计算机考研系列书课包

玩转操作系统

主讲人 刘财政

第三讲　处理机调度

本讲内容

考点一: 处理机调度 ★

考点二: 调度算法 ★★★★★

考点三: 上下文切换机制 ★★★

考点四: 死锁 ★★★★★

考点四:

死锁

考点框架

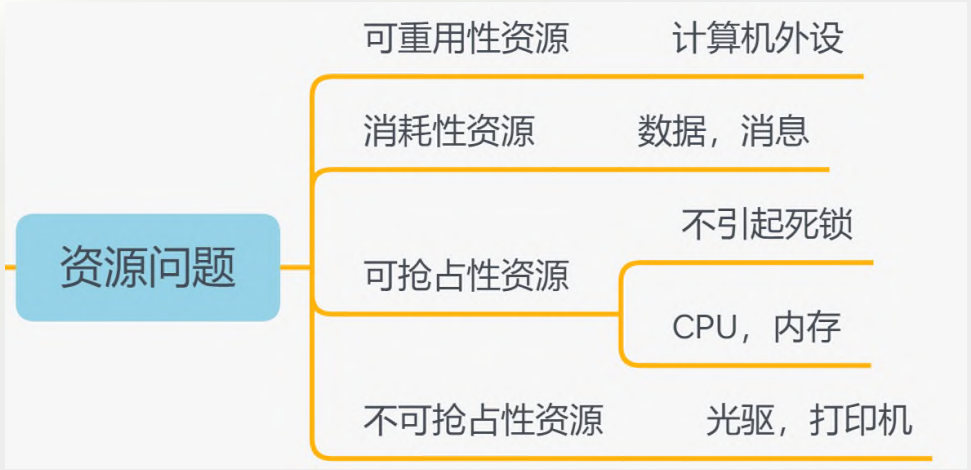
- 死锁的原因
- 死锁的必要条件
- 死锁预防方法
- 死锁避免算法
- 死锁检测与解除

考点四: 死锁

死锁的原因

考点四: 死锁

计算机中的资源

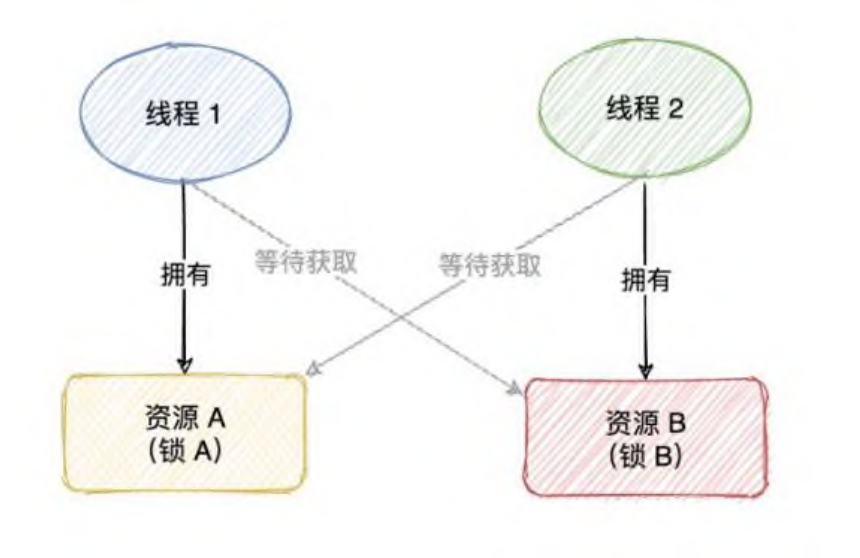


考点四: 死锁

死锁的概念

定义：多个进程在运行过程中因争夺资源而造成的一种僵局。

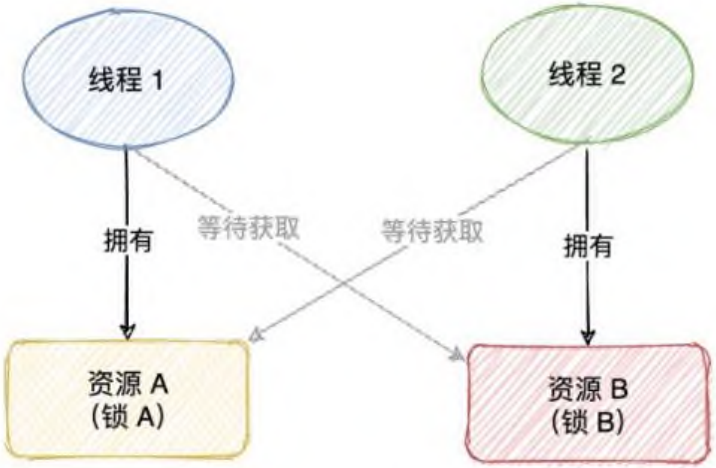
系统处于这种状态时，若无外力作用，将无法向前推进



考点四: 死锁

死锁的概念

- 线程1和线程2都需要资源A和资源B才可以执行完成，但是，某一以时刻，线程1拥有了资源A，但是等待获取资源B；线程1拥有了资源B，但是等待获取资源A；但是系统中没有足够的资源A和资源B以供分配，所以线程1和线程2都会被阻塞，如果没有外力作用，他们将永远无法执行。



考点四: 死锁

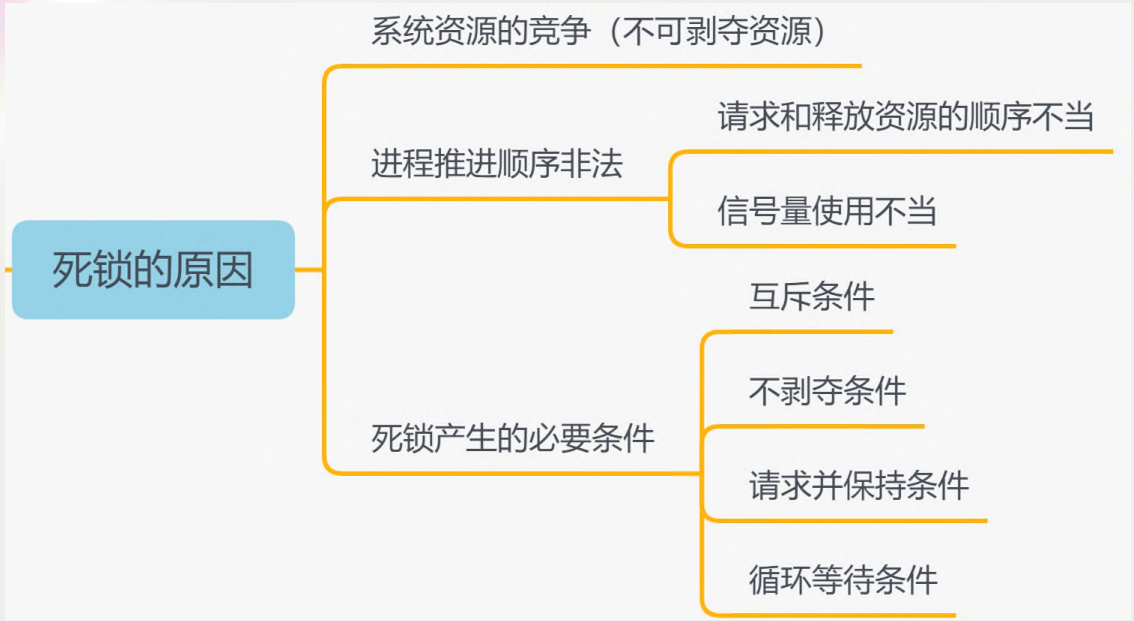
死锁的原因

- 锁产生的原因有两个
- 系统资源的竞争。在系统所配置的非剥夺性资源，由于它们的数量不能满足进程运行的需要，会使进程在运行过程中，因为争夺这些资源而进入僵局。
- 进程推进顺序非法。由于进程在运行中具有异步性特征，当进程的执行顺序不合理时，也可能会导致死锁。

考点四: 死锁

死锁的必要条件

考点四: 死锁


死锁的原因
系统资源的竞争（不可剥夺资源）
进程推进顺序非法
请求和释放资源的顺序不当
信号量使用不当
死锁产生的必要条件
互斥条件
不剥夺条件
请求并保持条件
循环等待条件

考点四: 死锁

- 互斥使用：进程对所分配到资源的使用具有排他性，即在一段时间内某资源只由一个进程占用。如果此时还有其它进程请求该资源，则请求者只能等待，直至占有该资源的进程使用完毕将其释放。由设备的固有条件决定，应予以保护
- 请求保持：进程已占用至少一个资源，又申请被其它进程占有不释放的资源，则请求进程阻塞，但不释放已占用资源
- 不可抢占：进程已获得的资源在使用完之前，不能被其他进程抢占，只能在使用完后由自己释放
- 循环等待：发生死锁时，必存在一个进程—资源循环链

