第一章

1.1.1. 对于一台 PC 而言，下列各项中(A)对系统必不可少。

A. OS　　B. Office 软件

C. C 语言编辑器　　D. 杀毒软件

1.1.2. 从用户的角度看，OS 是（ A ）。

A. 用户与计算机硬件系统之间的接口

B. 控制和管理计算机系统资源的软件

C. 合理组织计算机工作流程的软件

D. 一个大型的工具软件

1.1.3. 配置了 OS 的计算机是一台比原来的物理计算机功能更加强大的计算机，这样的计算机只是一台逻辑上的计算机，称为（ A ）计算机。

A. 虚拟　　B. 物理

C. 并行　　D. 共享

1.2.1. 与单道程序系统相比，多道程序系统的优点是（ D ）。

I. CPU 利用率高　　II. 系统开销小

III. 系统吞吐量大　　IV. I/O 设备利用率高

A. 仅 I、III　　B. 仅 I、IV

C. 仅 II、III　　D. 仅 I、III、IV

与单道程序系统相比，系统要付出额外的开销来组织作业和切换作业，故 II 错误。

1.2.2. 下列关于批处理系统的叙述中，正确的是（ A ）。

I. 批处理系统允许多个用户与计算机直接交互

II. 批处理系统分为单道批处理系统和多道批处理系统

III. 中断技术使得多道批处理系统和 I/O 设备可与 CPU 并行工作

A. 仅 II、III　　B. 仅 I

C. 仅 I、III　　D. 仅 I、III

1.2.3. （ C ）系统允许一台主机上同时连接多台终端，多个用户可以通过各自的终端同时交互地使用计算机。

A. 网络　　B. 分布式

C. 分时　　D. 实时

1.2.4. 下列（ D ）等的实现最好采用实时系统平台。

I. 航空订票系统　　II. 办公自动化系统

III. 机床控制系统　　IV. AutoCAD

V. 工资管理系统　　VI. 股票交易系统

A. I、II、III　　B. I、III、IV

C. I、IV、V　　D. I、III、VI

1.2.5. OS 的基本类型主要有（ B ）。

A. 批处理系统、分时系统和多任务系统

B. 批处理系统、分时系统和实时系统

C. 单用户系统、多用户系统和批处理系统

D. 实时系统、分时系统和多用户系统

1.3.1. 并发性是指若干事件在（ C ）发生。

A. 同一时刻　　B. 不同时刻

C. 同一时间间隔内　　D. 不同时间间隔内

1.3.2. 单处理机系统中，可并行的是（ D ）。

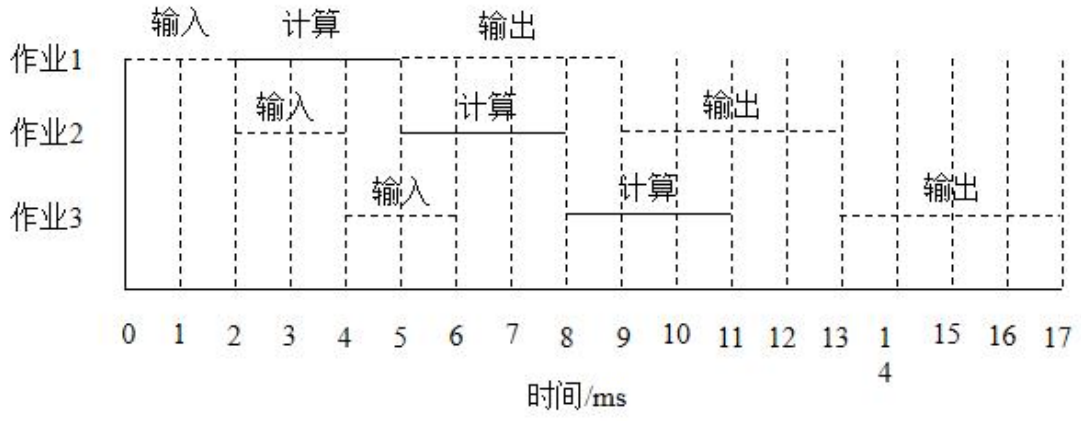
I. 进程与进程　　II. 处理机与设备

III. 处理机与通道　　IV. 设备与设备

A. I、II、III　　B. I、II、IV

C. I、III、IV　　D. II、III、IV

1.3.3. 某单 CPU 系统中有输入设备和输出设备各 1 台，现有 3 个并发执行的作业，每个作业的输入、计算和输出时间分别为 2ms、3ms 和 4ms，且都按输入、计算和输出的顺序执行，则执行完这 3 个作业需要的时间最少是多少？



1.4.1. 下列选项中，OS 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

1.5.1. 下列选项中，会导致用户进程从用户态切换到内核态的操作是（ B ）。

I. 整数除以 0　　II. sin( )函数调用

III. read 系统调用

A. 仅 I、II　　B. 仅 I、III

C. 仅 II、III　　D. I、II、III

整数除以 0 会发生异常，需要在内核态执行；系统调用也需要在内核态执行；sin( )函数调用在用户态执行。

1.5.2. 下列关于系统调用的叙述中，正确的是（ C ）。

I. 在执行系统调用服务程序的过程中，CPU 处于内核态

II. OS 通过提供系统调用来避免用户程序直接访问外设

III. 不同的 OS 为应用程序提供了统一的系统调用接口

IV. 系统调用是 OS 内核为应用程序提供服务的接口

A. 仅 I、IV　　B. 仅 II、III

C. 仅 I、II、IV　　D. 仅 I

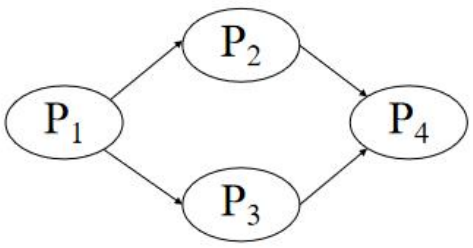
不同的 OS 为应用程序提供的系统调用接口不一定相同。

第二章

2.1.1. 画出下面 4 条语句对应的前驱图。

$P_1: a = x + 2y;$　　$P_2: b = a + 6;$

$P_3: c = 4a - 9;$　　$P_4: d = 2b + 5c;$



2.1.2. 程序运行时独占系统资源，只有程序本身才能改变系统资源状态，这是指（ D ）。

A. 程序顺序执行时的再现性

B. 并发程序失去再现性

C. 并发程序失去封闭性

D. 程序顺序执行时的封闭性

2.2.1. 进程和程序的本质区别在于（ D ）。

A. 前者分时使用 CPU，后者独占 CPU

B. 前者存储在内存，后者存储在外存

C. 前者具有异步性，后者具有可再现性

D. 前者可以并发执行，后者不能并发执行

2.2.2. 一个进程由（ ）、（ ）和（ ）构成。

程序段、相关的数据段、PCB

2.2.3. 一般情况下，分时系统中处于（ B ）的进程最多。

A. 执行状态　　B. 就绪状态

C. 阻塞状态　　D. 终止状态

分时系统将 CPU 的工作时间划分成若干个时间片，轮流为就绪队列中的进程服务。当某个进程获得时间片而进入执行状态时，就绪队列中的其他进程任然处于就绪状态，等待时间片的到来，因此，一般情况下，分时系统中处于就绪状态的进程最多。

