1. **简答题：**

**1．什么是并发？什么是并行？用日常生活中的例子举例说明。**

答：并发是指在一段时间内多个进程同时运行，宏观上同时，微观上依次执行。

并行是指若干进程在同一时刻同时运行。

日常生活例子：以食堂打饭为例。

并行：食堂排队打饭，如果把每个窗口看做CPU，每个同学看作进程，那么在某一刻在各个窗口打饭的同学可以认为是并行的。

并发：同样以打饭为例，对其中一个窗口而言，在该窗口排队的每个同学只能依次获得打饭机会，因此可以认为该队列的每个同学是并发的（在食堂开始服务的时间段内都能打到饭）。

**2.在操作系统中为什么要引入进程的概念？它会产生什么样的影响？**

答：在多道程序环境下，程序并发执行时失去了封闭性、具有间断性及运行结果不可再现等特征。特别是运行结果不可再现这一特征，决定了程序不能参与并发执行。因此，为了能够使程序并发执行，必须引入“进程”概念对并发执行的程序加以描述和控制。

引入进程这一概念后，通过PCB来管理进程运行过程中需要的描述信息、控制信息、管理信息等，可以使原来不能并发执行的程序可以并发执行。

**3.试说明PCB的作用具体表现在那些方面？为什么说PCB是进程存在的唯一标志？**

答：为了便于系统描述和管理程序的运行，操作系统为每个参与执行的程序（含数据）配置了一个专门的数据结构叫做PCB（又称为进程控制块）。

PCB的作用是使一个在多道程序环境下不能独立运行的程序成为一个能独立运行的基本单位，一个能与其他进程并发执行的进程。主要表现在：1）作为独立运行的标志；2）能够间断运行；3）提供进程管理需要的信息；4）提供进程调度的信息；5）实现进程间的同步与通信。

在进程存在的整个生命周期，操作系统都是通过PCB控制进程的运行，通过PCB感知进程的存在与否，所以说PCB是进程存在的唯一标志。

**4.说明进程的三个基本状态及其相互转换的典型原因？**

答：进程运行过程中，由于竞争资源，因此会在不同状态中转换。三种基本状态分别是：就绪状态、运行状态和阻塞状态。

三种基本状态转换的典型原因是：

就绪状态的进程，当进程调度获得CPU时，可从就绪状态转换为执行状态。

执行状态的进程，当有I/O请求时，会从执行状态转换为阻塞状态；在分时系统中，执行状态的进程时间片用完，会从执行状态转换为就绪状态。

阻塞状态的进程，当I/O请求完成，会从阻塞状态转换为就绪状态。

**5.试从动态性、并发性和独立性上比较进程和程序？**

答：从动态性看：进程是程序的执行，由创建而产生，由调度而执行，由撤销而消亡；而程序是个静态实体，不管运行与否均存在。

从并发性看：程序不能并发执行，而进程可以并发执行。

从独立性看：进程是独立运行，独立获取资源及独立接受调度的基本单位；未建立PCB的程序是不能作为独立运行的单位参与运行。

**6.什么临界资源和临界区？临界区管理的基本准则是什么？**

答：把在一段时间内只允许一个进程访问的资源称为临界资源。

在每个进程中访问临界资源的那段代码。

管理临界区的准则：

(1) 空闲让进。如果若干进程要求进入空闲的临界区，一次仅允许一个进程进入。

(2) 忙则等待。任何时候，处于临界区内的进程不可多于一个。

(3) 有限等待。进入临界区的进程要在有限时间内退出。

(4) 让权等待。如果进程不能进入自己的临界区，则应让出CPU，避免进程出现“忙等”现象

**7.为什么要在操作系统中引入线程？试从调度性、并发性、 拥有资源和系统开销4个方面对进程和线程进行比较。**

答：在OS中引入进程的目的是为了使多个程序能并发执行，以提高资源利用率和系统吞吐量，那么，在操作系统中再引入线程，则是为了减少程序在并发执行时所付出的时空开销，使OS具有更好的并发性。

进程与线程的比较：

从调度性来看：线程是调度和分派的基本单位，而进程是资源拥有的基本单位，线程共享进程的全部资源。一个进程内的线程切换到另一个进程内的线程时要引起进程的切换。

从并发性来看：在引入线程的OS中，不仅进程之间可以并发执行，而且在一个进程中的多个线程之间亦可并发执行，因而OS具有更好的并发性，从而能更好地提高系统的资利用率及系统的吞吐量。