考点四: 死锁

- 面对死锁，需要对死锁进行处置，常见的死锁处理策略有：

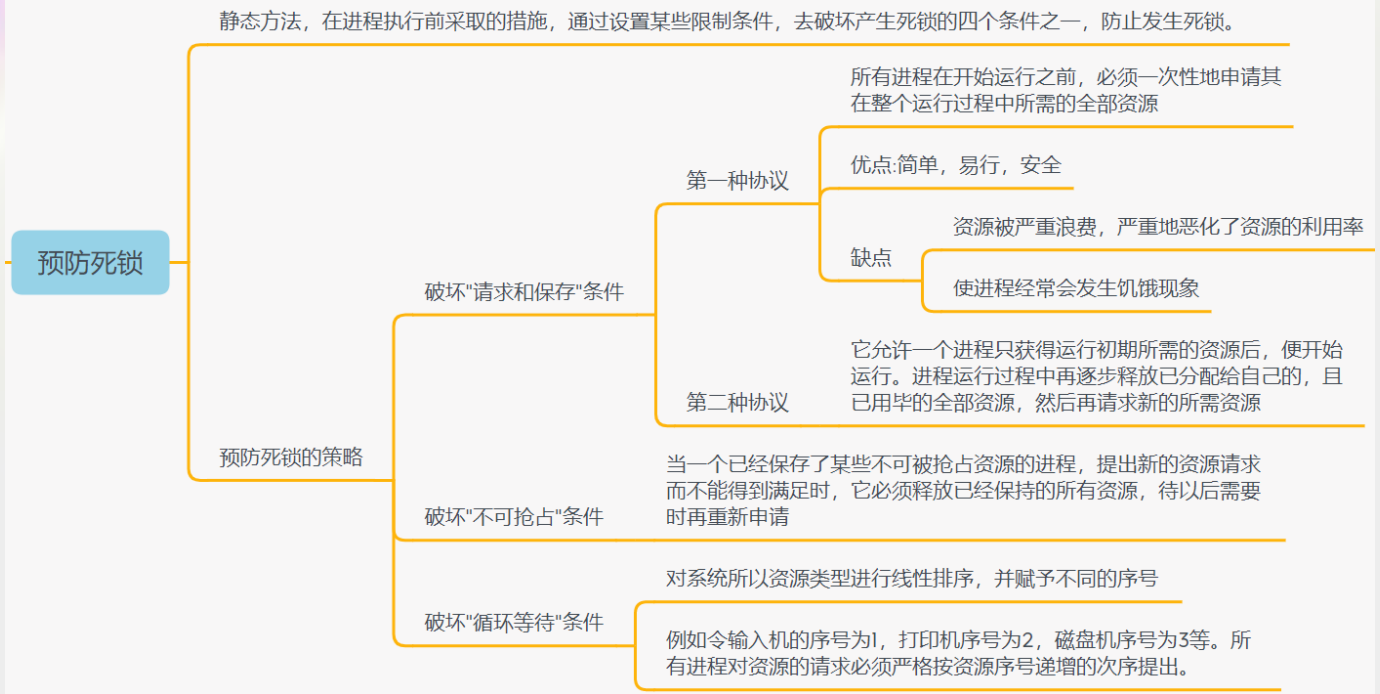
考点四: 死锁

死锁预防方法

考点四: 死锁

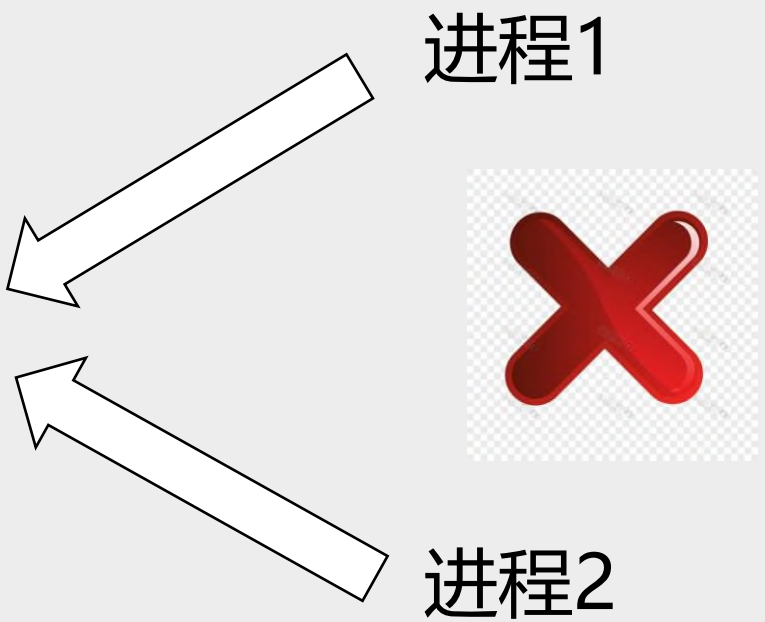
- 预防死锁方法是一种较简单和直观的事先预防的方法。
- 该方法是通过设置某些限制条件，去破坏产生死锁的四个必要条件中的一个或几个条件，来预防发生死锁。
- 预防死锁是一种较易实现的方法，已被广泛使用。

考点四: 死锁


静态方法，在进程执行前采取的措施，通过设置某些限制条件，去破坏产生死锁的四个条件之一，防止发生死锁。
预防死锁
预防死锁的策略
破坏"请求和保存"条件
第一种协议
所有进程在开始运行之前，必须一次性地申请其在整个运行过程中所需的全部资源
优点:简单，易行，安全
缺点
资源被严重浪费，严重地恶化了资源的利用率
使进程经常会发生饥饿现象
第二种协议
它允许一个进程只获得运行初期所需的资源后，便开始运行。进程运行过程中再逐步释放已分配给自己的，且已用毕的全部资源，然后再请求新的所需资源
破坏"不可抢占"条件
当一个已经保存了某些不可被抢占资源的进程，提出新的资源请求而不能得到满足时，它必须释放已经保持的所有资源，待以后需要时再重新申请
破坏"循环等待"条件
对系统所以资源类型进行线性排序，并赋予不同的序号
例如令输入机的序号为1，打印机序号为2，磁盘机序号为3等。所有进程对资源的请求必须严格按资源序号递增的次序提出。

考点四: 死锁

- 条件1：互斥条件：由设备的固有条件所决定，不仅不能改变，还应加以保证



考点四: 死锁

条件2：请求保持条件：可破坏

- 协议1：所有进程在开始运行之前，必须一次性地申请其在整个运行过程中所需的全部资源。特征如下：
  - 运行期间不再申请新资源，破坏“请求”条件
  - 等待期间不会占有任何资源，破坏“保持”条件
  - 协议简单安全，但会造成资源浪费，经常发生进程饥饿现象

考点四: 死锁

条件2：请求保持条件：可破坏

- 协议2：进程获得运行初期所需的资源后，便可开始运行。如某进程将磁带上的数据复制至磁盘，将磁盘文件排序后打印。需要磁带机、磁盘文件、打印机。
- 协议1需要一次性申请到3个资源，而协议2先申请前2个资源，排序后释放磁带机和磁盘文件，然后请求打印机。特点是资源利用率高，能有效减少进程发生饥饿现象的概率

考点四: 死锁

条件3 ： 不可抢占：可破坏

- 已经保持了某些不可被抢占资源的进程，提出新的资源请求而不能得到满足时，其申请到的资源可以被剥夺，以后需要时再重新申请
- 特点：实现复杂，代价较大

考点四: 死锁

条件4：循环等待：可破坏

- 对系统所有资源类型进行线性排序，并赋予不同的序号
- 进程必须按照序号递增顺序申请资源

考点四: 死锁

条件4 ： 循环等待：可破坏

- 特点
  - 资源序号安排很重要
  - 系统资源利用率和吞吐量比其他策略有明显改善
- 缺点
  - 资源序号的相对稳定性导致新设备的增加受到限制
  - 由于作业使用资源的顺序和系统资源序号顺序未必相同，可能导致资源浪费

考点四: 死锁

死锁避免算法

考点四: 死锁

安全状态

- 安全状态，是指**系统能按某种进程顺序**(P_1, P_2, …， P_n)(称〈P_1, P_2, …, P_n〉序列为安全序列)，来**为每个进程P_i分配其所需资源**，直至满足每个进程对资源的最大需求，**使每个进程都可顺利地完成。**
- 如果系统无法找到这样一个安全序列，则称系统处于不安全状态。

考点四: 死锁

安全状态

假定系统中有三个进程P1、 P2和P3，共有12台磁带机。进程P1总共要求10台磁带机，P2和P3分别要求4台和9台。假设在T0时刻，进程P1、P2和P3已分别获得5台、2台和2台磁带机，尚有3台空闲未分配，如下表所示： （P2、 P1、P3）

进 程	最 大 需 求	已 分 配	可 用
P_1	10	5	3
P_2	4	2	
P_3	9	2	

考点四: 死锁

安全状态

例如，在T_0时刻以后，**P_3又请求1台磁带机**，若此时系统把剩余3台中的1台分配给P_3，则系统便进入**不安全状态**。 因为，此时也无法再找到一个安全序列，例如，把其余的2台分配给P_2，这样，在P_2完成后只能释放出4台，既不能满足P_1尚需5台的要求，也不能满足P_3尚需6台的要求，致使它们都无法推进到完成，彼此都在等待对方释放资源，即陷入僵局，结果导致**死锁**。

进 程	最 大 需 求	已 分 配	可 用
P_1	10	5	3
P_2	4	2	
P_3	9	2	

考点四: 死锁

安全状态

- 如果系统无法找到这样一个安全序列，则称系统处于不安全状态
- 只要处于安全状态就不会发生死锁
- 并不是所有的不安全状态都会导致系统进入死锁状态，但当系统进入不安全状态后，可能进入死锁状态

考点四: 死锁

死锁避免算法思想

- 在避免死锁的方法中，允许进程动态地申请资源，但系统在进行资源分配之前，应先计算此次资源分配的安全性。
- 若此次分配不会导致系统进入不安全状态，则将资源分配给进程；
- 否则，令进程等待。

考点四: 死锁

死锁避免的数据结构

- 可利用资源向量Available。这是一个含有m个元素的数组，其中的每一个元素代表一类可利用的资源数目，其初始值是系统中所配置的该类全部可用资源的数目，其数值随该类资源的分配和回收而动态地改变。如果Available［j］=K，则表示系统中现有Rj类资源K个。

考点四: 死锁

死锁避免的数据结构

- 最大需求矩阵Max。这是一个n×m的矩阵，它定义了系统中n个进程中的每一个进程对m类资源的最大需求。如果Max［i,j］=K，则表示进程i需要Rj类资源的最大数目为K。

考点四: 死锁

死锁避免的数据结构

- 分配矩阵Allocation。这也是一个n×m的矩阵，它定义了系统中每一类资源当前已分配给每一进程的资源数。如果Allocation［i,j］=K，则表示进程i当前已分得Rj类资源的数目为K。

考点四: 死锁

死锁避免的数据结构

- 需求矩阵Need。这也是一个n×m的矩阵，用以表示每一个进程尚需的各类资源数。如果Need［i,j］=K，则表示进程i还需要Rj类资源K个，方能完成其任务。 Need［i,j］=Max［i,j］-Allocation［i,j］

考点四: 死锁

银行家算法预分配

- 设$Request_i$是进程P_i的请求向量，如果$Request_i[j]=K$，表示**进程P_i需要K个R_j类型的资源**。当P_i发出资源请求后，系统按下述步骤进行检查：

(1) 如果$Request_i[j] \le Need[i,j]$，便转向步骤2；否则**认为出错**，因为它所需要的资源数已超过它所宣布的最大值。

(2) 如果$Request_i[j] \le Available[j]$，便转向步骤(3)；否则， 表示尚无足够资源，**P_i须等待。**

考点四: 死锁

银行家算法预分配

(3) 系统**试探着把资源分配给进程P_i**，并修改下面数据结构中的数值：

Available [j] =Available [j] -$Request_i$ [j] ；

Allocation [i,j] =Allocation [i,j] +$Request_i$ [j] ；

Need [i,j] =Need [i,j] -$Request_i$ [j] ；

考点四: 死锁

银行家算法预分配

（4）系统执行安全性算法，检查此次资源分配后，系统是否处于安全状态。

- 若安全，才正式将资源分配给进程P_i，以完成本次分配；
- 否则， 将本次的试探分配作废，恢复原来的资源分配状态，让进程P_i等待。

考点四: 死锁

银行家算法安全性算法

(1) **设置两个向量**:

① 工作向量Work: 它表示系统可提供给进程继续运行所需的各类资源数目，它含有*m*个元素，在执行安全算法开始时，Work=Available;

② Finish: 它表示系统是否有足够的资源分配给进程，使之运行完成。开始时先做Finish [i] =false; 当有足够资源分配给进程时， 再令Finish [i] =true。

考点四: 死锁

银行家算法安全性算法

(2) 从进程集合中找到一个**能满足下述条件的进程**:

① Finish [i] =false;

② Need [i,j] ≤Work [j] ;

若找到， 执行步骤(3)， 否则，执行步骤(4)。

考点四: 死锁

银行家算法安全性算法

(3) 当进程P_i获得资源后，可顺利执行，直至完成，并**释放出分配给它的资源**，故应执行：

Work [j] = Work [i] +Allocation [i,j] ;

Finish [i] = true;

go to step 2;

考点四: 死锁

银行家算法安全性算法

(4) 如果所有进程的Finish [i] =true都满足， 则表示系统处于安全状态；否则，系统处于不安全状态。

考点四: 死锁

假定系统中有五个进程 $\{P_0, P_1, P_2, P_3, P_4\}$ 和三类资源 $\{A, B, C\}$ ，各种资源的数量分别为10、5、7，在 T_0 时刻的资源分配情况如图所示。

进程 \ 资源情况	Max			Allocation			Need			Available		
	A	B	C	A	B	C	A	B	C	A	B	C
P_0	7	5	3	0	1	0	7	4	3	3	3	2
P_1	3	2	2	2	0	0	1	2	2			
P_2	9	0	2	3	0	2	6	0	0			
P_3	2	2	2	2	1	1	0	1	1			
P_4	4	3	3	0	0	2	4	3	1			

考点四: 死锁

(1) T_0时刻的安全性:

T0 时刻的资源分配情况表

资源 / 情况 / 进程	Max			Allocation			Need			Available		
	A	B	C	A	B	C	A	B	C	A	B	C
P_0	7	5	3	0	1	0	7	4	3	3	3	2
P_1	3	2	2	2	0	0	1	2	2			
P_2	9	0	2	3	0	2	6	0	0			
P_3	2	2	2	2	1	1	0	1	1			
P_4	4	3	3	0	0	2	4	3	1			

考点四: 死锁

(1) T_0时刻的安全性:

T0 时刻的资源分配情况表

进程 \ 资源情况	Max			Allocation			Need			Available		
	A	B	C	A	B	C	A	B	C	A	B	C
P_0	7	5	3	0	1	0	7	4	3	3	3	2
P_1	3	2	2	2	0	0	1	2	2			
P_2	9	0	2	3	0	2	6	0	0			
P_3	2	2	2	2	1	1	0	1	1			
P_4	4	3	3	0	0	2	4	3	1			

考点四: 死锁

(1) T_0时刻的安全性:

资源 / 情况 / 进程	Max			Allocation			Need			Available		
	A	B	C	A	B	C	A	B	C	A	B	C
P_0	7	5	3	0	1	0	7	4	3	~~3~~	~~3~~	~~2~~
P_2	9	0	2	3	0	2	6	0	0			
P_3	2	2	2	2	1	1	0	1	1			
P_4	4	3	3	0	0	2	4	3	1			

考点四: 死锁

(1) T_0时刻的安全性:

资源 / 情况 / 进程	Max			Allocation			Need			Available		
	A	B	C	A	B	C	A	B	C	A	B	C
P_0	7	5	3	0	1	0	7	4	3	5	3	2
P_2	9	0	2	3	0	2	6	0	0			
P_3	2	2	2	2	1	1	0	1	1			
P_4	4	3	3	0	0	2	4	3	1			

考点四: 死锁

(1) T_0时刻的安全性:

资源 / 情况 / 进程	Max			Allocation			Need			Available		
	A	B	C	A	B	C	A	B	C	A	B	C
P_0	7	5	3	0	1	0	7	4	3	5	3	2
P_2	9	0	2	3	0	2	6	0	0			
P_4	4	3	3	0	0	2	4	3	1			

考点四: 死锁

(1) T_0时刻的安全性:

资源 / 情况 / 进程	Max			Allocation			Need			Available		
	A	B	C	A	B	C	A	B	C	A	B	C
P_0	7	5	3	0	1	0	7	4	3	7	4	3
P_2	9	0	2	3	0	2	6	0	0			
P_4	4	3	3	0	0	2	4	3	1			

考点四: 死锁

(1) T_0时刻的安全性:

资源 / 情况 / 进程	Max			Allocation			Need			Available		
	A	B	C	A	B	C	A	B	C	A	B	C
P_2	9	0	2	3	0	2	6	0	0	7	4	3
P_4	4	3	3	0	0	2	4	3	1	7	5	3

考点四: 死锁

(1) T_0时刻的安全性:

资源 / 情况 / 进程	Max			Allocation			Need			Available		
	A	B	C	A	B	C	A	B	C	A	B	C
P_2 ⇩	9	0	2	3	0	2	6	0	0 ✓	7	5	3
P_4	4	3	3	0	0	2	4	3	1			

考点四: 死锁

(1) T_0时刻的安全性:

资源 / 情况 / 进程	Max			Allocation			Need			Available		
	A	B	C	A	B	C	A	B	C	A	B	C
P_4	4	3	3	0	0	2	4	3	1	7	5	3
										10	5	5

考点四: 死锁

(1) T_0时刻的安全性:

资源 / 情况 / 进 程	Max			Allocation			Need			Available		
	A	B	C	A	B	C	A	B	C	A	B	C
										10	5	7

考点四: 死锁

(2) P_1请求资源：P_1发出请求向量**$Request_1(1，0，2)$**，系统按银行家算法进行检查：

① $Request_1(1, 0, 2) \leq Need_1(1, 2, 2)$

② $Request_1(1, 0, 2) \leq$ **$Available_1(3, 3, 2)$**

进程 \ 资源情况	Max			Allocation			Need			Available		
	A	B	C	A	B	C	A	B	C	A	B	C
P_0	7	5	3	0	1	0	7	4	3	3	3	2
P_1	3	2	2	2	0	0	1	2	2			
P_2	9	0	2	3	0	2	6	0	0			
P_3	2	2	2	2	1	1	0	1	1			
P_4	4	3	3	0	0	2	4	3	1			

考点四: 死锁

(2) P_1请求资源：P_1发出请求向量**$Request_1(1, 0, 2)$**，系统按银行家算法进行检查：

① $Request_1(1, 0, 2) \leq Need_1(1, 2, 2)$

② $Request_1(1, 0, 2) \leq$ **$Available_1(3, 3, 2)$**

进程 \ 资源情况	Max A B C	Allocation A B C	Need A B C	Available A B C
P_0	7 5 3	0 1 0	7 4 3	3 3 2 (2 3 0)
P_1	3 2 2	2 0 0 (3 0 2)	1 2 2 (0 2 0)	
P_2	9 0 2	3 0 2	6 0 0	
P_3	2 2 2	2 1 1	0 1 1	
P_4	4 3 3	0 0 2	4 3 1	

考点四: 死锁

③ 系统先假定可为P_1分配资源，并修改Available, $Allocation_1$和$Need_1$向量，由此形成的资源变化情况如图中的圆括号所示。

进程 \ 资源情况	Max A B C	Allocation A B C	Need A B C	Available A B C
P_0	7 5 3	0 1 0	7 4 3	3 3 2 (2 3 0)
P_1	3 2 2	2 0 0 (3 0 2)	1 2 2 (0 2 0)	
P_2	9 0 2	3 0 2	6 0 0	
P_3	2 2 2	2 1 1	0 1 1	
P_4	4 3 3	0 0 2	4 3 1	

考点四: 死锁

④ 再利用安全性算法检查此时系统是否安全。

进程 \ 资源情况	Work A	Work B	Work C	Need A	Need B	Need C	Allocation A	Allocation B	Allocation C	Work+Allocation A	Work+Allocation B	Work+Allocation C	Finish
P_1	2	3	0	0	2	0	3	0	2	5	3	2	true
P_3	5	3	2	0	1	1	2	1	1	7	4	3	true
P_4	7	4	3	4	3	1	0	0	2	7	4	5	true
P_0	7	4	5	7	4	3	0	1	0	7	5	5	true
P_2	7	5	5	6	0	0	3	0	2	10	5	7	true

考点四: 死锁

(3) P_4请求资源：P_4发出请求向量**$Request_4(3，3，0)$**，系统按银行家算法进行检查：

① $Request_4(3, 3, 0) \leq Need_4(4, 3, 1)$;

② $Request_4(3, 3, 0) \leq$**$Available(2, 3, 0)$**，让P_4等待。

考点四: 死锁

(4) P_0请求资源：P_0发出请求向量**$Requst_0(0，2，0)$**，系统按银行家算法进行检查：

① $Request_0(0, 2, 0) \leq Need_0(7, 4, 3)$;

② $Request_0(0, 2, 0) \leq$ **$Available(2, 3, 0)$;**

③ 系统暂时先假定可为P_0分配资源，并修改有关数据，如图所示。

进程 \ 资源情况	Max A B C	Allocation A B C	Need A B C	Available A B C
P_0	7 5 3	0 1 0	7 4 3	3 3 2 (2 3 0)
P_1	3 2 2	2 0 0 (3 0 2)	1 2 2 (0 2 0)	
P_2	9 0 2	3 0 2	6 0 0	
P_3	2 2 2	2 1 1	0 1 1	
P_4	4 3 3	0 0 2	4 3 1	

考点四: 死锁

(4) P_0请求资源：P_0发出请求向量**$Requst_0(0，2，0)$**，系统按银行家算法进行检查：

① $Request_0(0, 2, 0) \leq Need_0(7, 4, 3)$;

② $Request_0(0, 2, 0) \leq$ **$Available(2, 3, 0)$;**

③ 系统暂时先假定可为P_0分配资源，并修改有关数据，如图所示。

进程 \ 资源情况	Allocation A B C	Need A B C	Available A B C
P_0	0 3 0	7 2 3	2 1 0
P_1	3 0 2	0 2 0	
P_2	3 0 2	6 0 0	
P_3	2 1 1	0 1 1	
P_4	0 0 2	4 3 1	

