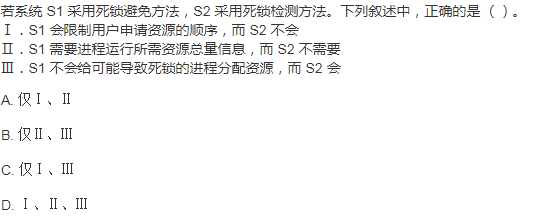
1. 语句#include <stdlib.h>是正确的，而且程序编译速度比#include “stdlib.h”要快

<>只查找系统目录，如果自己写的头文件，用<>包含，会报语法错误

“ ”首先搜索本地目录，如果本地没有，再搜索系统目录

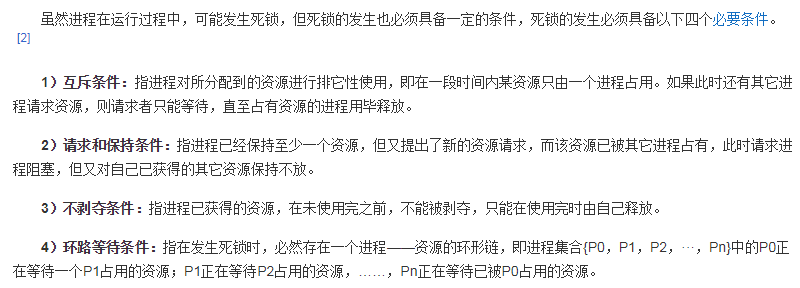
可以把系统文件放到本地目录下，改成“ ”，可以优先使用

2.



死锁处理主要有三种策略：死锁预防，死锁避免，死锁检测

死锁预防是破坏死锁四个必要条件的一个或几个：



**死锁预防**典型的是**顺序资源分配法**，限制进程申请资源的顺序!后面的进程只能申请编号更大的资源。

死锁避免算法，典型的是银行家算法，不会限制申请资源的顺序，但是会限制分配资源的顺序，避免进入不安全状态！因此，需要知道全局性的资源需求矩阵以及资源总量。

死锁检测，是最宽松的，啥也不限制，只有当出现死锁且被检测到时才会去处理解除。因此，这种灵活性的算法在Unix上实现了。常用资源分配图结合死锁定理检测。

综上，II、III才是正确的。要明白限制用户申请资源的顺序是很大的限制，限制分配资源的顺序是稍微宽松些的限制，检测是不做限制。

3. 当进程因时间片用完而让出处理机时，该进程转变为（就绪）状态

进程的状态：在三态模型中，进程状态分为三个基本状态：即运行态、就绪态、阻塞态（又称为等待态或睡眠态）。

一个进程从创建而产生至撤销而消亡的整个生命期间，有时占有处理器执行，有时虽可运行但分不到处理器、有时虽有空闲处理器但因等待某个事件的发生而无法执行，这一切都说明进程和程序不相同，它是活动的且有状态变化的，这可以用一组状态加以刻画。为了便于管理进程，一般来说，按进程在执行过程中的不同情况至少要定义三种不同的进程状态：

（1）运行(running)态：进程占有处理器正在运行。

（2）就绪(ready)态：进程具备运行条件，等待系统分配处理器以便运行。

（3）等待(wait)态：又称为阻塞(blocked)态或睡眠(sleep)态，指进程不具备运行条件，正在等待某个事件的完成。

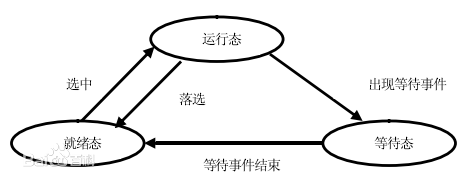
通常，一个进程在创建后将处于就绪状态。每个进程在执行过程中，任意时刻当且仅当处于上述三种状态之一。同时，在一个进程执行过程中，它的状态将会发生改变。引起进程状态转换的具体原因如下：

（1）运行态一一等待态：等待使用资源或某事件发生，如等待外设传输;等待人工干预。

（2）等待态一一就绪态：资源得到满足或某事件己经发生，如外设传输结束；人工干预完成。

（3）运行态一一就绪态：运行时间片到，或出现有更高优先权进程。

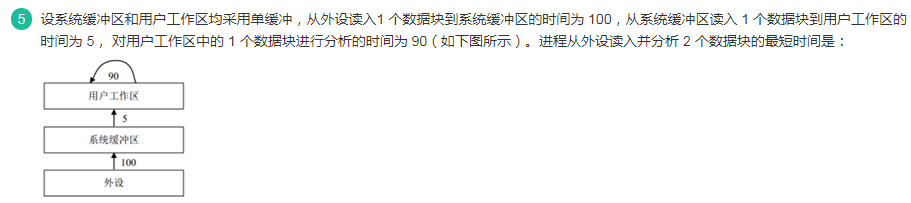
（4）就绪态一一运行态：CPU空闲时被调度选中一个就绪进程执行。



4.



5.

300

6. CPU流水线中常见冲突有哪些

7. 在CPU、GPU、FPGA、DSP等硬件中(或其它计算类硬件)，挑选至少2种加以对比说明？

8. 请对比Linux内核中noop，deadline，CFQ等IO调度算法。

9. 以DDR4-2666双通道内存条为例，计算内存条带宽；以PCIe Gen2 x16为例，计算总线吞吐？

10. 给定两个已经排序好的数组（可能为空）nums1和nums2，size分别为m和n，找到所有元素的中位数，要求算法时间复杂度为O(log(m+n))。  
例子1：nums1 = [1, 3]，nums2 = [2] ，则中位数是 2.0

例子2：nums1 = [1, 2]，nums2 = [3, 4]，则中位数为 (2 + 3)/2 = 2.5

（注：不能使用本地IDE）

11.



12.