简答题部分

1.2 题→3-6 章
3.4 题→7-8 章
5.6 题→9-11 章

第 1, 2 道简答题

1. 什么是进程?

   定义：(有很多种定义)
   一个正在执行中的程序。
   一个正在计算机上执行的程序实例。
   能分配给处理器并由处理器执行的实体。
   一个具有以下特征的活动单元：一组指令序列的执行、一个当前状态和相关的系统资源集。

2. 五状态，七状态的图?

3. 为什么从二状态到五状态？为什么从五状态到七状态？与长/中/短程调度结合是什么样子?

   引入五状态图原因：有一部分进程执行 I/O 阻塞，如果不细分，可能造成无效调度，存在一些处于非运行态但已就绪等待执行的进程，同时还存在另外一些处于阻塞态等待 I/O 操作结束的进程。
   引入七状态图原因：处理器速度比 I/O 速度快的多，可能造成所有进程都在等待。

4. ~~进程创建的过程~~

5. 进程和线程的概念，区别是什么?

6. *线程的优点是什么？

7. ~~用户级线程和内核级线程的优缺点比较?~~

   ~~ULT，用户级线程 进程内线程的创建或终止，内核都是不知道的。内核是以进程为调度单位的，而且制定地为一个进程指定状态~~
   ~~长处：线程切换不须要内核态特权。能够有自己的调度算法，而不涉及改变操作系统的基本调度算法。缺点：一个应用程序中的一个线程堵塞，其余线程也会堵塞。~~
   ~~内核尽管把一个处理器分给一个进程，实际上是仅仅处理了一个线程~~
   ~~KLT，有关线程的管理的全部工作都是由内核完毕的，应用程序部分没有线程库等线程管理的代码，调度也是内核基于线程完毕的。长处：能够把一个进程的多个线程调度到多个处理器里一个线程被堵塞，处理器能够调度到还有一个线程。缺点：切换须要内核状态的改变~~

8. *死锁的条件

9. 处理死锁的方法?

   ① 死锁的防止。系统按预定的策略为进程分配资源，这些分配策略能使死锁的四个必要条件之中的一个不成立。从而使系统不产生死锁。
   ②死锁的避免。系统动态地测试资源分配情况，仅当能确保系统安全时才给进程分配资源。
   ③死锁的检测。对资源的申请和分配不加限制，仅仅要有剩余的资源就呆把资源分配给申请者，操作系统要定时推断系统是否出现了死锁，当有死锁发生时设法解除死锁。

10. 死锁的预防？破坏形成条件
11. 死锁的避免？
12. 死锁的检测？
13. 哲学家就餐为什么会发生死锁？解决方式？

第 3, 4 道简答题

1. ~~内存的分区固定分区和动态分区的特点？优缺点？~~
2. ~~三种动态放置算法？~~
3. 什么是内存的重定位？

   重定位：将程序或数据的内存地址进行调整，以便它们能够在不同的内存地址上运行。

   可用空间通常被多个进程共享，不能确定程序在执行中会被放在内存中的什么位置。重定位将程序或数据在内存中的地址进行调整的过程，以便它们能够在不同的内存地址上运行。重定位的主要目的是使得程序能够在不同的内存区域中加载和执行，而不需要修改程序的源代码。
4. 为什么重定位？

   [1] 提高内存利用率：重定位允许进程动态分配到可用的内存区域

   [2] 确保进程可以在新位置继续正常运行，而不受原始内存地址的限制。

   [3] 与物理内存位置无关：不需要关心程序在执行时会被加载到内存的哪个位置

   [4] 逻辑地址到物理地址的转换：重定位动态地将程序中的逻辑地址转换为实际的物理地址

   [5] 支持多任务和系统性能优化
5. 分页？

   每个进程被划分成多个小的，大小固定的块，称为 page，内存被划分多个大小固定的块，称为 frame，frame 和 page 的大小相等。Page 和 frame 之间存在映射关系，操作系统可以将 page 映射到任何空闲的 frame 中
6. 分段？

   将程序按照逻辑划分成大小不一的段，不要求每个段的大小都相等。分段可以是不连续的。
7. 内部碎片，外部碎片？

   内部碎片：被装入数据小于分区大小，导致分区内部有空间浪费，这种现象被称为内部碎片

   外部碎片：由于程序不断地被加载和置换,内存中产生了大量不连续的空闲内存块，这些空闲内存块的总大小可能足够满足进程的内存请求，但由于它们不是连续的，因此无法满足请求。随时间的推移，内存中产生了越来越多的碎片，内存利用率随之下降的现象被称为外部碎片
8. ~~Local address？physical address？~~

