第一章　操作系统概论

考点一：操作系统的作用

1、操作系统是计算机用户与硬件的接口。

2、操作系统是计算机系统资源的管理者。

考点二：单道批处理系统

1、单道批处理操作系统内存中只有一道作业，可以自动成批处理作业。

2、单道批处理系统的特点：自动性、顺序性、单道性。

3、单道批处理系统的缺点：由于作业独占 CPU 和内存，当作业进行 I/O 时，CPU 只能等待 I/O 完成而无事可做，使得 CPU 资源不能得到充分利用。

考点三：多道批处理系统

1、多道批处理系统的特点：多道性、无序性、调度性、复杂性。

2、多道程序系统的优点是能够提高 CPU、内存和 I/O 设备的利用率和系统的吞吐量。

3、多道批处理系统的缺点：系统平均周转时间长，缺乏交互能力。

考点四：分时系统

1、分时系统允许多个用户通过终端同时使用计算机。

2、分时系统的特点：多路性、独立性、及时性和交互性。

3、分时系统的优点：向用户提供了人机交互的方便性，使多个用户可以通过不同的终端共享主机。

考点五：实时系统

1、实时系统主要用于实时控制和实时信息处理领域，要求系统在指定时间内开始响应和在指定时间内完成。

2、实时系统的特点：多路性、独立性、及时性、交互性和可靠性。

3、实时系统应用于各种工业现场的自动控制、智能机器人、海底探测和航空航天等领域。

考点六：操作系统的特征

1、现代操作系统都支持多任务，具有并发性、共享性、虚拟性和异步性。

2、并发性：指两个或多个时间在同一时间间隔内发生。与并行有区别，并行是指多个事件同时发生。

3、共享性：指系统中的资源可供内存中多个并发执行的进程共同使用。资源共享有两种方式：互斥共享和同时共享。

4、虚拟性：指通过某种技术把一个物理实体变成若干逻辑上的对应物。

5、异步性：进程以不可预知的速度向前推进。内存中的每个程序何时执行、何时暂停、以怎样的速度向前推进，以及每道程序总共需要多少时间才能完成等，都是不可预知的。

考点七：操作系统的功能

操作系统的主要功能包括：内存管理、进程管理、设备管理和文件管理。

考点八：操作系统的体系结构

1、单体结构模型：软件工程出现以前的早期操作系统及目前一些小型操作系统采用的体系结构。

2、层次结构模型：基本思想是将操作系统分解为多个小的、容易理解的层，系统的功能被隔离在不同层中，每一层提供对系统功能的部分抽象，然后采用调用的顺序，形成一连串彼此；连续的对系统功能的“抽象串”，最终形成对整个系统的完整抽象。

3、微内核结构：是操作系统发展的一个里程碑，它产生了一种完全不同的操作系统体系结构，提供了操作系统发展的新途径。

考点九：指令的执行

1、指令的执行过程分是反复取指令和执行指令的过程。PC 始终存在下一条待取指令指令的地址。指令执行的结果就是使寄存器或内存单元的值发生变化，指令执行的过程也就是存储体内部不断变化的过程。

2、取指令和执行指令是由硬件完成的，不同硬件的体系结构支持不同的指令集合，为某一种硬件平台开发的操作系统不能直接在另一种体系结构的硬件上运行。

3、任何高级程序设计语言的程序被编译成指令集合，其中的每一条指令属于计算机体系结构的指令集。

第二章　进程管理

考点一：进程的并发执行

1、程序顺序执行的特点：

（1）顺序性：处理机的操作，严格按照程序所规定的顺序执行，即只有前一操作结束后，才能执行后继操作。

（2）封闭性：程序一旦开始运行，其结果不受外界因素的影响。

（3）可再现性：只要程序执行时的环境和初识条件相同，当程序多次重复执行时，其执行结果相同。

2、程序并发执行的特点：

（1）间断性：程序在并发执行时，由于他们共享资源，而资源数量又往往少于并发执行的程序数量，系统不能保证每个程序不受限地占用资源。

（2）失去封闭性：程序在并发执行时，由于他们共享资源或者合作完成同一项任务，系统的状态不再是只有正在执行的某一个程序可以“看见”和改变。

（3）不可再现性：程序在并发执行时，由于失去了封闭性，也将导致其失去执行结果的可再见性。

考点二：进程的描述

1、进程是允许并发执行的程序在某个数据集合上的运行过程，由程序块/段（也称用户正文段）、用户数据段以及进程控制块（PCB）组成。

2、进程控制块是进程实体的一部分，是操作系统中最重要中最重要的数据结构。进程控制块中记录了操作系统所需要的、用于描述进程情况及控制进程情况及控制进程运行所需的全部信息。

