关于操作系统基本概念，正确的是（ A）

A. 我们通过操作系统来管理计算机硬件资源

B. 操作系统本身是一种硬件程序

C. 操作系统的基本作用是管理用户的程序

D. C 编译器是操作系统的一部分

计算机是按照（图灵）模型工作的

实模式与保护模式的说法正确的是（D ）

A. 系统开始运行后，操作系统处于实模式

B. 保护系统能够保护系统不被修改

C. 它们都是操作系统启动时使用的寻址方式

D. 它们对寻址的解释方式不同

引导程序启动时，为何需要将代码从内存的 0x7c00 拷贝至 0x9000?

A. 为给后面载入操作系统腾出地方

B. 因为开发程序员有强迫症，一定要移到整数字节的地方

C. 为了给中断向量表腾出空间

D. 因为开发的程序员想炫技，这样显得很悬很酷有档次

程序只有被装入（ 内存 ）中才能被运行。

按下电源开关，（ BIOS ）从引导扇区将操作系统代码载入到内存。

内存是一种 RAM，磁盘是一种 ROM

单个用户通常不能总是使得 CPU 和 I/O 设别都忙，通过（多道程序设计）的方式可以保证 CPU 总有作业执行。

下列（B ）资源不是操作系统应该管理的。

A. 内存

B. 源程序

C. CPU

D. 外存

如果一个进程需要从键盘输入一些字符，则应该在键盘输入之前发出中断信号

并发的英文是(concurrency)

操作系统的运行是一种（ 中断 ）驱动机制。

当程序运行错误如发生了除零操作，这也会触发中断

操作系统对硬盘进行 I/O 控制时，通常采用（直接内存）方式。

硬盘、显存等：DMA 方式

键盘、鼠标等：中断方式

在操作系统中，一个进程没有运行结束，就可以开始其它进程的运行，这种方式被称为**并发。**

执行系统调用的过程包括如下主要操作：

① 返回用户态

② 执行 trap 指令

③ 传递系统调用参数

④ 执行相应的服务程序

正确的执行顺序是(3-2-4-1)

开机启动时，操作系统处于内核模式。

在多道程序设计的计算机系统中,CPU（可以被多个进程交替占用 ）

操作系统程序可以执行特权指令。

计算机开机后，操作系统被加载到（RAM ）中。

当前操作系统将时间片设为 10ms，为此，操作系统需要设置时钟为 10ms，设置时钟的指令应运行在**内核模式。**

操作系统提供给编程人员的接口是**系统调用。**

在用户态执行的是**命令解释程序。**

进程调度与切换程序属于操作系统内核程序。

WEEK 6

1、一个进程从运行态变为就绪态，必然引起进程的切换

2、一个进程在 CPU 上的时间片结束（或用完）后，进程将进入就绪态

3、只有在运行状态的进程才能进入阻塞状态

4、进程自身决定从运行到阻塞

5、同一进程经过多次创建，运行在不同的数据集上，形成了不同的进程

6、若进程所请求的 I/O 完成后，将使其状态从阻塞变为就绪

7、操作系统对硬盘进行 I/O 控制时，通常采用直接内存访问方式

8、在单处理器系统中，若同时存在 10 个进程，则处于就绪队列中的进程最多有（9）个，处于阻塞队列中的进程最多有（10）个

9、DMA 的主要目的是提高数据传输效率

10、进程获得处理器运行是通过调度得到的

11、进程申请处理器（CPU）而得不到满足时，其状态变为就绪态

✕在单处理器系统中，任何时刻都只有一个进程处于运行状态

纠正：在单处理系统中，任何时刻真正在运行的作业至多只能有一种

✕优先级一旦确定不能改动

纠正：进程优先级{静态优先级}：一旦确定不能改动{动态优先级}：根据进程执行情况变化调整优先级

12、调用本地函数操作不需要使用特权指令（随着进程执行时间增加，优先级降低）

13、微内核特点：系统更加可靠、不如单一内核稳定、添加系统服务时不必修改内核

14、在父进程里使用 wait 函数可以让父进程等待子进程

15、函数通过实参传递值是放在栈段；进程的优先级是放在 PCB 中；一个进程的 PCB 是放在内存的内核空间中；PCB 是进程存在的唯一标记；进程的堆栈指针应该放在 PCB 里；进程的全局变量应该存储在静态存储区；进程创建时操作系统应该为进程创建一个 PCB

