1.下列进程调度算法中，综合考虑进程等待时间和执行时间的是（D）

A．时间片轮转调度算法 B．短进程优先调度算法

C．先来先服务调度算法 D．高响应比优先调度算法

2.某计算机系统中有8台打印机，由K个进程竞争使用，每个进程最多需要3台打印机。该系统可能会发生死锁的K的最小值是（ C ）

A．2 B．3 C．4 D．5

3.下列选项中，操作系统提供给应用程序的接口是（ A ）

A．系统调用 B．中断 C．库函数 D．原语

4.下列选项中，导致创建新进程的操作是（ C ）

Ⅰ用户登陆成功 Ⅱ设备分配 Ⅲ 启动程序执行

A．仅Ⅰ和Ⅱ B．仅Ⅱ和Ⅲ C．仅Ⅰ和Ⅲ D．Ⅰ、Ⅱ、Ⅲ

5.设与某资源相关联的信号量初值为3，当前值为1，若M表示该资源的可用个数，N表示等待该资源的进程数，则M、N分别是（ B ）

A．0，1 B．1，0 C．1，2 D．2，0

6.下列选项中，降低进程优先权级的合理时机是（ A ）

A.进程时间片用完 B.进程刚完成I/O，进入就绪队列

C.进程长期处于就绪队列中 C.进程从就绪状态转为运行态

7.进程P0和P1的共享变量定义及其初值为：

boolean flag[2];

int turn=0;

flag[0]=false; flag[1]=false;

若进行P0和P1访问临界资源的类C代码实现如下：

void p0( ) //进程p0

{ while(TRUE)

{ flag[0]=TURE; turn=1;

while(flag[1]&&(turn==1)) ;

临界区；

flag[0]=FALSE;

}

}

void p1( ) //进程p1

{ while(TRUE)

{ flag[1]=TURE; turn=0;

while(flag[0]&&(turn==0)) ;

临界区；

flag[1]=FALSE;

}

}

则并发执行进程P0和P1时产生的情况是（ D ）

A．不能保证进程互斥进入临界区，会出现“饥饿”现象

B．不能保证进程互斥进入临界区，不会出现“饥饿”现象

C．能保证进程互斥进入临界区，会出现“饥饿”现象

D．能保证进程互斥进入临界区，不会出现“饥饿”现象

(注：Peterson算法本身不会导致饥饿问题！进程进入临界区不出来的情况不是Peterson算法的范畴！！！)

8.下列选项中，不可能在用户态发生的事件是（ C ）

A． 系统调用 B 外部中断 C 进程切换 D 缺页

### **进程切换为什么不可能在用户态发生？**

进程切换的本质：需要直接操作CPU寄存器、内存管理单元（MMU）等硬件资源，并修改内核数据结构（如进程控制块PCB）。这些操作必须由内核特权指令完成。

触发条件：即使因时间片耗尽（用户态触发的计时器中断）导致需要切换进程，实际的切换动作（保存当前进程上下文、加载新进程上下文）也必须由内核在内核态完成。

用户态无权限：用户态程序无法直接调度其他进程，否则会破坏系统的隔离性和安全性。

**外部中断为什么可以在用户态触发？**

触发时机：外部中断（如键盘输入、硬件故障）可能在任何时候发生，包括用户态程序执行期间。中断发生后，CPU会自动切换到内核态处理中断，但中断的触发本身与CPU当前状态无关。