考点四: 死锁

死锁检测与解除

考点四: 死锁

- 系统发生死锁时不会对用户造成多大影响，或者系统很少发生死锁时允许采用鸵鸟策略。
- 解决死锁的问题通常代价很大，这样开销比不允许发生死锁及检测和解除死锁的小。因此鸵鸟策略是平衡性能和复杂性的一种方法。
- 如果系统中既不采取预防死锁的措施，也不采取避免死锁的措施，系统就很可能发生死锁。

考点四: 死锁

- 在这种情况下，系统应当提供两个算法:
- 死锁检测算法:用于检测系统状态，以确定系统中是否发生了死锁。
- 死锁解除算法:当认定系统中已经发生了死锁，利用该算法可将系统从死锁状态中解脱出来。

考点四: 死锁

死锁检测：通过系统设置的检测机构，**及时检测死锁的发生**，精确**确定**与死锁**有关**的**进程**和**资源**，采取适当措施，**清除**已发生**死锁**的**进程**

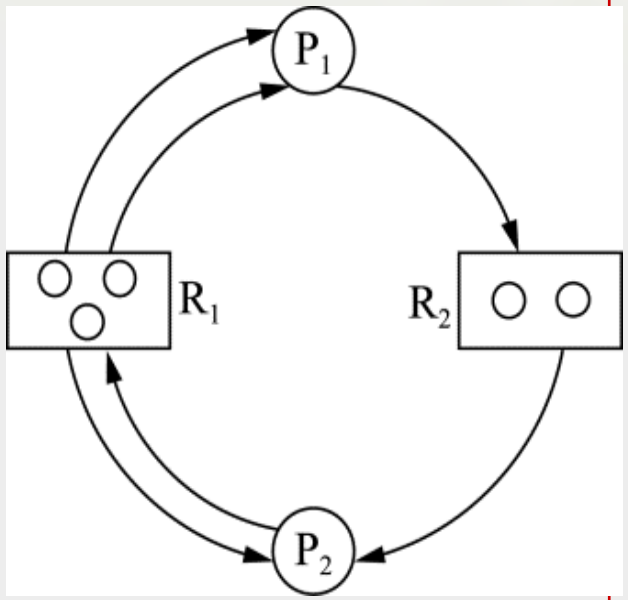
- 条件：系统必须保存有关资源的请求和分配信息
- 方法：资源分配图

考点四: 死锁

死锁检测：资源分配图描述系统死锁情况

G = (N, E)，其中G为图，N表示G的顶点集合，E表示G的边集合

- N分为两个互斥的子集：进程集P={P1, …, Pm}，资源集R={R1, …, Rn}
- e∈E，连接P的一个节点和R的一个节点
- e1=<Pi, Rj>：资源请求边，表示进程Pi请求一个单位的Rj资源
- e2 = <Rj, Pi>：资源分配边，表示把一个单位的资源Rj分配给进程Pi

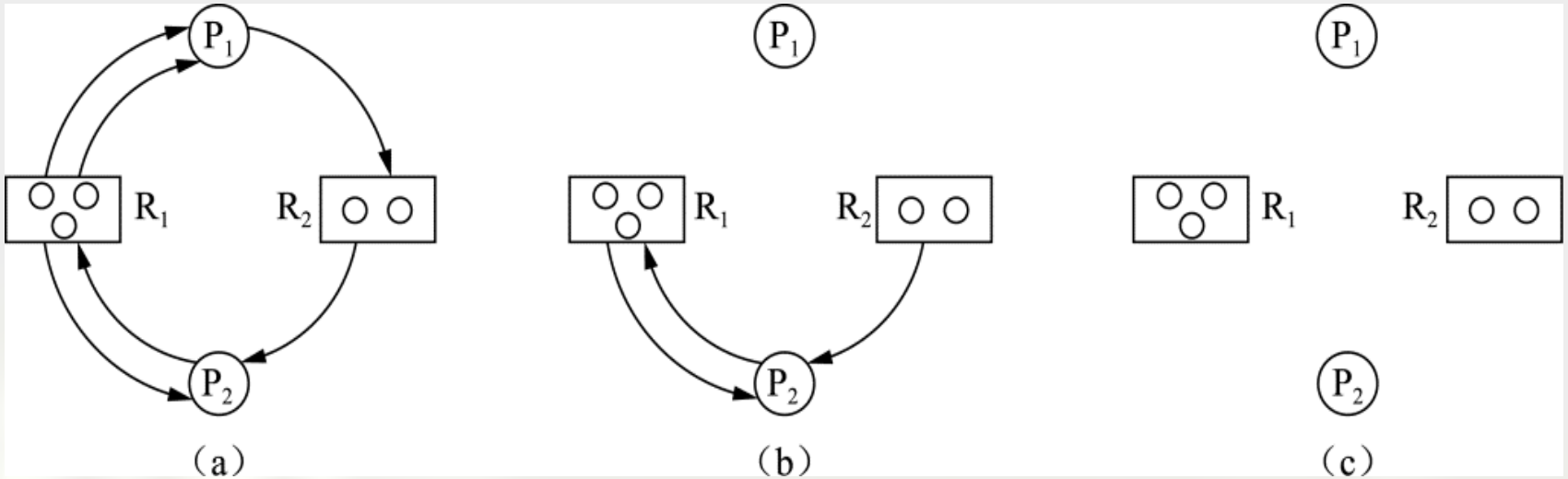


考点四: 死锁

检测定理：检测定理，利用**简化资源分配图**检测当前系统是否处于死锁状态

- 在资源分配图中找出一个既不阻塞又非独立的进程节点P_i
- 顺利时，P_i可获得所需资源而继续运行，直至完成，释放其占有的全部资源，相当于消去P_i的请求边和分配边，使之成为孤立节点

➢ 所有进程均可简化为孤立结点，则没有死锁，否则死锁



考点四: 死锁

死锁解除：当**检测到**系统中已发生**死锁**时，须将**进程**从死锁状态中**解脱**出来

- 和死锁检测相配套的措施
- 实施方法：**回收一些资源**，再将这些资源**分配**给处于**阻塞**状态的**进程**，使之**转为就绪状态**继续**运行**

考点四: 死锁

- 抢占资源：强行剥夺部分处于死锁状态的进程所占用的资源，并将其分配给其它处于死锁状态的进程，使之能够继续运行，从而解除死锁

考点四: 死锁

- 撤销/挂起/终止进程
  - 终止所有处于死锁状态的进程：简单，代价可能会很大
  - 逐个终止进程：依次选择付出代价最小的进程予以终止，直至有足够的资源打破循环等待，将系统从死锁状态解脱出来为止

考点四: 死锁

- 付出代价最小的死锁解除算法：将解除进程的代价按升序排列，每次终止队首进程，直至死锁解除

【牛刀小试】

1．【广东工业大学 2014】产生死锁的主要原因是进程运行推进的顺序不合适，（ ）。

A．系统资源不足和系统中的进程太多

B．资源的独占性和系统中的进程太多

C．进程调度不当和资源的独占性

D．资源分配不当和系统资源不足

D【**解析**】产生死锁的主要原因：

①系统资源不足；

②进程运行推进的顺序不合适；

③资源分配不当。因此选择D。

2．【重庆理工大学 2017】死锁和安全状态的关系是(　)。

A．死锁状态有可能是安全状态　B．死锁状态一定是不安全状态

C．安全状态也可能是死锁状态　D．不安全状态必定产生死锁

- 如果系统无法找到这样一个安全序列，则称系统处于不安全状态
- 只要处于安全状态就不会发生死锁
- 并不是所有的不安全状态都会导致系统进入死锁状态，但当系统进入不安全状态后，可能进入死锁状态

2．【重庆理工大学 2017】死锁和安全状态的关系是(　)。

A．死锁状态有可能是安全状态　B．死锁状态一定是不安全状态

C．安全状态也可能是死锁状态　D．不安全状态必定产生死锁

B【解析】安全状态一定不死锁，死锁状态一定是不安全状态，不安全状态不一定死锁，因此选择B。

3．【燕山大学　2012】进程因资源不足发生死锁，解除这个死锁可使用的最好方法是(　)。

A．重新启动系统　　B．挂起其他进程

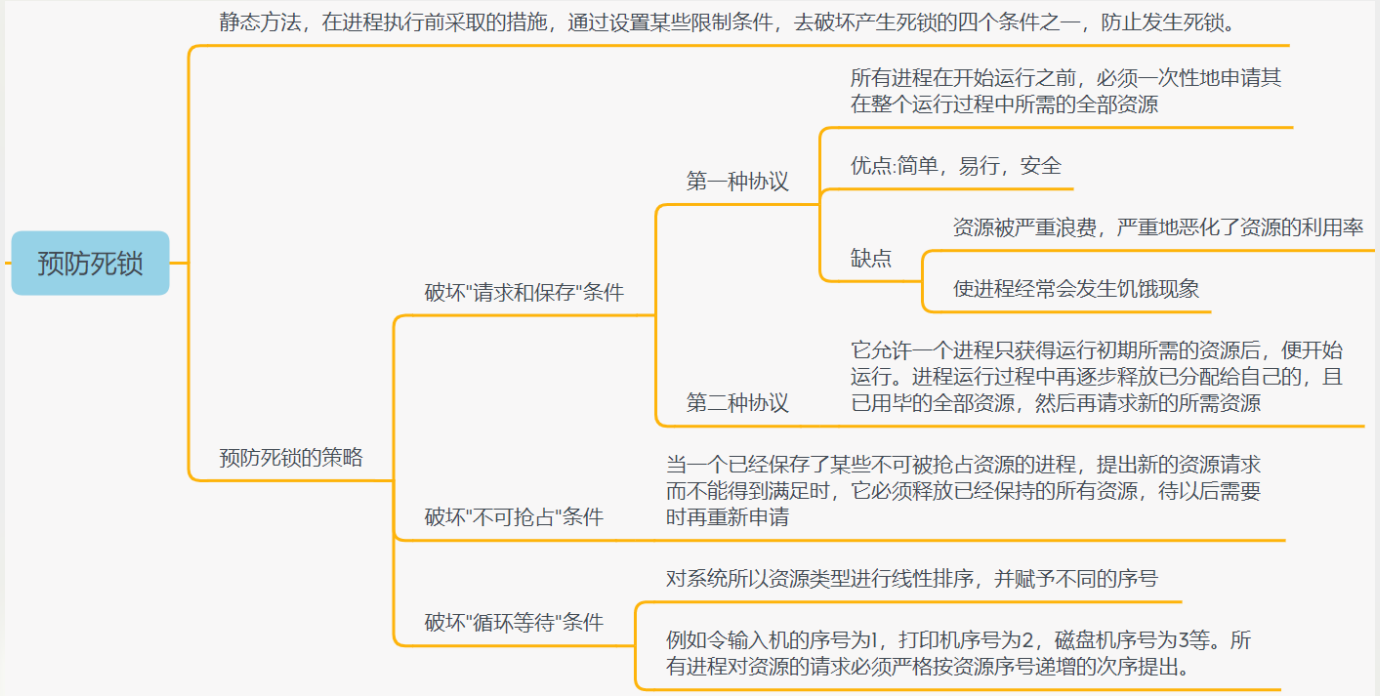
C．挂起死锁进程　　D．撤销死锁进程

C【**解析**】资源剥夺：挂起某些死锁进程，并抢占它的资源，将这些资源分配给其他的死锁进程。但应防止被挂起的进程长时间得不到资源，而处于资源匮乏状态。因此选择C。