2.2.4. 在单处理机系统中，关于进程的叙述，正确的是（ D ）。

A. 一个进程一旦创建完成就进入执行状态

B. 只能有一个进程处于就绪状态

C. 一个进程可以同时处于就绪状态和阻塞状态

D. 最多只有一个进程能处于运行状态

2.2.5. 已经获得除（ C ）以外的运行所需所有资源的进程处于就绪状态。

A. 存储器　　B. 打印机

C. CPU　　D. 磁盘空间

2.2.6. 当一个进程（ B ）时，称其处于阻塞状态。

A. 等待进入内存

B. 等待协作进程的一个消息

C. 等待一个时间片

D. 等待 CPU 调度

2.2.7. 一个进程的读磁盘操作完成后，OS 针对该进程必做的是（ A ）。

A. 修改进程状态为就绪状态

B. 降低进程优先级

C. 为进程分配用户内存空间

D. 延长进程的时间片

2.2.8. 下列选项中，可能导致当前进程 P 阻塞的事件是（ C ）。

I. 进程 P 申请临界资源

II. 进程 P 从磁盘读数据

III. 系统将 CPU 分配给高优先级进程

A. 仅 I　　B. 仅 II

C. 仅 I、II　　D. I、II、III

III 会让进程进入就绪状态。

2.2.9. 下列进程状态的转换中，（ C ）是不可能发生的。

A. 就绪->运行

B. 运行->就绪

C. 就绪->阻塞

D. 阻塞->就绪

2.2.10. 在实时系统中，当内存资源无法满足执行紧迫任务的需求时，OS 可能将正在运行的进程的状态变为（ B ）状态。

A. 活动就绪　　B. 静止就绪

C. 活动阻塞　　D. 静止阻塞

2.2.11. 进程的状态和优先级信息存放在（ B ）中。

A. JCB　　B. PCB

C. 快表　　D. 页表

2.2.12. OS 通过（　）来感知进程的存在。

PCB

2.3.1. OS 中有一组特殊的程序，它们不能被系统中断。在 OS 中它们称为（ B ）。

A. 初始化程序　　B. 原语

C. 子程序　　D. 控制模块

2.4.1. 两端程序分别如下，其中，R1、R2 是寄存器，counter 是共

享变量，且其初值为 1。

程序 A：R1 = counter; R1 = R1 + 1; counter = R1;

程序 B：R2 = counter; R2 = R2 - 1; counter = R2;

在这两端程序对应地进程并发执行后，counter 所有可能的值是什么？

按这个顺序执行

R1 = counter; //R1 = 1

R1 = R1 + 1; //R1 = 2

counter = R1; //counter = 2

R2 = counter; //R2 = 2

R2 = R2 - 1; //R2 = 1

counter = R2; //counter = 1

结果为 1；

按这个顺序执行

R1 = counter; //R1 = 1

R1 = R1 + 1; //R1 = 2

R2 = counter; //R2 = 1

R2 = R2 - 1; //R2 = 0

counter = R1; //counter = 2

counter = R2; //counter = 0

结果为 0；

按这个顺序执行

```
R1 = counter; //R1 = 1
R1 = R1 + 1; //R1 = 2
R2 = counter; //R2 = 1
R2 = R2 - 1; //R2 = 0
counter = R2; //counter = 0
counter = R1; //counter = 2
```

结果为 2；

2.4.2.关于临界区，正确的说法是（ A ）。

A. 访问不同临界资源的两个进程不要求必须互斥地进入临界区

B. 临界区是包含临界资源的一段数据区

C. 临界区是一种用于进程同步的机制

D. 临界区是访问临界资源的一个进程或线程

2.4.3. 临界区是指并发进程中访问共享变量的（ D ）段。

A. 信息管理　　B. 信息存储

C. 数据　　D. 程序

2.4.4. 下列准则中实现临界区互斥机制所必须遵循的是（ C ）。

I. 两个进程不能同时进入临界区

II. 允许进程访问空闲的临界资源

III. 进程等待进入临界区的时间是有限的

IV. 不能进入临界区且处于执行状态的进程立即放弃 CPU

A. I、IV　　B. II、III

C. I、II、III、IV　　D. I、III、IV

2.4.5. 在下列同步机制中，可以实现让权等待的是（ B ）。

A. Swap 指令　　B. 信号量机制

C. Test-and-Set 指令

2.4.6. 设与某资源相关联的信号量初值为 3，当前值为 1，则该资源的可用个数和等待资源的进程数分别为（ B ）。

A. 0、1　　B. 1、0

C. 1、2　　D. 2、0

2.4.7. 若记录型信号量 S 的初值为 17，当前值为-17，则表示有（ A ）个等待进程。

A. 17　　B. 34

C. 35　　D. 18

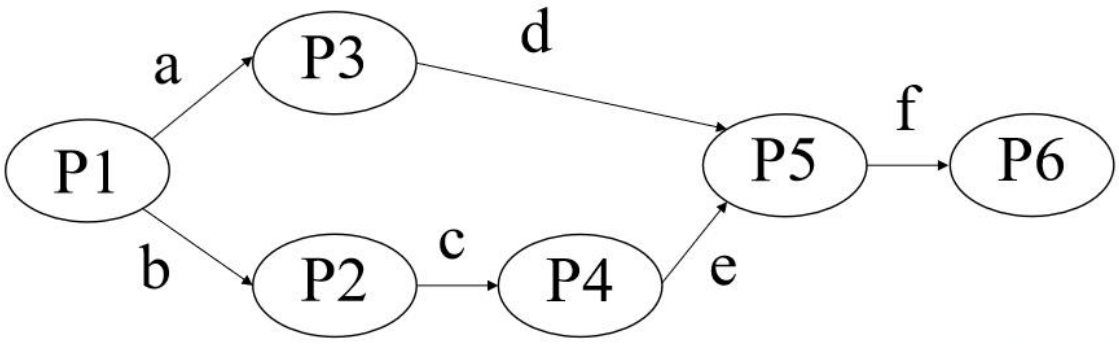
2.4.8. 如果 3 个进程共享一个互斥段，每次最多允许 2 个进程进入该互斥段，则信号量的变化范围是（ A ）。