关键区别：题目问的是“发生的事件”，中断的触发是事件的发生，而后续处理由内核完成。因此，中断可以在用户态被触发（尽管处理需要内核）。

9.中断处理和子程序调用都需要压栈以保护线程，中断处理一定会保存而子程序调用不需要保存其内容的是（ B ）

A．程序计数器 B. 程序状态字寄存器

C 通用数据寄存器 D 通用地址寄存器

**子程序调用（函数调用）：**

通过CALL指令（或类似机制）主动跳转到子程序。

需要保存返回地址（PC）：以便子程序执行完后能回到调用点。

不需要保存PSW：因为子程序调用是程序主动发起的控制流转移，不会改变CPU状态（如中断开关、特权级等）。

可能保存通用寄存器（如C/D选项）：由调用约定决定（如被调用者保存规则）。

10.一个多道批处理系统中仅有P1, P2两个作业，P2比P1晚5ms到达，它的计算和I/O操作顺序如下：

P1:计算60ms，I/O 80ms，计算20ms

P2: 计算120ms，I/O 40ms，计算40ms

若不考虑调度和切换时间，则完成两个作业需要的时间最少是（ B ）

1. 240ms B. 260ms C. 340ms D. 360ms

11.若某单处理器多进程系统中有多个就绪态进程，则下列关于处理机调度的叙述中错误的是（ C ）

A.在进程结束时能进行处理机调度

B.创建新进程后能进行处理机调度

C.在进程处于临界区时不能进行处理器调度

D.在系统调用完成并返回用户态时能进行处理机调度

**调度器不关心进程是否在临界区，只要满足调度条件（如时间片到、更高优先级进程就绪），即可触发调度。**

12.假设5个进程P0, P1, P2, P3, P4共享三类资源R1, R2, R3，这些资源总数分别是18，6，22. T0时刻的资源分配情况如下表所示，此时存在的一个安全序列是：（ D ）**（审题，不要看走眼了！！！）**



1. P0,P2,P4,P1,P3 B. P1, P0, P3, P4, P2 C. P2,P4,P3,P1,P0 D. P3,P4,P2,P1,P0

13.下列关于进程和线程的叙述中，正确的是（ A ）

1. 不管系统是否支持线程，进程都是资源分配的基本单位
2. 线程是资源分配的基本单位，进程是调度的基本单位
3. 系统级线程和用户级线程的切换都需要内核支持
4. 同一进程中的各个线程拥有各自不同的地址空间

二、综合应用题：(2009年408真题)

三个进程P1、P2、P3互斥使用一个包含N（N > 0）个单元的缓冲区。P1每次用produce()生成一个正整数并用put()送入缓冲区某一空单元中；P2每次用getodd()从该缓冲区中取出一个奇数并用countodd()统计奇数个数；P3每次用geteven()从该缓冲区中取出一个偶数并用counteven()统计偶数个数。请用信号量机制实现这三个进程的同步与互斥活动，并说明所定义信号量的含义。要求用伪代码描述。

要点：

（1）缓冲区是一互斥资源，因此设互斥信号量mutex。

（2）同步问题：P1、P2因为奇数的放置与取用而同步，设同步信号量odd；P1、P3因为偶数的放置与取用而同步，设同步信号量even；P1、P2、P3因为共享缓冲区，设同步信号量empty。

semaphore mutex=1;

semaphore odd=0，even=0;

semaphore empty=N;

main()

cobegin{

Process P1

while(True)

{

number=produce()；

P(empty);

P(mutex);

put()；

V(mutex);

if number%2= =0

V(even);

else

V(odd);

}

Process P2

while (true)

{

P(odd);

P(mutex);

getodd()；

V(mutex);

V(empty);

countodd()；

}

Process P3

while (true)

{

P(even);

P(mutex);

geteven()；

V(mutex);

V(empty);

counteven()；

}

}coend

【评分说明】

①能正确给出互斥信号量定义与含义的，给1分。

②能正确给出3个同步信号量定义与含义的，各给1分，共3分。

③能正确描述P1、P2和P3进程活动的，各给1分，共3分。

④wait()、signal()等同于P、V。

设计该题的目的是考查学生对进程间同步概念的理解，以及利用信号量实现进程同步的能力。为了简化问题，让学生集中精力设计信号量，我们在题干中特别定义了几个函数，用produce()生成一个正整数，用put()送入缓冲区某一空单元中，用getodd()从该缓冲区中取出一个奇数，用countodd()统计奇数个数，用geteven()从该缓冲区中取出一个偶数，用counteven()统计偶数个数；其用意就是让学生集中在信号量的定义和使用上，不要考虑具体功能的实现细节。

该题是一个经典同步问题——“生产者-消费者问题”的变形，所以可以对生产者-消费者问题做适当修改。这里的变化主要在于我们把产品分成了两类，这样就有两类消费者；生产者每生产一件产品，需要判别属于哪一类，根据产品的类别不同向不同的消费者发送同步信息。消费者根据产品的有无来决定是否继续消费。

此题统计难度为0.427，与估计难度一致，此题设计较好，达到了设计的目的。因为进程同步与互斥是操作系统中的重点和难点，也是许多考试必考的内容，所以很多考生比较重视对此部分内容的复习，对同步与互斥问题比较清楚。