16、操作系统提供给编程人员的接口是系统调用

17、在用户态执行的是（命令解释程序）

18、操作系统必须提供（中断处理）功能

19、分时系统：交互性、独立性、及时性、多路性

批处理系统：分为单道批处理系统和多道批处理系统，不具备交互性，能提高 CPU 利用率

分时操作一般不具备长期调度

20、进程：操作系统资源分配的基本单位，是 PCB 结构与程序和数据的组合

线程：处理器任务调度和执行的基本单位

一个进程至少有一个线程，线程是进程的一部分，所以线程也被称为轻权进程或者轻量级进程。

不管系统中是否有线程，进程都是拥有资源的独立单位

21、支持多道程序设计的操作系统在运行过程中不断地选择新进程运行来实现 CPU 的共享，不是引起操作系统选择新进程的直接原因是(D)。

A. 运行进程的时间片用

B. 运行进程出错

C. 运行进程要等待某一事件的发生

D. 有新进程进入就绪状态

本题考查的是进程调度的时机。操作系统中将进程的状态分成 3 种：运行、阻塞、就绪。运行状态的进程占用 CPU，就绪状态的进程已经准备好接受调度，但是还没有占有 CPU。阻塞进程正在等待某件事情的发生。进程调度只能发生在时间片用完，当前进程等待某事件的发生，或者出错，有更高优先级的进程进入队列(只能出现在优先权调度方法或者抢占式调度中)。

22、fork 函数返回值为 0 时表示当前进程为子进程；返回值为-1 表示调度失败；返回值>0 表示当前是父进程，返回值是子进程 ID　fork 函数有两个返回值

23、子进程可以被赋予新的功能，并且与父进程并发执行;父进程创建了子进程后，子进程与父进程不共享内存空间

24、多个进程在处理器（cpu）上执行时，进程之间可能是无关的，也可能是有交互性的

25、进程创建的两个条件：加载到内存、分配资源。

创建时需要做的事：填写一个该进程的 PCB，分配该进程适当的内存空间，将该进程插入就绪队列

成功创建后进入就绪态等待 CPU 分配，在创建时不需要分配 CPU.

26、若系统中没有运行进程，则表明系统中一定没有就绪进程

27、随着进程的运行，堆与栈段会增加或减少，一个进程占用的内存大小是动态变化的

28、操作系统是一种中断驱动机制

29、在 Ubuntu 系统中，使用 fork 函数来创建子进程

30、进程的两种通信方式：共享内存；消息传递

31、程序必须加载到内存中才能运行

WEEK 8

1、两个合作进程无法通过高级程序设计语言中的全局变量交换数据，可以通过文件系统、消息传递系统、共享内存交换数据。

2、在一个多线程系统中，堆是线程之间共享的

3、多线程系统的特长：

基于 GUI 的调试程序用不同的线程分别处理用户输入、计算、跟踪等操作；Web 服务器利用线程相应 HTTP 请求；利用线程并行地在执行矩阵乘法运算

4、在多对一的线程模型中，当一个多线程进程中的某个进程被阻塞后，整个进程都会阻塞

5、操作系统是通过进程控制块来对并发执行的进程进行控制和管理

6、进程在处理器上执行时，进程之间可能是无关的，也可能是有交互性的

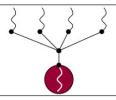
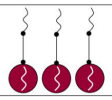
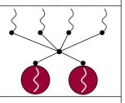
7、线程独享的资源：寄存器、栈、优先级 不是线程独享的：代码