从拥有资源来看：进程是拥有资源的基本单位，线程不拥有资源，但继承进程的资源。

从系统开销来看：创建或撒消进程，系统都要为之分配或回收资源，OS的开销将大于创建线程或撒消线程的开消。类似地在进程切换时，涉及到整个当前进程CPU环境的保存以及新被调度运行的进程的CPU环境的设置。而线程切换只须保存和设置少量寄存器的内容，并不涉及存储器管理方面的操作。因此，进程切换的开销也远大于线程切换的开销。

1. **综合题：**

**8. 画出下列语句的前趋图：并指出哪些语句可以并发执行？**

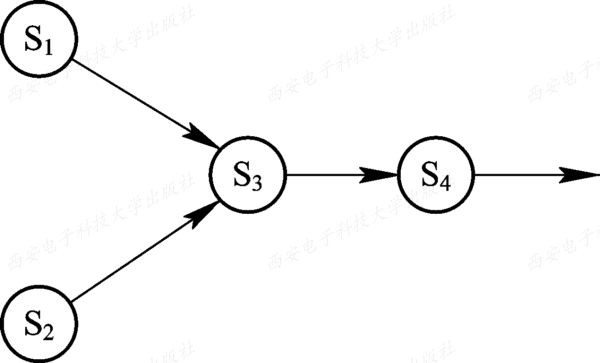
S1:a=x+y

S2:b=z+1

S3:c=a-b

S4:w=c+1

答：可画出如图所示的前趋关系。可以看出：S3必须在S1和S2被执行后方能执行；S4必须在S3之后执行；其中，S1和S2可以并发执行，因为它们彼此互不依赖。



**9.假设有输入、加工和输出3个并发进程共享一个缓冲区B，输入进程负责从输入设备读入一条记录，每读一条记录后把它存放在缓冲区B中，加工进程在缓冲区B中加工输入进程存入的记录。输出进程负责把加工后的记录打印输出。缓冲区B中每次只能存放一条记录，当记录被加工输出后，缓冲区B中才可存放下—条新记录。请用P、V操作来描述它们并发执行时能正确工作的程序。**

答：设fullB,emptyB分别为缓冲区B装满数据和空的信号量，Work为数据加工信号量，0为未加工，1为加工后的数据。

var fullB,emptyB,Work,:semaphore:=0,1,0;

Begin

Parbegin

Procedure输入: begin

repeat

wait ( emptyB );

读一条记录放入缓冲区中

signal ( fullB );

until false;

end

Procedure加工: begin

repeat

wait ( fullB );

在缓冲区B中进行加工；

signal (work );

until false;

end

Procedure输出: begin

repeat

wait (work );

打印输出；

signal ( emptyB);

until false;

end

parend

end

**10. 有一窄桥每次只能过一辆车，每次为了保证正常通行，只要桥上没有车，就允许一端的车过桥，待其全部过完后才允许另一端的车过桥。请用信号量和PV操作写出过窄桥的同步算法。**

答：设mutex为桥的互斥信号量，rmutex,lmutex分别为向右，向左过桥的互斥信号量，设rightcount,leftcount分别为向右向左行驶的汽车数量

Var mutex,Rmutex,Lmutex , samphore :=1,1,1;

Rightcount,leftcount:integer:=0;

Parbegin

向右行驶:

begin

P( Rmutex);

If rightunt=0 then p(mutex);

rightcount:=rightcount+1;

V(Rmutex);

开始过桥；

P(Rmutex);

rightcount:=rightcount－1;

If rightcount=0 then v(mutex);

V(Rmutex);

End

向左行驶:

begin

P(Lmutex);

If leftcount=0 then p(mutex);

leftcount:=leftcount+1;

V(Lmutex);

开始过桥；

P(Lmutex);

leftcount:=leftcount－1;

If leftcount=0 then v(mutex);

V(Lmutex);

End

Parend

**11．某招待所有100个床位，住宿者住入要先登记(在****登记表上填写姓名及床位号)，离去时要撤销登记(在登记表上删去姓名和床位号)。请给出住宿登记及撤销登记过程的算法描述。**

答：设mutex=1为登记信号量

Empty=100为床位

登记：begin

Repeat

Wait(empty)

Wait(mutex)

登记表上填写姓名及床位号

Signal(mutex)

Until false

end

撤销登记：

Begin

Repeat

Wait(mutex)

撤销登记；

Signal(mutex)

Signal(empty)

Until false