A. 2、1、0、-1　　B. 3、2、1、0

C. 2、1、0、-1、-2　　D. 1、0、-1、-2

初值为 2

2.4.9. 有一组相互合作的进程 P1、P2、P3、P4、P5、P6，它们的执行过程须满足下图所示的同步关系，请使用信号量机制对该组进程进行同步。




```
P1(){S1; signal(a); signal(b); }
P2(){wait(a); S2; signal(c); }
P3(){wait(a); S3; signal(d); }
P4(){wait(c); S4; signal(e); }
P5(){wait(d); wait(e); S5; signal(f); }
P6(){wait(f); S6; }
main(){
  semaphone a, b, c, d, e, f;
  a = b = c = 0;
  d = e = f = 0;
  P1(); P2(); P3(); P4(); P5(); P6();
}
```

2.4.10. 下列关于管程的叙述中，错误的是（ A ）。

A. 管程只能用于实现进程的互斥

B. 管程是一种进程同步机制

C. 任何时候只能有一个进程在管程中执行

D. 管程中定义的变量只能被管程内的过程访问

管程不仅可以用于实现进程的互斥，还可以用于实现进程的同步。

2.4.11. 若 x 是管程内的条件变量，则当执行 x.wait 时，所做的工作是（ C ）。

A. 实现对变量 x 的互斥访问

B. 唤醒一个在 x 上阻塞的进程

C. 根据 x 的值判断该进程是否进入阻塞状态

D. 阻塞该进程，并将其插入 x 的阻塞队列中

2.5.1. 系统中有多个生产者进程和消费者进程，共享一个可以存放 1000 个产品信息的缓冲区（初始为空），当缓冲区未满时，生产者进程可以在其中放入一个其生产的产品信息，否则等待；当缓冲区不空时，消费者进程可以在其中取走 1 个产品信息，否则等待。要求 1 个消费者进程从缓冲区连续取走 10 个产品信息后，其他消费者进程才可以取产品信息。请用 P、V 操作或 wait( )、signal( )操作实现进程间的同步。

```
item buffer[1000];
item nextp;
item nextc[10];
int in = 0, out = 0;
semaphore empty = 1000, full = 0;
//资源信号量
semaphore mutex1 = 1;
//用于生产者和消费者之间的互斥
```

```
semaphore mutex2 = 1;
//用于消费者之间的互斥
void producer(){
    do{
        producer an item nextp;
        wait(empty);
        wait(mutex1);
        buffer[in] = nextp;
        in = (in + 1) % 1000;
        signal(mutex1);
        signal(full);
    while(TRUE);
}
void consumer(){
    do{
        wait(mutex2);
        for (i = 0; i < 10; i++){
            wait(full);
   wait(mutex1);
   nextc[i] = buffer[out];
   out = (out + 1) % n;
   signal(mutex1);
```

```
    signal(empty);
        }
        signal(mutex2);
        consumer the items in nextc;
    }
while(TRUE);
}
```

2.5.2. 有 n 位哲学家围坐在一张圆桌边，每位哲学家交替地就餐和思考。在圆桌中心有 m 个碗，每两位哲学家间有 1 根筷子。每位哲学家必须取到 1 个碗和两侧的筷子之后才能进餐，进餐完毕后将碗和筷子放回原位，继续思考。请用 P、V 操作或 wait( )、signal( )操作实现进程间的同步，要求防止出现死锁现象。

```
semaphore bowl = m; //用于协调哲学家对碗的使用
semaphore chopstick[n]; ////用于协调哲学家对筷子的使用
for (int i = 0; i < n; i++)
    chopstick[i] = 1;
void pi(){ //第 i 位哲学家的程序
    do{
        wait(bowl);
        if (i % 2 == 1){
            wait(chopstick[i]);
            wait(chopstick[(i + 1) % n]);
```

```
        }
        else{
            wait(chopstick[(i + 1) % n]);
            wait(chopstick[i]);
        }
        //eat
        signal(chopstick[i]);
        signal(chopstick[(i + 1) % n]);
        signal(bowl);
        //think
    }while(TRUE);
}
```

2.5.3. 两个进程 P1、P2 并发执行，并用 M1、M2 分别实现对两个互斥共享资源 R1 和 R2 的互斥访问。这两个进程以什么次序执行会导致死锁？在不影响程序功能的情况下，请修改算法以防止死锁。

```
void P1(){
    wait(M1);
    访问 R1;
    wait(M2);
    访问 R1 和 R2;
    signal(M1);
    signal(M2);
```

```
}
void P2(){
    wait(M2);
    访问 R2;
    wait(M1);
    访问 R1;
    signal(M1);
    signal(M2);
}
```

P1 执行了 wait(M1),申请成功,然后访问 R1,然后 P2 执行 wait(M2) ，申请成功，这时会出现死锁。

可以这样修改 P2：

```
void P2(){
    wait(M1);
    wait(M2);
    访问 R2;
    访问 R1;
    signal(M1);
    signal(M2);
}
```

2.6.1. 下面关于管道通信的叙述中，正确的是（ C ）。

A. 一个管道可实现双向数据同时传输

B. 管道的容量仅受磁盘容量大小的限制

C. 进程对管道进行读操作和写操作都可能被阻塞

D. 在一次通信过程中，一个管道只能有一个读进程或一个写进程对其进行操作

管道存在于内存中，因此容量大小不受磁盘容量大小的限制

2.6.2. 管道通信是以（ B ）为单位进行写入和读出的。

A. 消息　　B. 自然字符流

C. 文件　　D. 报文

2.6.3. 用信箱实现进程间互通信息的通信机制要有两个通信原语，它们是（ C ）。

A. 发送原语和执行原语

B. 就绪原语和执行原语

C. 发送原语和接收原语

D. 就绪原语和接收原语

2.7.1. 一个进程可以包含多个线程，各线程（ A ）。

A. 共享进程的虚拟地址空间

B. 地址空间完全独立

C. 是资源分配的单位

D. 共享堆栈

2.7.2. 在引入线程的 OS 中，把（ D ）作为调度和分派的基本单位，而把（ ）作为拥有资源的基本单位。

A. 进程、线程

B. 程序、线程

C. 程序、进程

D. 线程、进程

2.7.3. 下面的叙述中，正确的是（ C ）。

A. 在一个进程中创建一个新线程比创建一个新进程所需的工作多

B. 同一进程中的线程间通信和不同进程中的线程间通信差不多

C. 同一进程中的线程间切换由于有许多上下文相同而可以被简化

D. 同一进程中的线程间通信须调用内核

2.8.1. 下列关于线程的描述中，错误的是（ B ）。

A. 内核支持线程的调度由 OS 完成

B. OS 为每个用户级线程建立一个 TCB

C. 用户级线程间的切换比内核支持线程间的切换效率高

D. 用户级线程可以在不支持内核支持线程的 OS 上实现

在有些操作系统中，用户级线程对内核来说不可见，所以 OS 不会为它分配 TCB。用户级线程的切换可以在用户空间完成，而内核支持线程的切换需要 OS 进行调度，因此用户级线程的切换效率更高。