4．【西北大学　2015】把资源按类型排序编号，并要求进程严格按序申请资源，这种方法破坏了死锁四个必要条件中的哪一个条件？（　）

A．互斥条件　B．请求与保持条件

C．不可剥夺条件　D．循环等待条件



4．【西北大学　2015】把资源按类型排序编号，并要求进程严格按序申请资源，这种方法破坏了死锁四个必要条件中的哪一个条件？（　）

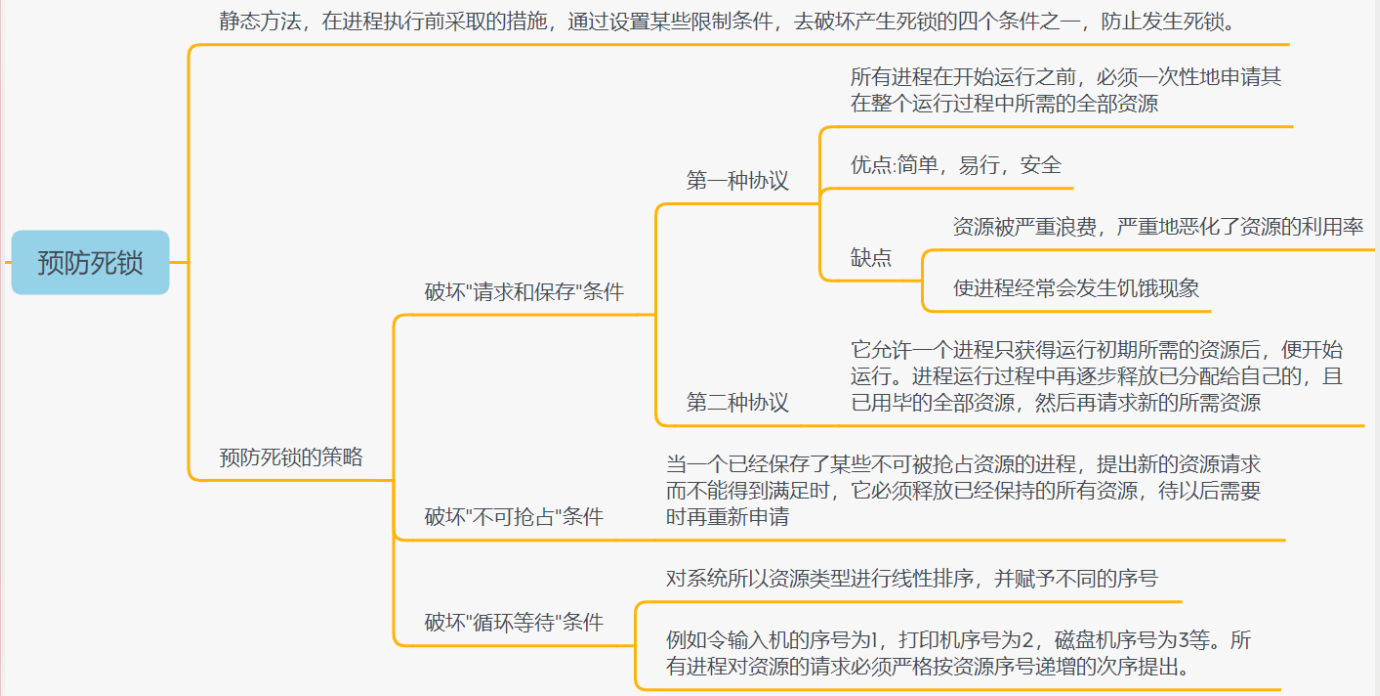
A．互斥条件　B．请求与保持条件

C．不可剥夺条件　D．循环等待条件

D【**解析**】所有进程资源的请求必须严格按照资源序号递增的次序提出，在所形成的资源分配图中不可能再出现环路，因而破坏了循环等待条件。因此选择D。

5．【西北大学　2018】采用按序资源分配的死锁预防方法是破坏了发生死锁的必要条件中的(　)条件。

A．互斥　　B．循环等待　　C．占有且等待　　D．不可剥夺



5．【西北大学　2018】采用按序资源分配的死锁预防方法是破坏了发生死锁的必要条件中的(　)条件。

A．互斥　　B．循环等待　　C．占有且等待　　D．不可剥夺

B【**解析**】所有进程资源的请求必须严格按照资源序号递增的次序提出，在所形成的资源分配图中不可能再出现环路，因而破坏了循环等待条件。因此选择B。

6．【汕头大学 2016】下列关于银行家算法的叙述中，正确的是(　)。

A．银行家算法破坏了死锁必要条件中的“请求和保持”条件

B．当系统处于安全状态时，系统中一定不会出现死锁进程

C．当系统处于不安全状态时，系统中一定会出现死锁进程

D．以上答案都不对

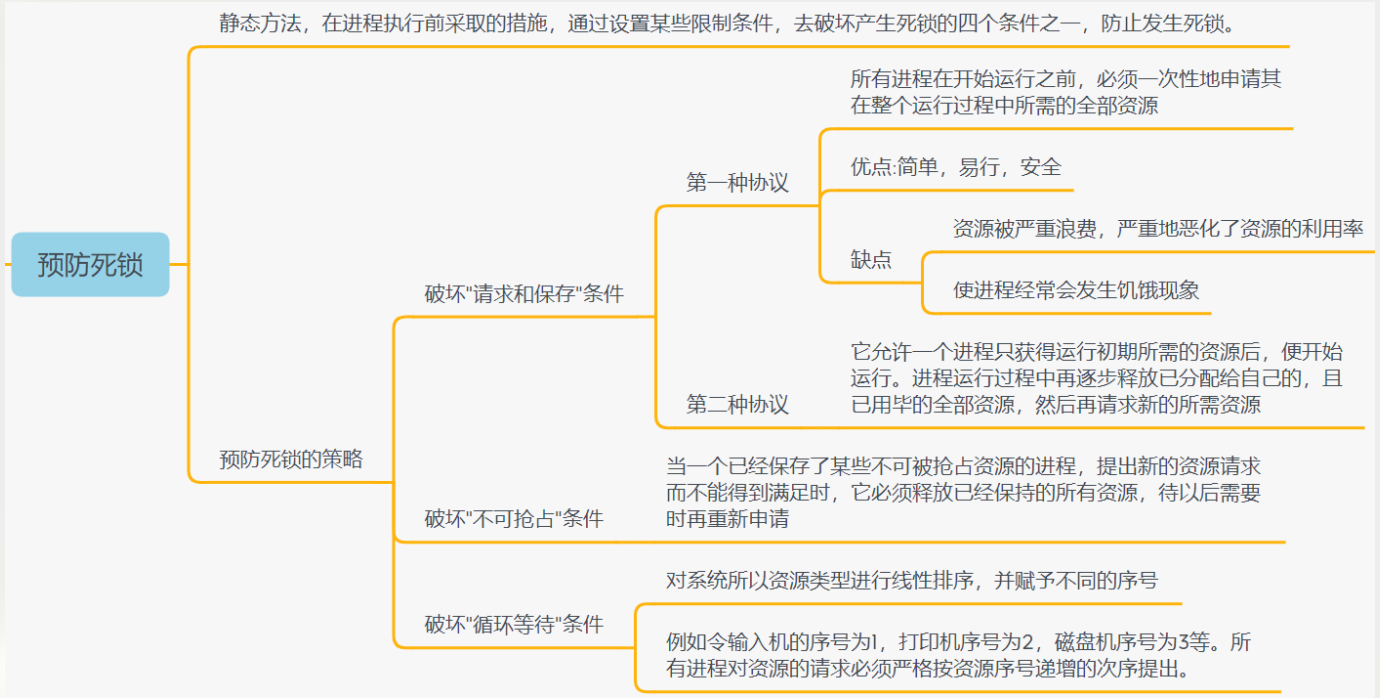
B【**解析**】

- 银行家算法是一种最有代表性的避免死锁的算法，破坏死锁产生的四个必要条件是预防死锁，选项A错误；
- 当系统处于安全状态时，系统中一定不会出现死锁进程，选项B正确；
- 当系统处于不安全状态时，系统中不一定会出现死锁进程，选项C错误。
- 因此选择B。

7．【南京理工大学　2017】下列预防死锁的方法中，(　　)破坏了产生死锁的四个必要条件之一“循环等待”。

A．银行家算法　　B．资源有序分配策略

C．剥夺资源法　　D．一次性分配策略



7．【南京理工大学　2017】下列预防死锁的方法中，(　　)破坏了产生死锁的四个必要条件之一“循环等待”。

A．银行家算法　　B．资源有序分配策略

C．剥夺资源法　　D．一次性分配策略

B【**解析**】银行家算法是避免死锁的算法；资源有序分配策略破坏了“循环等待”条件；剥夺资源法破坏了“不可剥夺”条件；一次性分配策略破坏了“请求与保持”条件。因此选择B。

8．【南京理工大学　2018】如果系统的资源分配图(　　)，则系统处于死锁状态。

A．出现了环路

B．每类资源只有一个，且出现了环路

C．每个进程节点都至少有一条请求边

D．上面三种情况都不对

B【**解析**】当每个资源类型只有一个时，有环等价于死锁。当资源类型有多个时，死锁必有环，但有环不一定死锁。因此选择B。

9．【华东师范大学 2013】(多选)对于死锁，以下哪个描述是正确的？（ ）

A．死锁避免(deadlock avoidance)中，不安全的状态必然发生死锁

B．死锁避免(deadlock avoidance)中，发生死锁必然处于不安全状态

C．资源分配图中有环（以资源类型和进程为节点），必然发生死锁

D．如果要求每个进程必须一次申请所有需要的资源，如果不能满足其要求，则不分配任何资源，那么死锁不可能发生

BD【**解析**】死锁避免中，不安全的状态可能发生死锁，发生死锁必然处于不安全状态；

资源分配图中有环（以资源类型和进程为节点），可能发生死锁；

死锁产生的四个必要条件：互斥条件、请求与保持条件、不可剥夺条件和循环等待条件，破坏其中的一条即可预防死锁，每个进程必须一次申请所有需要的资源，破坏了请求与保持条件，不可能发生死锁。因此选择BD。