8、进程之间交换数据的途径：共享文件；消息传递；访问共享存储区域

不是交换数据的途径：访问进程的地址空间

9、可能发生的状态转化：从运行到阻塞、从就绪到运行、从运行到就绪

10、阻塞到运行是不可能发生的

	用户级线程	核心级线程	用户+核心级
实现模型			
利用多核	差	好	好
并发度	低	高	高
代价	小	大	中
内核改动	无	大	大
用户灵活性	大	小	大

11、线程有自己的堆栈和局部变量,但线程没有单独的地址空间；线程包括 CPU 现场，可以独立执行程序

12、用信箱世家进程间互相通信的通信机制需要两个通信原语：发送原语和接受原语

WEEK 9

1、在作业调度算法中，当系统中同时存在 CPU 约束型与 I/O 约束型的作业时，一般来说，IO 约束型的作业优先级较高，应当先执行

2、假设系统中所有进程同时到达，则使进程平均周转时间最短的是短进程优先调度

3、作业是用户提交的，进程是系统生成的。两者的区别在于：作业是以用户任务为单位的，进程以操作系统控制为单位

4、先来先服务对短进程不利

5、满足短作业优先且不会发生饥饿现象的是：高响应比优先

6、不可能发生饥饿现象的是：时间片轮转法

7、先来先服务调度有利于 CPU 繁忙型作业，不利于 IO 繁忙型作业

8、先来先服务平均等待时间比时间片轮转法长

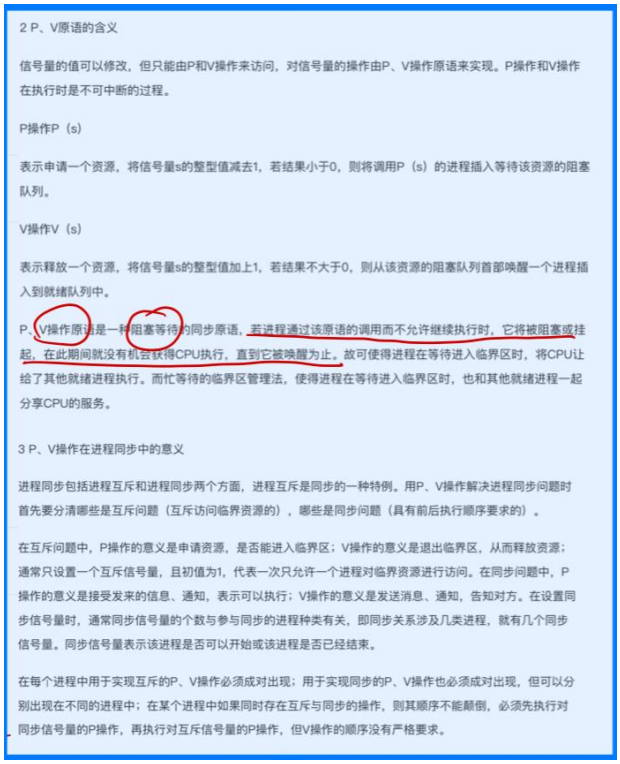
9、批处理系统通常采用最短作业优先、最短剩余时间优先进行调度

10、多级队列的优势是可以缓解饥饿现象，不能克服

11、进程调度算法准则：尽快响应交互式用户的请求；尽可能提高系统的吞吐量；尽量提高 CPU 的利用率

WEEK 10

1、如果进程 A 对信号量 S 进行 P 操作，则 S 的值应当减 1



2 P、V原语的含义

信号量的值可以修改，但只能由P和V操作来访问，对信号量的操作由P、V操作原语来实现。P操作和V操作在执行时是不可中断的过程。

P操作P（s）

表示申请一个资源，将信号量s的整型值减去1，若结果小于0，则将调用P（s）的进程插入等待该资源的阻塞队列。

V操作V（s）

表示释放一个资源，将信号量s的整型值加上1，若结果不大于0，则从该资源的阻塞队列首部唤醒一个进程插入到就绪队列中。

P、V操作原语是一种阻塞等待的同步原语，若进程通过该原语的调用而不允许继续执行时，它将被阻塞或挂起，在此期间就没有机会获得CPU执行，直到它被唤醒为止。故可使得进程在等待进入临界区时，将CPU让给了其他就绪进程执行。而忙等待的临界区管理法，使得进程在等待进入临界区时，也和其他就绪进程一起分享CPU的服务。