2.8.2. 下列关于进程和线程的叙述中，正确的是（ A ）。

A. 不管系统是否支持线程，进程都是资源分配的基本单位

B. 线程都是资源分配的基本单位，进程是调度的基本单位

C. 内核支持线程和用户级线程的切换都需要内核的支持

D. 同一进程中的各个线程拥有各自不同的地址空间

第三章

3.2.1. 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	运行时间
J1	0	3
J2	1	3
J3	1	2
J4	3	1

系统在 t=2 时开始调度作业。若分别采用 FCFS 和 SJF 调度算法，则选中的作业分别是（ D ）。

A. J2、J3　　B. J1、J4

C. J2、J4　　D. J1、J3

3.2.2. 假设某 OS 以单道批处理方式运行，现有 4 道作业，它们进入系统的时间的及运行时间如下表所示，试采用高响应比优先调度算法进行调度，请问这组作业的运行顺序、平均周转时间和平均带权周转时间分别是多少？

作业号	进入系统时间	运行时间（小时）
1	7:00	2
2	7:50	0.5
3	8:00	0.1
4	8:50	0.2

7:00-9:00 运行作业 1，周转时间 2 小时，带权周转时间 1；

这时，作业 2 的响应比为$\frac{\frac{7}{6}+0.5}{0.5}=\frac{20}{6}$

作业 3 的响应比为$\frac{1+0.1}{0.1}=11$

作业 4 的响应比为$\frac{\frac{1}{6}+0.2}{0.2}=\frac{11}{6}$

9:00-9:06 运行作业 3，周转时间 1.1 小时，带权周转时间 11；

这时，作业 2 的响应比为$\frac{\frac{19}{15}+0.5}{0.5}=\frac{53}{15}$

作业 4 的响应比为$\frac{\frac{4}{15}+0.2}{0.2}=\frac{35}{15}$

9:06-9:36 运行作业 2，周转时间 1.77 小时，带权周转时间 3.54；

9:36-9:48 运行作业 4，周转时间 0.97 小时，带权周转时间 4.85；

平均周转时间$\frac{2+1.1+1.77+0.97}{4}=1.46$ 小时，

带权平均周转$\frac{1+11+3.54+4.85}{4}=5.1$ 小时，

3.3.1. 下列有关基于时间片的进程调度的叙述中，错误的是（ B ）。

A. 时间片越短，进程切换的次数越多，系统开销越大

B. 当前进程的时间片用完后，该进程的状态变为阻塞状态

C. 时钟中断发生后，系统会修改当前进程在时间片内的剩余时间

D. 影响时间片大小的主要因素包括响应时间、系统开销和进程数量等

时钟中断是系统中特定的周期性时钟节拍，OS 通过它来确定时间间隔，实现时间调节。

3.3.2. 某系统采用抢占式 SJF 调度算法，下表给出了 5 个进程的到达时间和要求运行时间。

（1）请将表格填写完整；

（2）计算这 5 个进程的平均带权周转时间。

作业号	进入系统时间	运行时间（小时）
1	7:00	2
2	7:50	0.5
3	8:00	0.1
4	8:50	0.2

平均带权周转时间为 1.27。

进程	到达时间	运行时间	开始运行时间	完成时间	带权周转时间
P1	0:00	4小时	0:00	7:00	7/4
P2	1:00	1小时	1:00	2:00	1
P3	3:00	2小时	3:00	5:00	1
P4	6:00	5小时	7:00	14:00	8/5
P5	8:00	2小时	8:00	10:00	1

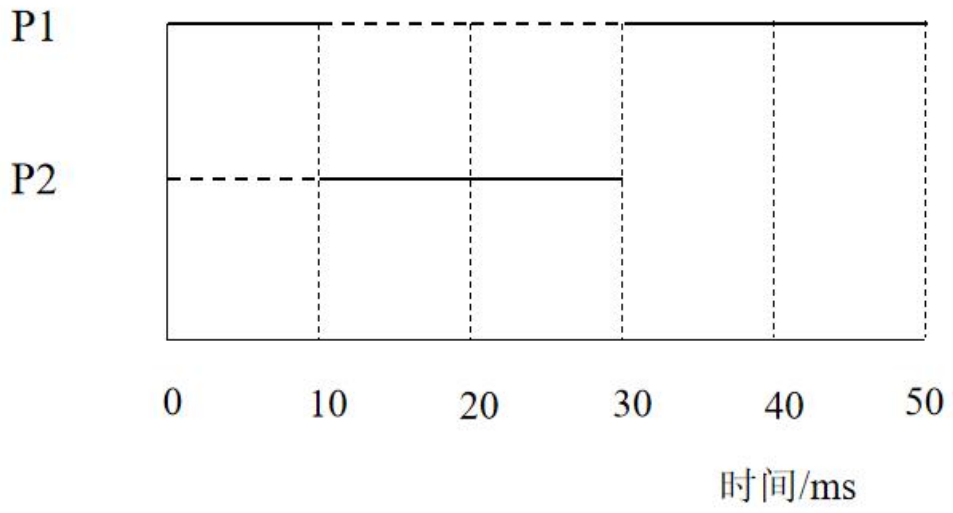
3.3.3. 下列调度算法中，不会产生饥饿现象的是（ A ）。

A. RR　　B. 静态优先级

C. 非抢占式 SJF　　D. 抢占式 SJF

3.3.4. 系统两级反馈队列调度算法进行进程调度。就绪队列 Q1 采用 RR 调度算法，时间片为 10ms；就绪队列 Q2 采用短进程优先调度算法。系统优先调度 Q1 队列中的进程，当 Q1 位空时系统才会调度 Q2 中的

进程；新创建的进程首先进入 Q1；Q1 的进程执行一个时间片后若未结束，则转入 Q2。若当前 Q1 和 Q2 为空，系统依次创建进程 P1、P2 后即开始调度进程，P1、P2 需要的 CPU 时间分别为 30ms 和 20ms,则进程 P1、P2 在系统中的平均等待时间为多少。



平均等待时间为(20+10)/2=15ms。

3.3.5. 5 个进程 P1、P2、P3、P4、P5 几乎同时到达，它们预期运行时间分别为 10、6、2、4、8 个时间单位。各进程的优先级为分别为 3、5、2、1、4（数值越大，优先级越高）。请按下列调度算法计算任务的平均周转时间。