   ~~逻辑地址：用户程序中使用的地址称为逻辑地址，逻辑地址是对内存的访问地址，而不是数据实际存储的位置。~~

   ~~物理地址：数据存储在内存中的实际的位置~~
9. 操作系统内存管理的设计？ [3]

   是否虚拟内存技术？分段分页或结合？内存管理算法？

10. 读取策略，置换策略，替换策略？

11. *替换策略几种算法？理想的不可实现？可实现的性能好坏排序

[1] 最佳置换策略 optimal; OPT：替换下次访问距当前时间最长的页
这个是不可实现的，操作系统不可能知道将来发生的事件

[2] 最近最少使用 Least Recently Used; LRU：置换距上次访问时间最长的页
需要为每个页标记上次的访问时间，开销较大　　性能最好（可实现）

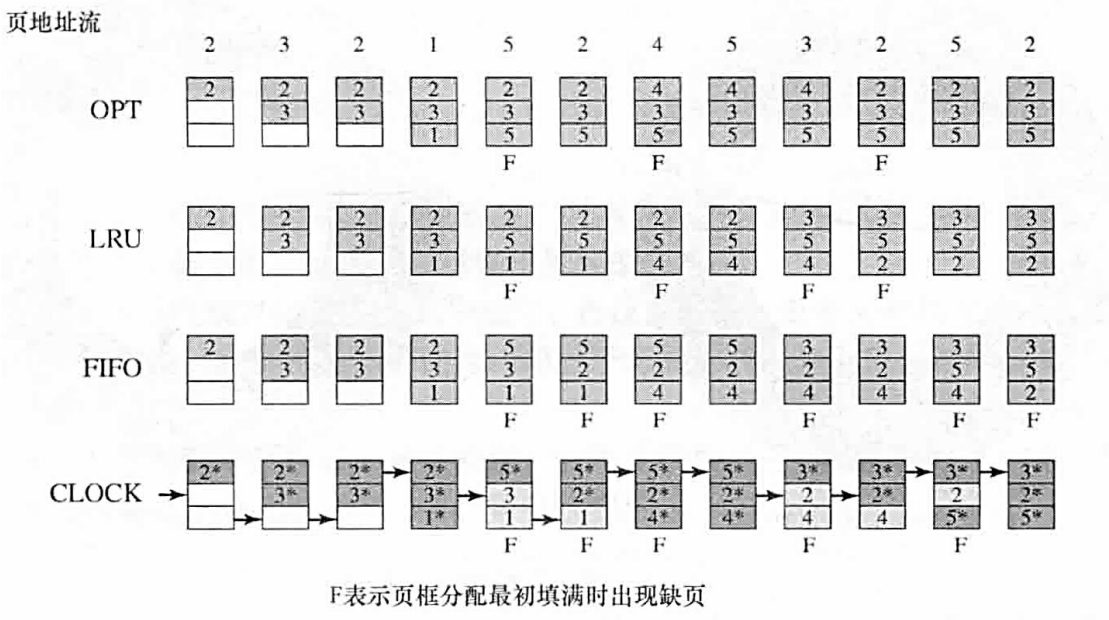
[3] 先进先出 FIFO：把分配给进程的页框视为循环缓冲区，按照轮转方式移动页。　　性能最差

[4] 时钟 Clock：页面首次加载到内存帧时，帧的使用位为 1，当一个页面被访问时，将其对应的使用位设置为 1。当发生缺页（Page Fault）且需要置换页面时检查指针指向的页面。如果其使用位为 1，清零并移动指针；如果其使用位为 0，选择该页面置换，并将新页面加载到该位置，同时将指针移动到下一位置。　　性能居中

注意指针的移动，不命中才移动，命中的话不移动

12. 访问轨迹？

不同替换策略的命中率 ratio 是多少？



13. 清除策略？

清除策略用于确定何时将一个被修改过的页写回到辅存

[1] 请求式清除（demand cleaning）：只要当一页被选择用于置换时才被写回辅存。

[2] 预约式清除（precleaning）：在需要用到它们所占据的页框之前就批量地写回辅存。

请求式清除与预清除的不同点：请求式清除的页面只有在被选择替换时才会被写出，而预清除在需要页面之前就被分批写出。

14. 加载控制？

加载控制策略：确定内存中的进程数目。控制驻留在内存中的进程数目不会过多也不会过少。

如果某一时刻驻留的进程太少，所有进程同时处于被阻塞的状态的可能性增

大，会有很多时间花费在交换上；如果驻留的进程太多，平均每个进程的驻留集的大小将不够用，频繁发生缺页中断，导致抖动

如何解决？[6]进程挂起

- 最低优先级进程
- 缺页中断进程
- 最后一个被激活的进程
- 驻留集最小的进程
- 最大空间的进程
- 具有最大剩余执行窗口的进程

15. 驻留集？

    决定为特定的进程分配多少内存(驻留集小则内存中进程多，驻留的进程少，会导致所有的进程都处于阻塞状态，交换；如果驻留的进程太多，抖动)