10．死锁的避免是根据(　)采取措施实现的。

A．配置足够的系统资源　　B．使进程的推进顺序合理

C．破坏死锁的四个必要条件之一　　D．防止系统进入不安全状态

D【**解析**】避免死锁的基本思想是在资源的动态分配过程中，用某种方法防止系统进入不安全状态。因此选择D。

11．解除死锁通常不采用的方法是(　)。

A．终止一个死锁进程　　B．终止所有死锁进程

C．从死锁进程处抢夺资源　D．从非死锁进程处抢夺资源

D【**解析**】解除死锁的方法有两个：

- ①剥夺资源，从其他进程剥夺足够数量的资源给死锁进程，但一般是从其他死锁进程处抢夺资源，而不是从非死锁进程处抢夺资源，这样有可能会造成新的死锁；②撤销进程，一种是终止全部死锁进程，另一种是按某种规则逐个终止死锁进程。因此选择D。

12．死锁的四个必要条件中，无法破坏的是(　)。

A．循环等待条件　　B．互斥条件

C．请求与保持条件　D．不可剥夺条件

B【**解析**】所谓破坏互斥条件，指允许多个进程同时访问资源，但有些资源根本不能同时访问，如打印机只能互斥使用。所以破坏互斥条件而预防死锁的方法不太可行，而且在有些场合应该保护这种互斥性。其他三个条件都可以实现，因此选择B。

13．下面是一个并发进程的程序代码，下列说法中正确的是(　　)。

```
1   semaphore x1=x2=y=1;
2   int c1=c2=0;
3   P1()                          P2()
4   {                             {
5     while (1){                    while (1){
6       P(x1);                        P(x2);
7       if(++c1==1)P(y);              if(++c2==1)P(y);
8       V(x1);                        V(x2);
9       computer (A);                 computer (B);
10      P(x1);                        P(x2);
11      if(--c1==0)v(y);              if(--c2==0)v(y);
12      V(x1);                         V(x2);
13      }                              }
14    }                             }
```

A．进程不会死锁，也不会“饥饿”

B．进程不会死锁，但是会“饥饿”

C．进程会死锁，但是不会“饥饿”

D．进程会死锁，也会“饥饿”

B【**解析**】仔细观察程序代码，可以看出是一个扩展的单行线问题。也就是说，某单行线只允许单方向的车辆通过，在单行线的入口设置信号量y，在告示牌上的车辆必须互斥进行，为此设置信号量x1和x2。若某方向的车辆需要通过时，首先要将该方向来车数量c1或c2增加1，并查看自己是不是第一个进入单行线的车辆，若是，则获取单行线的信号量y，并进入单行线。通过此路段以后驶出单行线时，将该方向的车辆数c1或c2减1（当然是利用x1或x2来互斥修改），并查看自己是不是最后一辆车，若是，则释放单行线的互斥量y，否则保留信号量y，让后继车辆继续通过。双方的操作如出一辙。考虑出现一种极端情况，即当某方向的车辆首先占据单行线且后来者络绎不绝时，另一个方向的车辆就没有机会通过该单行线了，从而造成“饥饿”。由于有信号量的控制，死锁的可能性就没有了（即双方同时进入单行线，在中间相遇，造成双方均无法通过的情景），因此选择B。

14．一个进程在获得资源后，只能在使用完资源后由自己释放，这属于死锁必要条件的(　)。

A．互斥条件　　B．请求和保持条件

C．不可剥夺条件　　D．循环等待条件

- 互斥使用：进程对所分配到资源的使用具有排他性，即在一段时间内某资源只由一个进程占用。如果此时还有其它进程请求该资源，则请求者只能等待，直至占有该资源的进程使用完毕将其释放。由设备的固有条件决定，应予以保护
- 请求保持：进程已占用至少一个资源，又申请被其它进程占有不释放的资源，则请求进程阻塞，但不释放已占用资源
- 不可抢占：进程已获得的资源在使用完之前，不能被其他进程抢占，只能在使用完后由自己释放
- 循环等待：发生死锁时，必存在一个进程—资源循环链

14．一个进程在获得资源后，只能在使用完资源后由自己释放，这属于死锁必要条件的(　)。

A．互斥条件　　B．请求和保持条件

C．不可剥夺条件　　D．循环等待条件

C【解析】所谓破坏互斥条件，指允许多个进程同时访问资源，但有些资源根本不能同时访问，如打印机只能互斥使用。所以破坏互斥条件而预防死锁的方法不太可行，而且在有些场合应该保护这种互斥性。其他三个条件都可以实现，因此选择C。

15．【浙江工商大学　2018】设系统中物理页的数量为150。在T_0时刻系统状态按下表所示分配给三个进程P_1，P_2和P_3。系统采用银行家算法实施死锁避免策略。对下列内存请求，请分别判断是否安全，如果是安全的，请给出一个可能的进程安全执行序列；如果不是安全的，请说明原因。

(1)进程P_4到达，共需要60个物理页，当前请求分配25页。

进程	最大物理页需求量	已分配物理页数量
P_1	70	25
P_2	60	40
P_3	60	45

(1)进程P_4到达，共需要60个物理页，当前请求分配25页。

进程	最大物理页需求量	已分配物理页数量	尚需
P_1	70	25	45
P_2	60	40	20
P_3	60	45	15

(1) T_0时刻，系统中物理页剩余数量为150-(25+40+45)=40，进程P_1还需物理页的数量为70-25=45，进程P_2还需物理页的数量为60-40=20，进程P_3还需物理页的数量为60-45=15，

(2) 此时进程P_4到达，分配25个物理页给进程P_4，进程P_4还需物理页的数量为60-25=35，

(1)进程P_4到达，共需要60个物理页，当前请求分配25页。

进程	最大物理页需求量	已分配物理页数量	尚需
P_1	70	25	45
P_2	60	40	20
P_3	60	45	15

(1) 此时系统的物理页数量为40-25=15，系统的物理页数量满足进程P_3，分配给P_3再收回来，系统的物理页数量为60，

(2) 系统的物理页数量满足进程P_2，分配给P_2再收回来，此时系统的物理页数量为60+40=100，

(3) 系统的物理页数量满足进程P_4，分配给P_4再收回来，此时系统的物理页数量为100+25=125，

(4) 系统的物理页数量满足进程P_1，分配给进程P_1再收回来，

(5) 此时系统的物理页数量为125+25=150，因此进程安全执行的序列可能为P_3，P_2，P_4，P_1。

15．【浙江工商大学　2018】设系统中物理页的数量为150。在T_0时刻系统状态按下表所示分配给三个进程P_1，P_2和P_3。系统采用银行家算法实施死锁避免策略。对下列内存请求，请分别判断是否安全，如果是安全的，请给出一个可能的进程安全执行序列；如果不是安全的，请说明原因。

(2)进程P_4到达，共需要50个物理页，当前请求分配35页。

进程	最大物理页需求量	已分配物理页数量
P_1	70	25
P_2	60	40
P_3	60	45

(2)进程P_4到达，共需要50个物理页，当前请求分配35页。

进程	最大物理页需求量	已分配物理页数量	尚需
P_1	70	25	45
P_2	60	40	20
P_3	60	45	15

- T_0时刻，系统中物理页剩余数量为150-(25+40+45)=40，
- 进程P_1还需物理页的数量为70-25=45，
- 进程P_2还需物理页的数量为60-40=20，
- 进程P_3还需物理页的数量为60-45=15，
- 此时进程P_4到达，分配35个物理页给进程P_4，进程P_4还需物理页的数量为50-35=15，
- 此时系统的物理页数量为40-35=5，系统的物理页数量不满足任何进程还需的物理页数量，因此找不到安全序列，故为不安全状态。

16．【吉林大学　2016】某系统采用银行家算法避免死锁，设某时刻剩余可用资源Available=(1，1，2)，其他变量状态如下表所示：

进程	Max			Allocation			Need		
	A	B	C	A	B	C	A	B	C
P_1	3	2	2	1	0	0			
P_2	6	1	3	5	1	1			
P_3	3	1	4	2	1	1			
P_4	5	2	2	1	0	2			

(1) 系统各类资源分别共有多少?

【解析 】 (1)系统剩余可用资源为(1,1,2)，已分配的资源为(1,0,0)+(5,1,1)+(2,1,1)+(1,0,2)=(9,2,4)，则系统各类资源分别共有(1,1,2)+(9,2,4)=(10,3,6)。

16．【吉林大学　2016】某系统采用银行家算法避免死锁，设某时刻剩余可用资源Available=(1，1，2)，其他变量状态如下表所示：

进程	Max			Allocation			Need		
	A	B	C	A	B	C	A	B	C
P_1	3	2	2	1	0	0			
P_2	6	1	3	5	1	1			
P_3	3	1	4	2	1	1			
P_4	5	2	2	1	0	2			

(2) 计算Need矩阵的值，说明当前状态安全。

(2)进程P_1还需资源为(3,2,2)-(1,0,0)=(2,2,2)，

进程P_2还需资源为(6,1,3)-(5,1,1)=(1,0,2)，

进程P_3还需资源为(3,1,4)-(2,1,1)=(1,0,3)，

进程P_4还需资源为(5,2,2)-(1,0,2)=(4,2,0)，

系统剩余可用资源为(1,1,2)，

16．【吉林大学　2016】某系统采用银行家算法避免死锁，设某时刻剩余可用资源Available=(1，1，2)，其他变量状态如下表所示：

进程	Max			Allocation			Need		
	A	B	C	A	B	C	A	B	C
P_1	3	2	2	1	0	0			
P_2	6	1	3	5	1	1			
P_3	3	1	4	2	1	1			
P_4	5	2	2	1	0	2			

(2) 计算Need矩阵的值，说明当前状态安全。

满足进程P_2的需求，将资源分配给P_2再收回，此时系统剩余可用资源为(1,1,2)+(5,1,1)=(6,2,3)，

满足进程P_1的需求，将资源分配给P_1再收回，此时系统剩余可用资源为(6,2,3)+(1,0,0)=(7,2,3)，

满足进程P_3的需求，将资源分配给P_3再收回，此时系统剩余可用资源为(7,2,3)+(2,1,1)=(9,3,4)，

满足进程P_4的需求，将资源分配给P_4再收回，此时系统剩余可用资源为(9,3,4)+(1,0,2)=(10,3,6)，则安全序列为P_2，P_1，P_3，P_4。