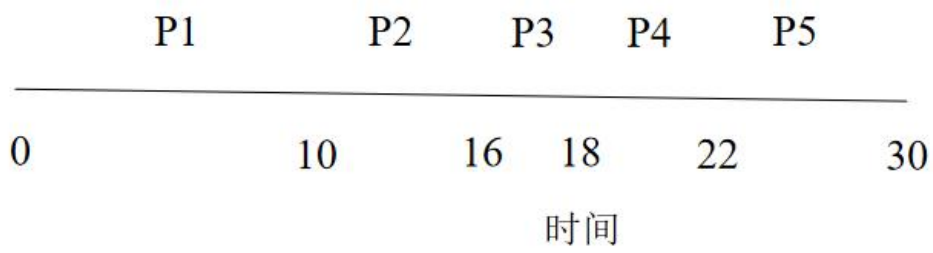
（1）FCFS（按 P1-P2-P3-P4-P5 顺序）调度算法。

（2）RR 调度算法，假定时间片大小为 2 个时间单位。

（3）优先级调度算法。

（1）FCFS 调度算法

平均周转时间为(10+16+18+22+30)/5=19.2。



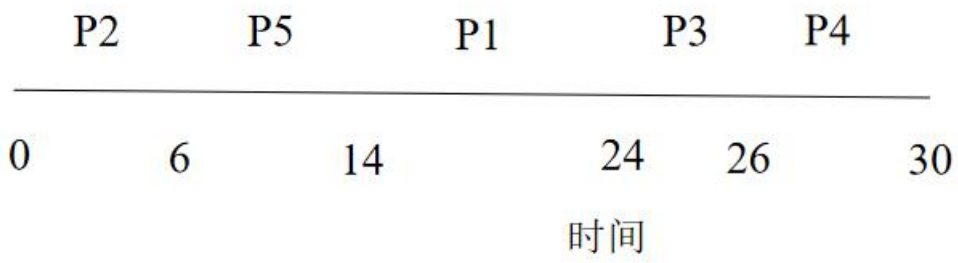
（2）RR 调度算法

平均周转时间为(30+22+6+16+28)/5=20.4。

P1	P2	P3	P4	P5	P1	P2	P4	P5	P1	P2	P5	P1	P5	P1
0	2	4	6	8	10	12	14	16	18	20	22	24	26	28	30

（3）优先级调度算法

平均周转时间为(6+14+24+26+30)/5=20。



3.7.1. 某系统中有 n 台互斥使用的同类设备，3 个并发进程分别需要 3、4、5 台该类设备，可确保系统不发生死锁的最小设备数为多少。

假设系统中有 9 台设备，3 个进程分别占有 2、3、4 台设备，这时出现死锁。若增加 1 台设备，则不会出现死锁。所以最小设备数为 10。

总结：设 m 个进程的需求量分别为 n1，...，nm，则不会出现死锁的最小资源数为

$$\left(n_1 - 1 + \cdots + n_m - 1\right) + 1$$

3.7.2. 某计算机系统 8 台打印机，由 K 个进程竞争使用它们，每个进程需要 3 台打印机。该系统可能会发生死锁的 K 的最小值是多少。

由前面的公式，不会出现死锁的最小资源数为 2K+1，所以 2K+1>8 即 K>3.5 ，所以 K 的最小值为 4.

3.7.3. 假设 5 个进程 P0、P1、P2、P3、P4 共享 3 类资源 R1、R2、R3，这些资源总数分别为 18、6、22。T 时刻的资源分配表如下所示。请检查系统是否处于安全状态，是的话计算一个安全序列。

这些资源总数分别为18、6、22。

	已分配资源			最大需求资源		
进程	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

这些资源总数分别为18、6、22。

	已分配资源			需求资源			可用资源			安全序列
进程	R1	R2	R3	R1	R2	R3	2	3	3	
P0	3	2	3	2	3	7	6	3	6	P1
P1	4	0	3	1	3	3	10	3	11	P2
P2	4	0	5	0	0	6	13	5	14	P0
P3	2	0	4	2	2	1	15	5	18	P3
P4	3	1	4	1	1	0	18	6	22	P4

3.7.4. 假设 5 个进程 P0、P1、P2、P3、P4 共享 4 类资源 A、B、C、D，假设出现如下的进程资源分配情况。

（1）该状态是否安全？为什么？

（2）如果进程 P0 提出资源请求（0,0,0,1），则系统能否将资源分配给它？为什么？

进程	已分配资源	还需资源	当前可用资源
P0	1, 1, 1, 0	0, 3, 3, 1	0, 3, 2, 2
P1	0, 2, 3, 1	0, 3, 4, 2	
P2	0, 2, 1, 2	1, 0, 3, 4	
P3	0, 3, 1, 0	0, 3, 2, 0	
P4	1, 0, 2, 1	0, 4, 2, 3	

（1）存在一个安全序列 P3,P0,P1,P4,P2，因此该状态是安全的。

进程	已分配资源	还需资源	当前可用资源	安全序列
			0, 3, 2, 2	
P0	1, 1, 1, 0	0, 3, 3, 1	0, 6, 3, 2	P3
P1	0, 2, 3, 1	0, 3, 4, 2	1, 7, 4, 2	P0
P2	0, 2, 1, 2	1, 0, 3, 4	1, 9, 7, 3	P1
P3	0, 3, 1, 0	0, 3, 2, 0	2, 9, 9, 4	P4
P4	1, 0, 2, 1	0, 4, 2, 3	2, 11, 10, 6	P2

（2)由于Request0(0,0,0,1)＜Need0(7,4,3) ，Request0(0,0,0,1)＜ Available(0,3,2,2)，所以尝试将资源分配给P0，并修改相关数据。

进程	已分配资源	还需资源	当前可用资源	安全序列
			0, 3, 2, 1	
P0	1, 1, 1, 1	0, 3, 3, 0	0, 6, 3, 1	P3
P1	0, 2, 3, 1	0, 3, 4, 2	1, 7, 4, 2	P0
P2	0, 2, 1, 2	1, 0, 3, 4	1, 9, 7, 3	P1
P3	0, 3, 1, 0	0, 3, 2, 0	2, 9, 9, 4	P4
P4	1, 0, 2, 1	0, 4, 2, 3	2, 11, 10, 6	P2

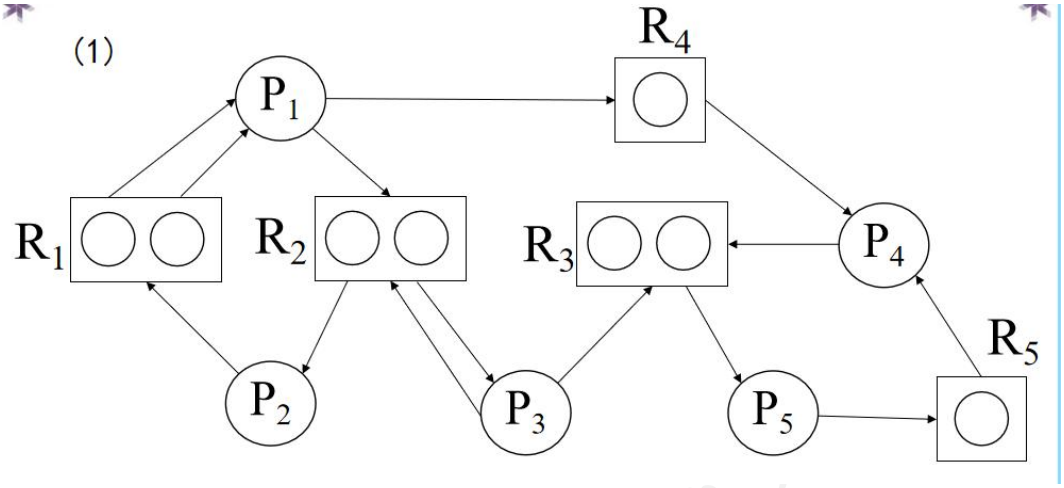
存在一个安全序列P3,P0,P1,P4,P2, 因此可实施资源分配。

3.8.1. 假设系统有5类独占资源：R1、R2、R3、R4、R5。各类资源分别有2、2、2、1、1个。系统有5个进程：P1、P2、P3、P4、P5。其中P1已占有2个R1,且申请1个R2和1个R4;P2已占有1个R2,且申请1个R1； P3已占有1个R2，且申请1个R2和1个R3；P4已占有1个R4和1个R5，且申请1个R3； P5已占有1个R3，且申