16. 什么是抖动 thrusting？

    操作系统读取一个块时，必须把另一块换出。如果一块正好在将要被用到之前换出，操作系统就不得不把它取回来。如果处理器的大部分时间都用于处理换入换出操作而不是执行指令，称为系统抖动。

第 5, 6 道简答题　　9-11 章

1. 处理器调度类型？

   通过满足系统目标（如满足响应时间，吞吐率，处理器分配和效率等）的方式，把进程分配到一个或多个处理器中执行。

2. *长程，中程，短程调度指的作用？

   长程调度：决定是否将进程加入到待执行的进程池中

   中程调度：决定时候将进程的部分或全部加载到主存中

   短程调度：决定哪一个就绪的进程将被处理器执行

3. *短程调度的准则是什么

   面向用户的准则和面向系统的准则

   面向用户：用户或进程感知到的系统行为相关（响应时间），用于交互式系统

   面向系统：重点是处理器使用的效果和效率（吞吐量）

4. 六种调度策略？FCFS，Round Robin，SPN，SRT，HRRN，FeedBack

   // 其中三种是抢占式的，三种是非抢占式的，那些容易实现，哪些不容易实现，原因是什么？Scheduling Policy

   FCFS First Come First Served 先来先服务：非抢占。每个进程准备好之后就加入到就绪队列中，先执行先到达的进程，再执行后到达的进程。对于长进程表现更好，偏向于处理器密集型，不利于 I/O 密集型

   Round-Robin 轮转：抢占。是基于时钟的抢占策略。通过设置时间片，每隔一段时间产生时钟中断，中断发生时，正在执行的进程被放到就绪队列中，然后基于 FCFS 选择下一个作业。轮转偏向分时系统和事务处理系统，不利于 I/O 系统

   Shortest Process Next 最短进程优先 SPN：非抢占。对进程的执行时间进行估计，下一次执行预计运行时间最短的进程，短进程会跳过长进程，提前加载到就绪队列开头。难点是需要估计剩余运行的时间，导致较大开销，并且会导致长进程的饥饿。

SRT Shortest Remaining Time 最短剩余时间：抢占。总是选择剩余时间最短的那个程序执行，当一个新作业加入到就绪队列中时，其剩余时间将与当前进程的剩余时间进行对比。如果新进程比当前运行的进程需要更少的时间就能完成，则将当前运行的进程挂起，运行新进程。难点还是需要估计进程预计服务时间。

HRRN Highest Response Ratio Next 最高响应比优先：非抢占。选择响应率 R 最大的下一个进程，需要估计预计服务时间。使用归一化周转时间（周转时间是等待时间+预计服务的时间），R=（等待时间+预计服务时间）/ 预计服务时间