3、进程控制块中保留的处理机状态信息通常包括通用寄存器、指令计数器、程序状态字和用户栈指针。

考点三：进程的状态

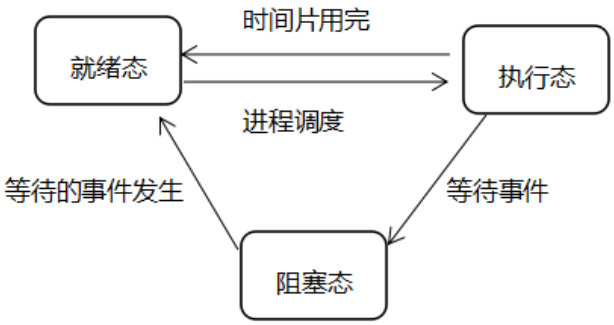
1、进程的 3 中基本状态：

（1）就绪态：进程一旦获得 CPU 就可以投入运行的状态。

（2）执行态：进程获得 CPU 正在运行的状态。

（3）阻塞态：进程由于等待资源或某个事件的发生暂停执行的状态，系统不会为处于阻塞态的进程分配 CPU。

2、进程状态的转换：



考点四：进程的创建与撤销

1、创建进程的条件：

① 用户登录；② 作业调度；③ 提供服务；④ 应用请求。

2、创建进程的过程：

① 申请空白 PCB；② 为新进程分配资源；③ 初始化进程控制块；④ 将新进程插入就绪队列。

3、进程撤销的条件：

① 当进程正常执行完毕，调用终止进程的系统调用，请求操作系统删除该进程

② 一个进程调用适当的系统调用，终止另外一个进程。

4、进程撤销的过程：

① 从进程 PCB 中读进程状态

② 若进程正在执行，则终止进程的执行

③ 若进程有子孙进程，在大多数情况下需要终止子孙进程

④ 释放资源

⑤ 将终止进程的 PCB 移出

考点五：进程的阻塞与唤醒

1、进程阻塞和唤醒条件：

① 请求系统服务；② 启动某种操作；③ 新数据尚未到达；④ 无新工作可做。

2、进程阻塞的过程：

① 将进程的状态改为阻塞态；② 将进程插入相应的阻塞队列；③ 转进程调度程序，从就绪进程中选择进程为其分配 CPU。

3、进程唤醒的过程：

① 将进程从阻塞态中移出；② 将进程状态有阻塞状态改为就绪态；③ 将进程插入就绪队列。

考点六：操作系统内核

1、操作系统内核是计算机硬件的第一次扩充，内核执行操作系统与硬件关系密切，执行频率高的模块，常驻内存。

2、操作系统内核的功能：

（1）支持功能：时钟管理、中断管理和原语操作。

（2）资源管理功能：进程管理、存储器管理和设备管理。

考点七：中断

1、由于某些事件的出现，中止现行进程的运行，而由操作系统去处理出现的事件，待适当的时候让被中止的进程继续运行，这个过程称为中断。引起中断的事件称为中断源。

2、中断分为同步中断（内部中断）和异步中断（外部中断），外部中断又可分为外部可屏蔽中断和外部不可屏蔽中断。

考点八：系统调用

1、系统调用是系统程序与用户接口之间的接口，在类 UNIX 系统中，系统调用多使用 C 语言提供的库函数作为接口。

2、用户空间是指用户进程所处的地址空间，一个用户进程不能访问其他进程的用户空间只有系统程序才能访问其他用户空间。当 CPU 执行用户空间的代码时，称该进程在用户态执行。

