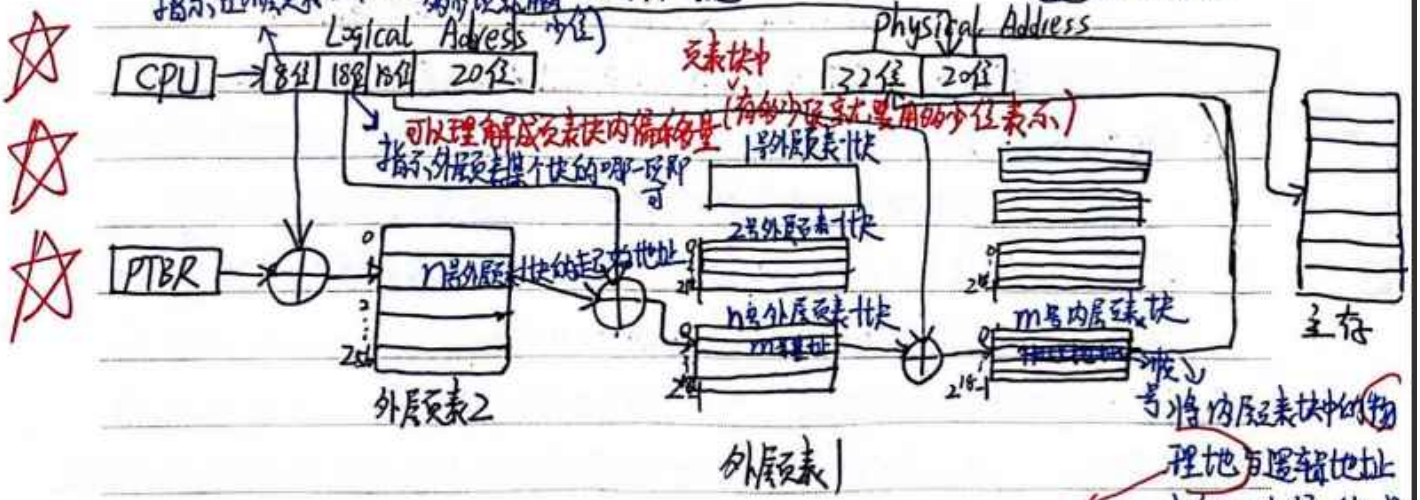
外层页表2只有256项且存在一个块中所以只要8位来找这256项中的一项

这18位用于搜索外层页表1

外层页表1的每一块中有2^{18}项（页表块号是隐含的）只要知道要找该页表块中的哪一项就行了所以要18位来表示这2^{18}项中的哪一项

这18位用于搜索内层页表

内层页表中每一页的块号是隐含的，所以也只用知道要找这个页表块中的哪一项就行了，所以也是2^{18}→18位



将内层页表块中的物理地址与逻辑地址中的页内偏移量拼接得到目标元在主存内的地址

这里的物理地址是对应帧号（也就是帧号）

有多少帧就用多少位表示

（直接将其与页内偏移量拼接得到物理地址）