FeedBack 反馈：抢占。不需要知道预计服务时间，关注的是已经执行了的时间

5. 抢占和非抢占

   抢占：当前正在运行的进程可能被操作系统中断，并转移到就绪态。

   非抢占：一旦进程处于运行状态，他就不断执行直到终止，或者因为等待 I/0，或者因而请求某些操作系统服务而阻塞自己。

6. 调度策略性能的评估？

   这玩意怎么考？

   ？？？？？？？

7. 归一化周转时间？

   Normalized turnaround time = 周转时间/服务时间

8. ~~执行 I/O 的三种技术？programming，interrupt，DMA~~

   ~~程序控制 I/O：处理器发出命令，进程进入忙等待，直到操作完成才能继续执行。~~

   ~~中断驱动 I/O：处理器发送命令，如果是阻塞的，执行的下一条指令来自操作系统；如果是非阻塞的，继续执行后续指令。~~

   ~~直接存储器访问 DMA：DMA 模块控制主存与 I/O 设备之间的数据交换~~

9. ***I/O 的发展历程？Evolution of the I/O function**

   CPU 逐渐从 I/O 任务中解脱出来（越来越多 I/O 功能在没有 CPU 参与下执行）

   （1）CPU 直接控制外围设备

   （2）增加控制器或 I/O 模块，CPU 使用非中断的程序控制 I/O

   （3）中断的控制器或 I/O 模块

   （4）I/O 模块通过 DMA 直接控制存储器

   （5）有单独处理器的 I/O 模块

   （6）I/O 模块有了自己的存储器，成为 I/O 处理器

10. 操作系统的设计目标？

    效率和通用性

11. **I/O 缓冲，原因？作用？**

    定义：在请求发出之前就执行输入传送，在请求发出一段时间之后开始执行输出传送。 预输入，缓输出

    原因：①进程必须等待 I/O 完成后才能继续进行 ②在进行 I/O 操作期间，进程的某些页必须保存在主存中

    两类 I/O 设备：

    （1）面向块：固定大小的块，一次传送一块。磁盘、USB 智能卡等

（2）面向流：以字节流的方式输入/输出数据。终端、打印机、鼠标等

作用：提高数据传输效率，减少设备操作次数，平衡 CPU 与 I/O 设备的速度差异。缓冲是提高操作系统效率和单个进程性能的一种方法。

12. 磁盘调度的几个时间，寻道时间，旋转延迟，传输时间，存取时间？

寻道时间：磁头定位到磁道所需的时间

旋转延迟：将磁盘待访问位置旋转到磁头可以读取的范围所需要的时间，磁盘转半圈的时间

传输时间：磁头通过相应的扇区读或写的时间

存取时间：到达读或写位置需要的时间，寻道+旋转+传输

排队延迟：当一个进程发出一个 I/O 请求时，须在队列中等待该设备可用

13. *磁盘调度算法？Disk scheduling policies？×6

Random 随机调度：随机顺序处理，性能最差，作为基准

FIFO 先进先出：按照顺序处理请求，对所有请求公平。如果有很多进程，性能可能接近随机调度。

Priority 优先级：按优先级调度，注重目标完成，不是优化磁盘使用，提供良好的交互响应时间。导致优先级较低的请求饥饿

LIFO 后进先出：适用于 transaction processing systems 事务处理系统，优先处理新到达的请求，可能导致饥饿

SSTF Shortest Service Time First 最短服务时间优先：选择磁头移动最短的 I/O 请求。始终选择最小寻道时间的请求。导致远端请求饥饿

SCAN 电梯算法：磁头仅朝一个方向移动，满足所有未完成的请求，直到到达该方向的最后轨道后向反向移动，按顺序处理所有请求。

C-SCAN circular SCAN：扫描仅限于一个方向，当到达一个方向的最后轨道后，磁头返回到磁盘的另一端，然后重新开始扫描。

N-step-SCAN N 步扫描：将磁盘请求队列分段为长度为 N 的子队列。一次处理一个子队列，对子队列使用 SCAN 算法。

FSCAN 两队列扫描：使用两个子队列，一个用于当前请求，一个用于新请求，交替处理两个队列，避免磁头粘滞

14. 文件系统层次结构？

用户→访问方法→逻辑 I/O→基本 I/O 管理程序→基本文件管理系统→设备驱动程序

15. 文件组织标准，五种？organization and access

（1）堆（pile）：最简单的文件组织形式。按照到达顺序收集数据，每条记录由一串数据组成。堆的目的仅仅是积累大量数据并保存记录，通过穷举查找进行数据访问

（2）顺序文件（Sequential File）：最常用的文件组织形式，所有记录长度相同，包含相同数量的字段，字段长度固定。按特定顺序排列

（3）索引顺序文件（the Indexed Sequential File）：保留了顺序文件的关键特征，新增了用于支持随机访问的文件索引 index 和溢出 overflow 文件

（4）索引文件（Indexed File）：只能通过索引来访问文件

（5）直接文件或散列文件（Direct / Hash）：直接访问磁盘和任何一个地址已知的块。通常一次只访问一条记录。

16. 记录组块的三种形式？Record Blocking

（1）固定组块（fixed blocking）：使用固定长度的记录，若干个完整的记录

被保存在一个块中。在每个块的末尾可能会有一些未被使用的空间（内部碎片）。**用于固定长度记录顺序文件**

（2）可变长度跨越式组块（variable-length spanned blocking）：使用可变长记录，紧缩到块中，完全利用块的空间。可能会跨越两个块，通过指向后继块的指针连接。

（3）可变长度非跨越式组块（variable-length unspanned blocking）：使用可变长记录，但是不会跨越块，如果当前剩余空间不足以放置下一条记录，则无法使用这一空间

17. 空闲空间管理 Free space management *4

位表：使用一个向量，向量的每一位表示磁盘的一个块，1 为占用，0 为空

链式空闲区：通过使用指向每个空闲区的指针和他们长度的值，将空闲区通过指针链接在一起

索引：将空闲区视为一个文件，使用索引表

空闲块列表：每个块都有一个顺序号，所有空闲块的序号都保存磁盘上的一个保留区中

//RAID 和高速磁盘 SSD 会考单选

RAID 独立磁盘冗余阵列，用多个独立的磁盘组成在一起形成一个大的磁盘系统，从而实现比单块磁盘更好的存储性能和更高的可靠性。