3、系统空间是指含有一切系统核心代码的地址空间，当 CPU 执行系统核心代码时，称进程处于系统态执行。

考点九：时钟管理

1、计算机的很多活动都是由定时测量来控制的，两种定时测量：

（1）保存当前的系统时间和日期

（2）维持定时器

2、操作系统依靠时钟硬件和时钟驱动程序完成上述两种定时测量功能。

3、操作系统内核可以利用时钟机制防止一个进程垄断 CPU 或者其他资源。

考点十：线程

1、线程是进程的一个实体，是被系统独立调度和分派的基本单位。

2、线程只拥有在运行中必须的资源，包括程序计数器、一组寄存器和栈，但它可与同属一个进程的其他线程共享进程所拥有的全部资源。一个线程可与创建和撤销另一个进程。同一个进程中的多个线程可与并发执行。线程在运行中存在间断性，也有就绪、执行、阻塞三种形态。

3、进程和线程密切相关，可以从以下几个角度来说明线程和进程的关系：

（1）资源和调度。线程是程序执行的基本单位，进程是拥有资源的基本单位。

（2）地址空间资源。不同进程的地址空间是互相独立的，而同一进程中的各线程共享同一地址空间。

（3）通信关系。进程之间的通信必须使用操作系统提供的进程间通信机制，而同一进程中的各线程间可以通过直接读写全局变量来通信，甚至无需操作系统的参与。

（4）并发性。多个进程和多个进程之间均可并发执行，而且同一进程中多个线程之间可以并发执行。

（5）系统开销。相比进程而言，线程在创建、撤销及上下文切换时系统开销很小，且速度更快。

考点十一：线程的控制

1、线程控制是线程实现中最基本的功能，它包括创建新线程、终止线程、线程调度和线程切换，以及线程由于等待某个事件的发生而被阻塞与该事件发生后线程被唤醒。

2、线程创建：

① 用户线程的创建是通过调用线程库中的实用程序完成的

② 内核线程的创建是由内核完成的

考点十二：进程的同步

1、同步机制应遵循的准则：空闲让进、忙则等待、有限等待和让权等待。

2、信号量机制：

（1）整型信号量机制：用整型变量值来标记资源的使用情况。若整型量>0，说明有可用资源;若整型量<=0，说明资源忙，进程必须等待。对于一次只允许一个进程访问的临界资源，可定义一个用户互斥的整型信号量，并将其初始化为 1， 整型信号量的值只能通过两个特定的原子操作 wait 和 signal 来改变。

```
Var s integer;
wait(s){        //申请资源
while s<=0 do no-op ;
    s=s-1; //占用资源
}
signal(s){    //释放资源
s=s+1
}
```

整型信号量的互斥：初始变量为 1

整型信号量的协调：初始变量为 0

（2）记录型信号量机制：

① 记录型信号量的数据类型：

```
Type semaphore = record
            Value : integer;       //资源数量
            L : list of process;   //阻塞队列
    end
```

② 记录型信号量的 wait（s）和 signal（s）操作：

```
procedure wait(s)
    var s : semaphore;
    begin
        s.value = s.value-1; //申请资源
        if s.value <0 then block(s.L);
end
```

```
procedure signal(s)
    var s:semaphore;
    begin
        s.value=s.value +1;//释放一个资源
        if s.value <=0 then wakeup(s.L);
end
```

考点十三：生产者与消费者问题

1、对于生产者和消费者同步模型，需要实现生产者“满则等待”和消费者“空则等待”2 个同步点。

2、生产者进程同步代码描述：

```
Producer:
    brgin
repeat
    ......
    produce an item in nextp;
    wait(empty);            //申请空缓冲区
    wait(mutex);            //申请公共缓冲池的互斥访问权
    buffer(in)=nextp;       //将消息放入 in 指针指向的缓冲区
    in=(in+1)mod n;         //in 指针指向下一个空缓冲区
    signal(mutex);          //释放对公共缓冲池的互斥访问权
    signal(full);           //释放消息资源
    until false;
    end
```

3、消费者进程同步代码的描述：

```
Consumer:
begin
repeat
......
    wait(full);             //申请消息
    wait(mutex);            //申请公共缓冲池的互斥访问权
    nextc=buffer(out);      //从 out 指针指向的缓冲区中取消息
    out=(out+1)mod n;       //out 指针指向下一个装有消息的缓冲区
    signal(mutex);          //释放对公共缓冲池的互斥访问权
    signal(empty);          //释放空缓冲区
    consume item in nextc;
until false;
end
```