17．有三个进程P_1，P_2和P_3并发工作。进程P_1需要资源S_3和资源S_1；进程P_2需要资源S_2和资源S_1；进程P_3需要资源S_3和资源S_2。问：

(1)若对资源分配不加限制，会发生什么情况？为什么？

【解析】 (1)若对资源分配不加限制，上述进程可能会进入死锁状态。因为P_1，P_2，P_3是并发工作的，所以完全有可能发生下述情况：P_1获得了S_3，P_2获得了S_1，P_3获得了S_2，但同时它们又循环等待下一个进程释放所占用的资源，从而进入死锁状态。

17．有三个进程P_1，P_2和P_3并发工作。进程P_1需要资源S_3和资源S_1；进程P_2需要资源S_2和资源S_1；进程P_3需要资源S_3和资源S_2。问：

(2)为保证进程正确运行，应采用怎样的分配策略？列出所有可能的方法。

(2)为保证进程正确运行，可采取下列资源分配策略：

①要求进程一次性申请它所需的全部资源。这样可破坏“请求与保持”条件。

②要求进程严格按资源号递增的顺序申请资源，即P_1先申请S_1，再申请S_3；P_2先申请S_1，再申请S_2；P_3先申请S_2，再申请S_3。这样可破坏“循环等待”条件。

谢谢大家

【真题实战】

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_____。

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

D【解析】选项中的算法都是维护一个进程的就绪队列，队列内进程按照不同条件进行排序，而CPU总是最先处理队列的首元素。

C先来先服务算法中的进程是按到达的顺序先后进入队列，队首的进程到达最早，等待时间也最长，C项只考虑了等待时间，不符合。

A项时间片轮转算法是指系统设定一个固定的短时间段处理进程，每当时间段用完，系统就产生一次中断，轮换到下一个进程，进程轮换的依据一般是先来的进程先轮转。这样看来，时间片轮转算法也与进程的等待时间而非执行时间有关，A项不符合。

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_____。

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

D【解析】选项中的算法都是维护一个进程的就绪队列，队列内进程按照不同条件进行排序，而CPU总是最先处理队列的首元素。

B项短进程优先算法中的进程是按执行时间的长短排序进入队列，队首的进程执行时间最短，B项只考虑了执行时间，不符合。

D项高响应比优先算法的响应比=（等待时间+执行时间）/执行时间，进程是按照响应比的大小排序进入队列，队首进程的响应比最高。由式子知该算法既考虑了等待时间也考虑了执行时间，符合题目要求，选择D项。

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_____。

A．2　　B．3　　C．4　　D．5

C 【解析】解法一：死锁是指多个进程在对资源进行争夺时，每个进程都需要集合中其他进程触发某种事件才能继续运行的一种状态。

在本题中，只要有一个进程可以拿到3台打印机，那么其就可以顺利运行，并释放资源供其他进程使用。所以，要使系统死锁，那么系统中的进程能拿到的打印机要<3台打印机。同时要使K最小，那也就需要每个进程能拿到的打印机取最大值，也就是2台打印机，一共8台打印机，K最小为8/2=4台，选C。

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_____。

A．2　　B．3　　C．4　　D．5

C 【解析】解法二：K=4时，尽力均分的情况是每个进程2台，都需要再获得1台才能顺利执行，而它们都占有着自己的2台不释放，这种情况如果没有外力的介入，进程就一直处于等待状态，符合死锁状态，C正确。K=5时，均分的情况是1、1、2、2、2，也类似于C项的情况，属于死锁状态，但不是K的最小值，D不符合。所以选C。在使用代入法求最大/最小值时，要由大到小/由小到大代入，第一个满足条件的选项即为正确选项，避免了多余计算。

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_____。

A．2　　B．3　　C．4　　D．5

C 【解析】解法二：代入法。如果K=2，那么这两个进程都可以拿到3台打印机并顺利执行，不会产生死锁，A不正确。如果K=3，要想出现死锁，就要尽力均匀分。8/3=2……2，也就是说最均匀的情况下，每个进程都分到2台打印机，还剩2台打印机。任意进程都可以申请剩下打印机中的一台，并顺利执行。即使可能存在某个进程无法再申请到打印机，但是其他两台打印机一定能执行完毕并归还资源，归还后的资源十分充足，完全够等待中的进程顺利执行，B不正确。

3、设与某资源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_____。

A．0、1　　B．1、0　　C．1、2　　D．2、0

B【解析】信号量是用来同步进程并发访问共享资源的一种机制，其初始值为 3，当前值为 1，表示该资源已经被占用 2 个，还有 1 个可用，那么没有等待该资源的进程。因此，该资源的可用个数 M=1，等待该资源的进程数 N=0。因此，选项 B 正确。

4、下列选项中，满足短任务优先且不会发生饥饿现象的调度算法是_____。

A．先来先服务　　B．高响应比优先

C．时间片轮转　　D．非抢占式短任务优先

B【解析】响应比=作业响应时间/作业执行时间 =(作业执行时间+作业等待时间)/作业执行时间。高响应比算法，在等待时间相同情况下，作业执行时间越少，响应比越高，优先执行，满足短任务优先。随着等待时间增加，响应比也会变大，执行机会就增大，所以不会产生饥饿现象。其他算法总结如下

	先来先服务	高响应比优先	时间片轮转	非抢占式短任务优先
饥饿	否	否	否	是
短任务优先	否	是	否	是

5、某时刻进程的资源使用情况如下表所示。

进程	已分配资源			尚需分配			可用资源		
	R1	R2	R3	R1	R2	R3	R1	R2	R3
P1	2	0	0	0	0	1	0	2	1
P2	1	2	0	1	3	2			
P3	0	1	1	1	3	1			
P4	0	0	1	2	0	0			

此时的安全序列是_____。

A．P1,P2,P3,P4　　B．P1,P3,P2,P4

C．P1,P4,P3,P2　　D．不存在

D【解析】根据银行家算法过程，可用资源能够让P1执行完成，并回收P1的资源，系统可用资源是（2，2，1）。可用资源能够让P4执行完成，并回收P4的资源，系统可用资源是（2，2，2）。

通过比较发现，此时已 无法满足任何进程的需求，因此不存在安全序列。

6、假设 5 个进程 P0、P1、P2、P3、P4 共享三类资源 R1、R2、R3，这些资源总数分别为 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（）。

进程	已分配资源			资源最大需求		
	R1	R2	R3	R1	R2	R3
P0	3	2	3	5	5	10
P1	4	0	3	5	3	6
P2	4	0	5	4	0	11
P3	2	0	4	4	2	5
P4	3	1	4	4	2	4

A. P0, P2, P4, P1, P3　　B. P1, P0, P3, P4, P2

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

6、假设 5 个进程 P0、P1、P2、P3、P4 共享三类资源 R1、R2、R3，这些资源总数分别为 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（）。

进程	已分配资源			资源最大需求		
	R1	R2	R3	R1	R2	R3
P0	3	2	3	5	5	10
P1	4	0	3	5	3	6
P2	4	0	5	4	0	11
P3	2	0	4	4	2	5
P4	3	1	4	4	2	4

D【解析】根据题中给出的条件，可得到资源的分配情况如图所示，

进程	已分配资源			资源最大需求			还需要资源			可用资源		
	R1	R2	R3	R1	R2	R3	R1	R2	R3	R1	R2	R3
P0	3	2	3	5	5	10	2	3	7			
P1	4	0	3	5	3	6	1	3	3			
P2	4	0	5	4	0	11	0	0	6	2	3	3
P3	2	0	4	4	2	5	2	2	1			
P4	3	1	4	4	2	4	1	1	0			

D【解析】根据题中给出的条件，可得到资源的分配情况如图所示，

进程	已分配资源			资源最大需求			还需要资源			可用资源		
	R1	R2	R3	R1	R2	R3	R1	R2	R3	R1	R2	R3
P0	3	2	3	5	5	10	2	3	7			
P1	4	0	3	5	3	6	1	3	3			
P2	4	0	5	4	0	11	0	0	6	2	3	3
P3	2	0	4	4	2	5	2	2	1			
P4	3	1	4	4	2	4	1	1	0			

(R1、R2、R3)资源的总数为(18、6、22)。系统可以将资源分配给P1或者P3，因此A和C错误。

（1）假设首先分配给P3。当P3执行结束后，系统所剩的资源量为(4、3、7)，可以分配给任意一个进程都能执行结束。即P3、P4、P2、P1、P0是安全序列，因此D正确。

（2）假设首先分配给P1。当P1执行结束后，系统所剩的资源量为(6、3、6)，该资源不够分配给P0，因此B错误。

7、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_____。

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机调度

D. 在系统调用完成并返回用户态时能进行处理机调度

C【解析】选项A、B、D都是常见的可以进行处理机调度的情况。

对于C，当进程处于临界区时，说明进程正在占用处理机，可以进行剥夺该进程的处理机，并进行调度，但是由于改成处于临界区，其他进程不能进入临界区。因此C选项错误。这里混淆了临界区和处理机调度的概念。

8、某系统正在执行三个进程 P1、P2 和 P3，各进程的计算（CPU）时间和 I/O 时间比例如下表所示。

进程	计算时间	I/O 时间
P1	90%	10%
P2	50%	50%
P3	15%	85%

为提高系统资源利用率，合理的进程优先级设置应为_____。

A. P1 > P2 > P3　　B. P3 > P2 > P1

C. P2 > P1 = P3　　D. P1 > P2 = P3

B【**解析**】为了合理地设置进程优先级，应该将进程的 CPU 利用时间和 I/O 时间做综合考虑。 为了合理地设置进程优先级，应该将进程的 CPU 时间和 I/0 时间做综合考虑，对于 CPU 占用时间较少而 I/0 占用时间较多的进程，优先调度能让 I/0 更早地得到使用，提高了系统的 资源利用率，显然应该具有更高的优先级。答案选B。

9、下列关于银行家算法的叙述中，正确的是_____。

A. 银行家算法可以预防死锁

B. 当系统处于安全状态时，系统中一定无死锁进程

C. 当系统处于不安全状态时，系统中一定会出现死锁进程

D. 银行家算法破坏了死锁必要条件中的“请求和保持”条件

B【解析】银行家算法是避免死锁的方法，不是死锁预防算法，也不是破坏死锁产生的必要条件，因此A和D错误。

利用银行家算法，确保系统始终处于安全状态，那么一定没有死锁进程。

当系统进入不安全状态后，由于系统中还可能存在可用资源，并不一定会立即进入死锁状态，但是最终一定出现死锁现象，因此B正确，C错误。

10、下列调度算法中，不可能导致饥饿现象的是_____。

A．时间片轮转

B．静态优先数调度

C．非抢占式短作业优先

D．抢占式短作业优先

A【解析】 静态优先调度采用静态优先级作为调度准则，那么优先级低的进程可能将会得不到调度而饥饿。

短作业优先调度，不管是抢占式还是非抢占式，都是优先调度运行时间短的作业，那么运行时间长的作业可能将得不到调度而产生饥饿。时间片轮转算法采用轮询的模式，各个作业轮流调度，因此不会产生饥饿。