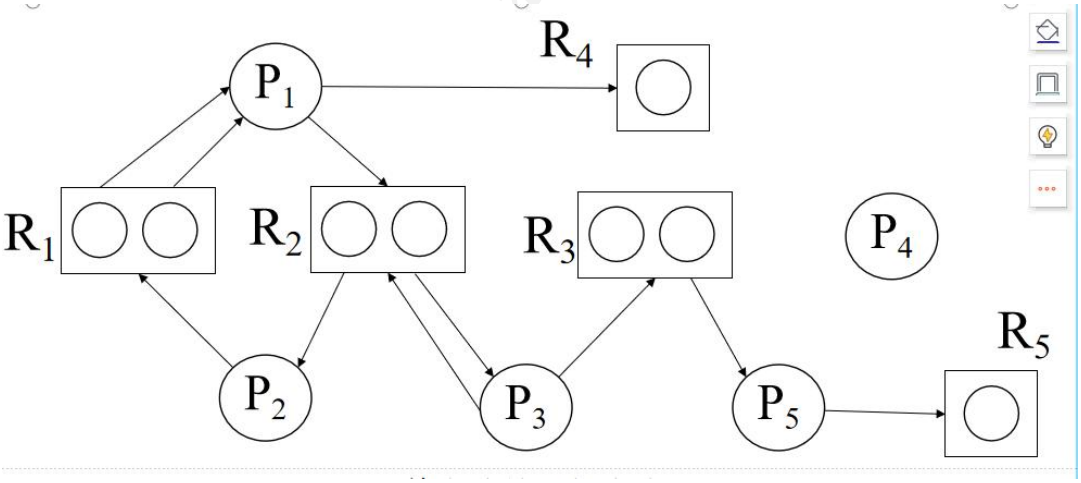
请 1 个 R5。

（1）画出该时刻的资源分配图；

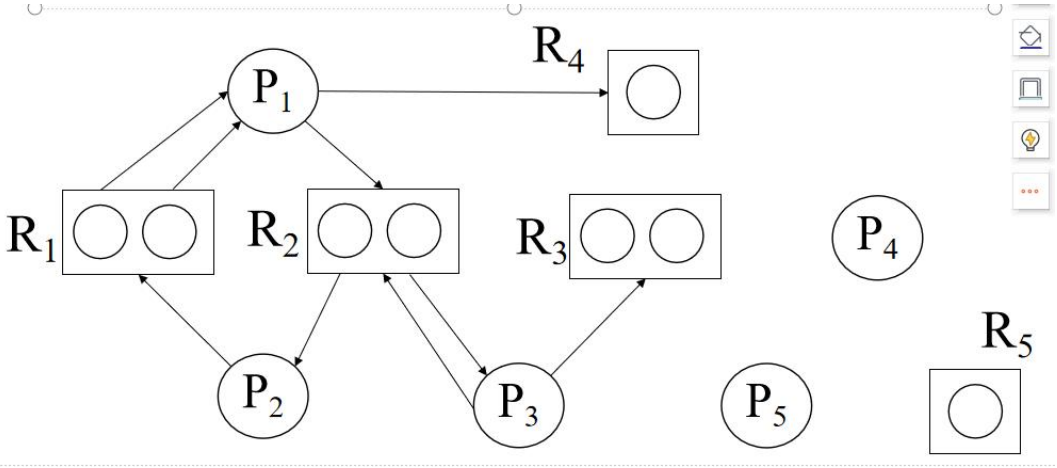
（2）判断该时刻系统是否处于死锁状态。



(2)Available = (0,0,1,0,0)，可以满足 P4 的要求， P4 执行完毕后，系统状态如下：



这时，Available = (0,0,1,1,1)，可以满足 P5 的要求， P5 执行完毕后，系统状态如下：



这时，Available = (0,0,2,1,1)，不能可以满足任何进程的要求，所以该资源分配图是不可完全简化的，所以系统处于死锁状态

4.2.1. 在多道程序环境中，用户程序的相对地址与装入内存后的实际物理地址不同，把相对地址转换为物理地址，这是 OS 的（ C ）功能。

A. 进程调度

B. 设备管理

C. 地址重定位

D. 资源管理

4.2.2. 关于程序链接，下列说法正确的是（ A ）。

A. 根据目标模块的大小和链接次序对相对地址进行修改

B. 根据装入位置把目标模块中的相对地址转换为绝对地址

C. 把每个目标模块中的相对地址转换为外部调用符号

D. 采用静态链接方式更有利于目标模块共享

4.2.3. 在程序运行前，先将一个程序的所有模块以及所需的库函数

链接成一个完整的装配模块，这种链接方式称为（ A ）。

A. 静态链接

B. 装入时动态链接

C. 可重定位链接

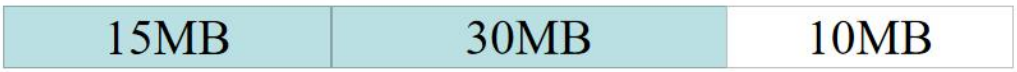
D. 运行时动态链接

4.3.1. 某基于动态分区存储管理的计算机，其内存容量为 55MB（初始为空闲），采用最佳适应算法，分配和释放的顺序为：分配 15MB，分配 30MB，释放 15MB，分配 8MB，分配 6MB。此时，内存中的最大空闲分区的大小是（ B ）MB。