3 P、V操作在进程同步中的意义

进程同步包括进程互斥和进程同步两个方面，进程互斥是同步的一种特例。用P、V操作解决进程同步问题时首先要分清哪些是互斥问题（互斥访问临界资源的），哪些是同步问题（具有前后执行顺序要求的）。

在互斥问题中，P操作的意义是申请资源，是否能进入临界区；V操作的意义是退出临界区，从而释放资源；通常只设置一个互斥信号量，且初值为1，代表一次只允许一个进程对临界资源进行访问。在同步问题中，P操作的意义是接受发来的信息、通知，表示可以执行；V操作的意义是发送消息、通知，告知对方。在设置同步信号量时，通常同步信号量的个数与参与同步的进程种类有关，即同步关系涉及几类进程，就有几个同步信号量。同步信号量表示该进程是否可以开始或该进程是否已经结束。

在每个进程中用于实现互斥的P、V操作必须成对出现；用于实现同步的P、V操作也必须成对出现，但可以分别出现在不同的进程中；在某个进程中如果同时存在互斥与同步的操作，则其顺序不能颠倒，必须先执行对同步信号量的P操作，再执行对互斥信号量的P操作，但V操作的顺序没有严格要求。

2、test_and_set 指令可以用来实现进程互斥

3、解决临界区问题的条件：有限等待；互斥；前进

4、竞争条件是因为进程并发执行引起的；竞争条件是可避免的；竞争条件的错误结果与进程的并发顺序有关

避免竞争条件应满足的四个条件

1. 任何两个进程不能同时处于临界区（互斥访问）
2. 不应对CPU的速度和数目做任何假设
3. 临界区外的进程不得阻塞其他进程（有空让进）
4. 不得使进程在临界区外无休止地等待（有限等待）

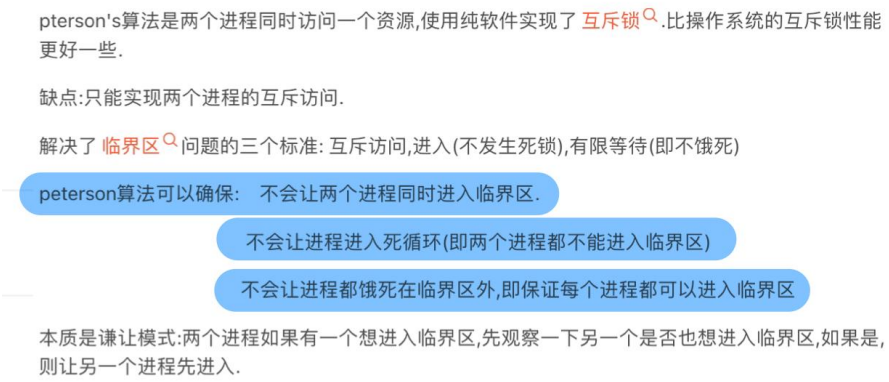
5、一个正在访问临界资源的进程由于申请等待 IO 操作而被中断时，它允许其他进程抢占处理器，但是不得进入该进程的临界区。

6、共享变量是指（可以被多个进程访问）的变量

7、临界区是指进程中（用于访问共享变量）的那段代码

临界资源：在一段时间里只允许一个进访问的资源

8、Peterson 算法是一个实现互斥锁的并发程序设计算法，可以控制两个线程访问一个共享的单用户资源而不发生访问冲突；不能解决进程同步问题，只能实现两个进程的互斥访问



9、可以被多个进程在任意时刻共享的代码必须是不允许任何修改的代码

10、对于两个并发进程，设互斥信号量为 mutex(初始值为 1)

Mutex=0:有一个进程进入临界区

Mutex=1:有一个进程进入临界区，另一个进程在等待进入临界区

11、使用关中断的方法可以实现进程互斥

12、同步信号量的初值一般设为 0；互斥信号量的初值一般设为 1