11、某系统有 n 台互斥使用的同类设备，三个并发进程分别需要 3、4、5 台设备，可确保系统不发生死锁的设备数 n 最小为_____。

A．9　　B．10　　C．11　　D．12

B【解析】三个并发进程分别需要 3、 4、 5 台设备，为了确保不发生死锁，可以给第一个进程分配 2 台，第二个进程分配 3 台，第三个进程分配 4 台。

当系统中再增加 1 台设备这 最后 1 台设备分配给任意一个进程都可以顺利执行完成，因此保证系统不发生死锁的最小设备数为10。

12、若系统S1采用死锁避免方法，S2采用死锁检测方法。下列叙述中，正确的是_____。

Ⅰ．S1 会限制用户申请资源的顺序，而 S2 不会

Ⅱ．S1 需要进程运行所需资源总量信息，而 S2 不需要

Ⅲ．S1 不会给可能导致死锁的进程分配资源，而 S2 会

A．仅Ⅰ、Ⅱ　　B．仅Ⅱ、Ⅲ

C．仅Ⅰ、Ⅲ　　D．Ⅰ、Ⅱ、Ⅲ

B【解析】通常死锁的处理采用三种策略：死锁预防、死锁避免、死锁检测和解除。死锁预防，采用破坏产生死锁的四个必要条件中的一个或几个,以防止发生死锁。其中为了破坏“破坏循环等待条件”，一般采用顺序资源分配法。该方法给系统的资源进行编号，并规定每个进程必须按编号递增的顺序请求资源，也就是限制了用户申请资源的顺序，故I的前半句属于死锁预防的方法，I描述错误。

12、若系统S1采用死锁避免方法，S2采用死锁检测方法。下列叙述中，正确的是_____。

Ⅰ．S1 会限制用户申请资源的顺序，而 S2 不会

Ⅱ．S1 需要进程运行所需资源总量信息，而 S2 不需要

Ⅲ．S1 不会给可能导致死锁的进程分配资源，而 S2 会

A．仅Ⅰ、Ⅱ　　B．仅Ⅱ、Ⅲ

C．仅Ⅰ、Ⅲ　　D．Ⅰ、Ⅱ、Ⅲ

B【解析】死锁避免算法，是避免系统处于不安全状态，典型的算法是银行家算法，该算法包含MAX、Allocation、Need和Avaliable四个数据结构。

核心思想是当进程申请资源时，系统会进行此次资源预分配。并检查预分配后系统是否处于安全状态,若分配后系统处于安全状态，则进行分配；若不安全则将本次的预分配作废。

12、若系统S1采用死锁避免方法，S2采用死锁检测方法。下列叙述中，正确的是_____。

Ⅰ．S1 会限制用户申请资源的顺序，而 S2 不会

Ⅱ．S1 需要进程运行所需资源总量信息，而 S2 不需要

Ⅲ．S1 不会给可能导致死锁的进程分配资源，而 S2 会

A．仅Ⅰ、Ⅱ　　B．仅Ⅱ、Ⅲ

C．仅Ⅰ、Ⅲ　　D．Ⅰ、Ⅱ、Ⅲ

B【解析】死锁检测的基于鸵鸟策略，即假设系统不会出现死锁，也就不会限制进程申请和分配资源的顺序，即在系统为进程分配资源时不采取任何措施,但提供死锁的检测（死锁定理）和解除的手段，因此I的后半句正确，Ⅱ、Ⅲ正确。

13、系统中有3个不同的临界资源R1、R2和R3，被4个进程p1、p2、p3及p4共享。各进程对资源的需求为：p1申请R1和R2，p2申请R2和R3，p3申请R1和R3，p4申请R2。若系统出现死锁，则处于死锁状态的进程数至少是_____。

A. 1　　B. 2　　C. 3　　D. 4

C【解析】本题请注意，系统出现死锁是本题的前提条件，也就是系统中已经出现了死锁。显然4个进程是处于死锁状态最多的进程数，但是题目考察的是最少。p4只需要使用R2，为了让处于死锁的进程最少，首先将R2分配给p4，允许其执行完成，R2不会处于死锁状态。又p1申请R1和R2，p2申请R2和R3，p3申请R1和R3，为了让系统出现死锁，可以让p1拿到R1,p2拿到R2，p3拿到R3（其实有很多种分配情形），此时p1、p2、p3均处于死锁状态。因此是3个进程处于死锁。

13、系统中有3个不同的临界资源R1、R2和R3，被4个进程p1、p2、p3及p4共享。各进程对资源的需求为：p1申请R1和R2，p2申请R2和R3，p3申请R1和R3，p4申请R2。若系统出现死锁，则处于死锁状态的进程数至少是_____。

A. 1　　B. 2　　C. 3　　D. 4

C 【解析】那么假如只有两个进程处于死锁，是不是最少的情况呢，答案是不能。因此3个资源，2个进程且每个进程最多需要2个资源，不管怎么分配，总有一个进程可以拿到两个资源。拿到两个资源的进程可以执行完成，并释放资源，允许其它进程继续执行。那么就不会出现死锁了。

14、假设4个作业到达系统的时刻和运行时间如下表所示。

作业	到达时刻 t	运行时间
J1	0	3
J2	1	3
J3	1	2
J4	3	1

系统在t=2时开始作业调度。若分别采用先来先服务和短作业优先调度算法，则选中的作业分别是_____。

A．J2、J3　　B．J1、J4　　C．J2、J4　　D．J1、J3

D【解析】在t=2时刻时开始作业调度，到达的作业是J1，J2，J3，根据FCFS算法，作业到达时刻越早，优先级越高，因此会选择 J1。根据短作业优先调度算法，作业运行时间越短，优先级越高，因此会选择J3。所以 D 正确。请注意，t=2时刻时，J4尚未到达。

15、在高响应比进程调度算法中,其主要影响因素是()。

A．等待时间　　B. 剩余运行时间

C. 已运行时间　　D. 静态优先级

A【解析】在高响应比进程调度算法中,响应比=(等待时间+要求服务时间)/要求服务时间。当进程要求服务时间(执行时间)相同时,等待时间越长,响应比越高。

16、系统中资源R的数量为12,进程P1,P2,P3对资源R的最大需求分别为10、4、9。若当前已分配给P1、P2,P3的资源R的数量分别为5、2、2,则系统()。

A.处于不安全状态

B.处于安全状态,且安全序列为P1→P2→P3

C.处于安全状态,且安全序列为P2→P3→P1

D.处于安全状态,且安全序列为P2→P1→P3

D【解析】利用银行家算法对当前系统资源分配进行评估:当前系统可用资源数为12-5-2-2=3,其中P2当前需求的资源数为4-2=2<3,故可为P2进行资源分配;当P2完成以后,系统可用资源数为3+2=5,而此时P1需求的资源数为10-5=5,故可为P1进行资源分配;当P1完成以后,系统可用资源数为5+5=10,而此时P3需求的资源数为9-2=7<10,故可为P3进行资源分配。综上所述,系统处于安全状态,且安全序列为P2→P1→P3。

17、高响应比优先的进程调度算法综合考虑了进程的等待时间和计算时间,响应比的定义是()。

A.进程周转时间与等待时间之比

B.进程周转时间与计算时间之比

C.进程等待时间与计算时间之比

D.进程计算时间与等待时间之比

B【解析】在高响应比优先的进程调度算法中,

响应比=(等待时间+计算时间)/计算时间=周转时间/计算时间。

18、若系统中有5台打印机,有多个进程均需要使用2台,规定每个进程一次仅允许申请1台,则至多允许()个进程参与竞争,而不会发生死锁。

A.5　　B.2　　C.3　　D.4

D【解析】系统中有5台打印机,每个进程需要使用2台。当有4个进程各申请了1台打印机后,当前系统中还有1台打印机可以分配,满足任一进程的需求,不会发生死锁。

19、系统中有4个进程都要使用某类资源。若每个进程最多需要3个该类资源,为保证系统不发生死锁,系统应提供该类资源至少是()。

A.3个　　B.4个　　C.9个　　D.12个

C【解析】系统中有4个进程,每个进程最多需要3个资源,先给每个进程分配2个资源,共需要8个资源,此时需要系统中还有1个空闲资源,分配给任一进程,才不会发生死锁,故至少需要9个资源。

20、某进程调度程序采用基于优先数(priority)的调度策略，即选择优先数最小的进程运行，进程创建时由用户指定一个nice作为静态优先数。为了动态调整优先数，引入运行时间cpuTime和等待时间 waitTime，初值均为0。进程处于执行态时，cpuTime定时加1，且 waitTime置0；进程处于就绪态时，cpuTime置0，waitTime定时加1。请回答下列问题。

(1)若调度程序只将nice的值作为进程的优先数，即 priority=nice，则可能会出现饥饿现象，为什么？

【解析】(1)由于采用了静态优先数nice，根据题意，nice小的优先级大。当就绪队列中总有优先数nice较小的进程就绪时，优先数较大的进程将一直没有机会运行，最终会出现饥饿现象。这也是静态优先级调度算法的缺点。

20、某进程调度程序采用基于优先数(priority)的调度策略，即选择优先数最小的进程运行，进程创建时由用户指定一个nice作为静态优先数。为了动态调整优先数，引入运行时间cpuTime和等待时间 waitTime，初值均为0。进程处于执行态时，cpuTime定时加1，且 waitTime置0；进程处于就绪态时，cpuTime置0，waitTime定时加1。请回答下列问题。

(2)使用nice、cpuTime和waitTime设计一种动态优先数计算方法，以避免产生饥饿现象，并说明 waitTime的作用。

【解析】(2)模仿高响应比优先调度算法的思想，可以设计动态优先数priority的计算公式为:prority=nice - waitTime / （cpuTime+1），这里的分母加1是为了处理cpuTime为0的情况，且prority小的优先级高，

初始时，进程的waitTime是0，则按照静态优先数nice的设置来安排进程的执行顺序。对于静态优先数nice的进程，随着waitTime的增加，可以使其动态优先数priority减小，并最终可以获得执行的机会，从而避免出现饥饿现象。

谢谢大家