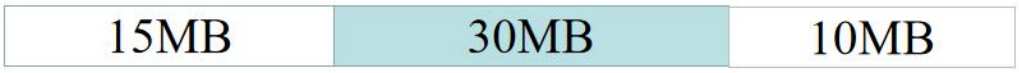
A. 7　　B. 9

C. 10　　D. 15

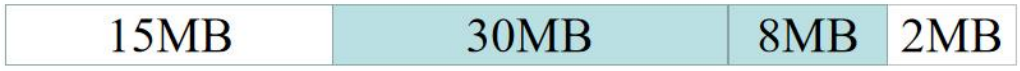
（1）分配15MB，分配30MB后：



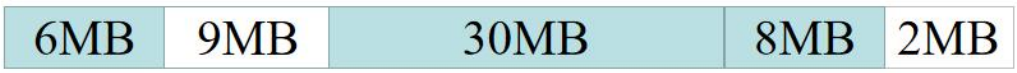
（2）释放15MB后：



（3）分配8MB后：



（4）分配6MB后：



4.3.2. 采用动态分区算法回收内存时，如果回收分区仅与空闲分区链插入点的前一个分区相邻接，那么需要在空闲分区表中（ B ）。

A. 增加一个新表项

B. 修改前一个分区表项的大小

C. 修改前一个分区表项的起始地址

D. 修改前一个分区表项的大小和起始地址

4.3.3.某系统采用动态分区分配方式管理内存，内存空间为 640KB，低端 40KB 存放 OS。对下列作业请求序列，画图表示使用首次适应算法进行内存分配和回收后内存的最终映像。

作业 1 申请 200KB，作业 2 申请 70KB；

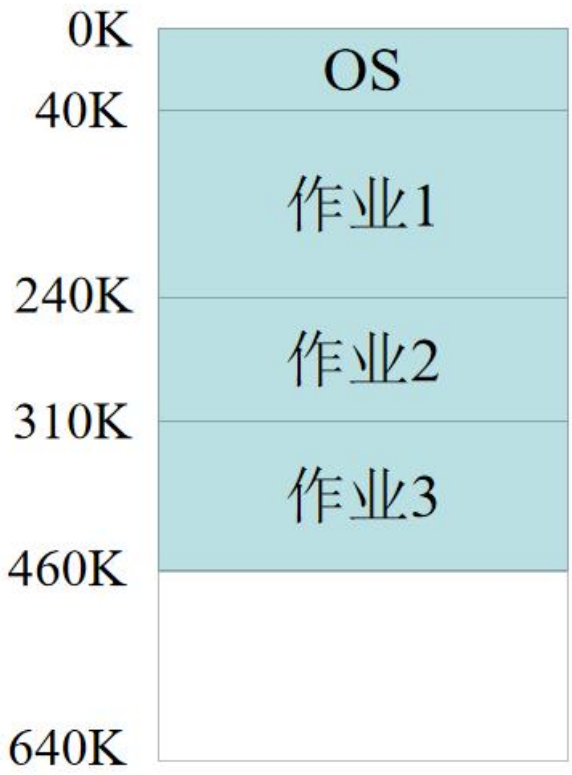
作业 3 申请 150KB，作业 2 释放 70KB；

作业 4 申请 80KB，作业 3 申请 150KB；

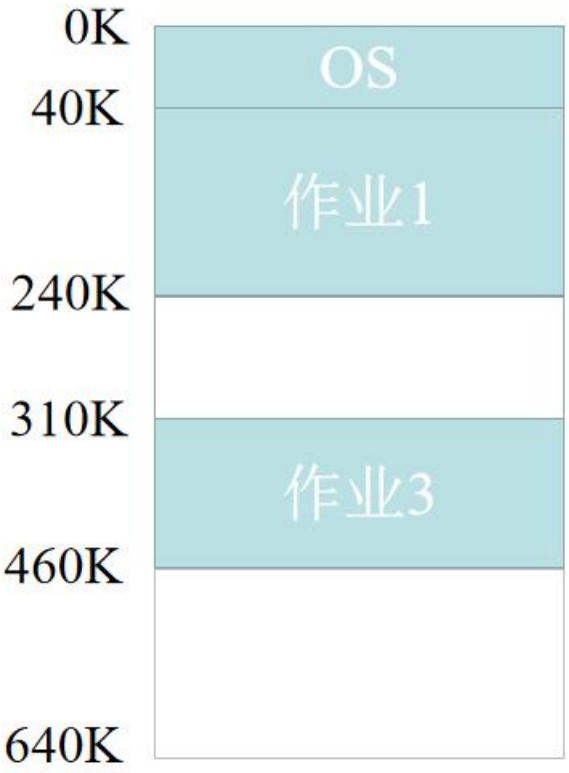
作业 5 申请 100KB，作业 6 申请 60KB；

作业 7 申请 50KB，作业 6 释放 60KB。

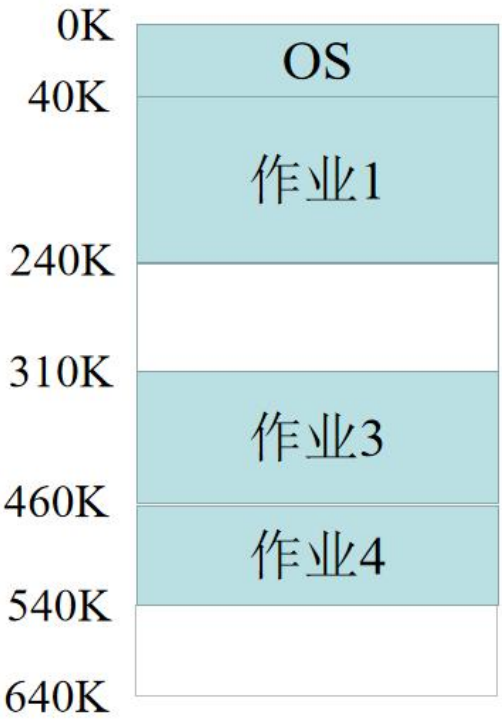
（1）作业 1 申请 200KB，作业 2 申请 70KB；作业 3 申请 150KB 后：



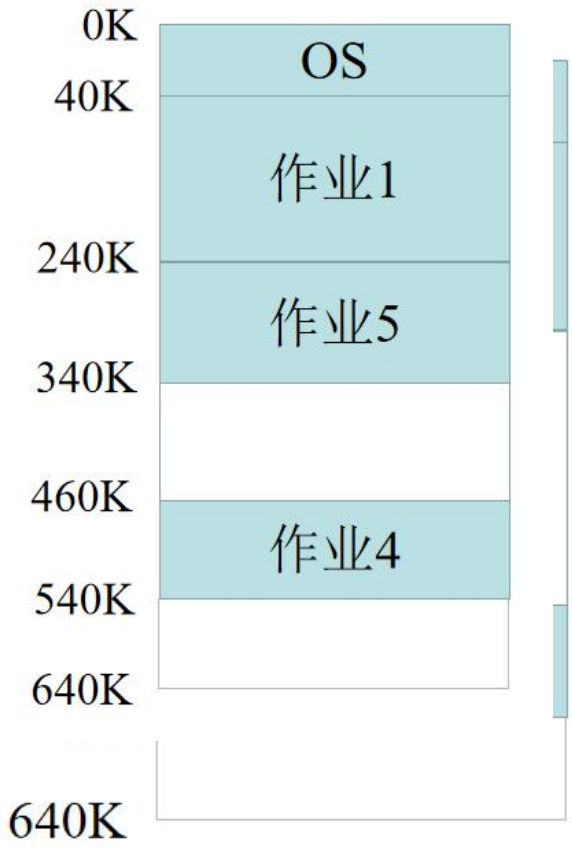
（2）作业 2 释放 70KB 后：



（3）作业 4 申请 80KB 后：

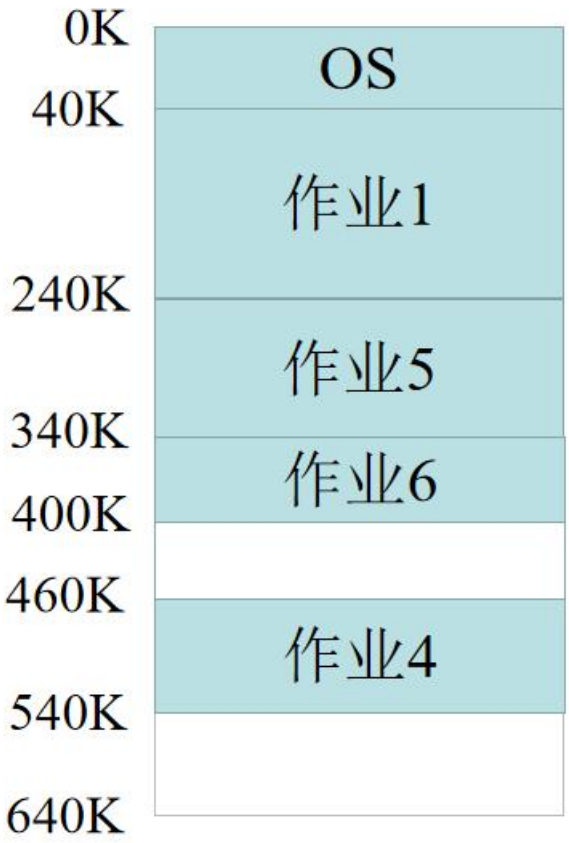


（4）作业 3 释放 150KB 后：

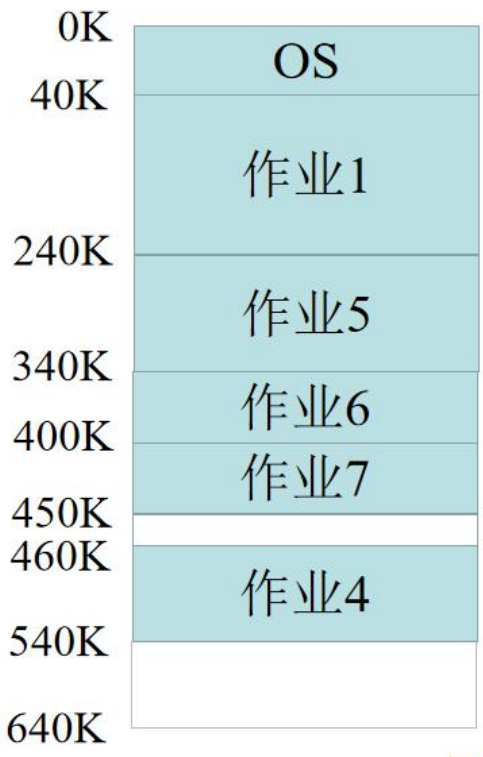


（5）作业 5 申请 100KB 后：

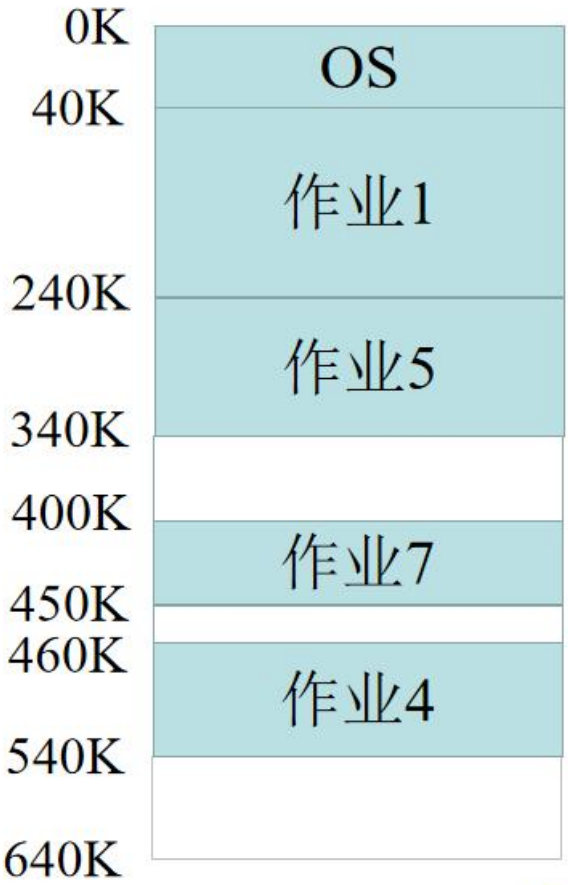
（6）作业 6 申请 60KB 后：



（7）作业 7 申请 50KB 后：



（8）作业 6 释放 60KB 后：



4.3.4. 可重定位内存的分区分配目的是（ A ）。

A. 解决碎片问题

B. 便于多作业共享内存

C. 便于用户干预

D. 便于回收空白分区

4.3.5. 不是基于顺序搜索的动态分区分配算法的是（ D ）。

A. 首次适应算法

B. 循环首次适应算法

C. 最坏适应算法

D. 快速适应算法

4.4.1. 对换技术的主要作用是（ B ）。

A. 将内存碎片合并为大的空闲空间

B. 提高内存利用率

C. 减少查找空闲分区的时间

D. 提高外部设备利用率

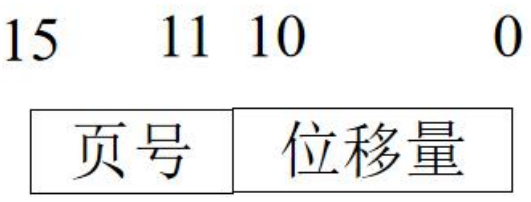
4.5.1. 某系统采用分页存储管理方式，拥有逻辑空间 32 页，每页 2KB；拥有物理空间 1MB。

（1）写出逻辑地址的格式；

（2）若不考虑访问权限等，则进程的页表最多有多少项？每项有多少位？

（3）如果物理空间减少一半，则页表结构应相应地做什么改变？

（1）该系统拥有逻辑空间 32 = 25 页，所以逻辑地址中页号用 5 位来描述。每页 2KB = 2^{11}B，所以页内地址用 11 位来描述，逻辑地址格式如下：



（2）每个进程最多有 32 个页面，所以，进程的页表项最多为 32 项。若不考虑访问权限等，则页表中只须给出页对应的物理块号，1MB / 2KB = 512 = 2^9，所以有 2^9 个块，故每个页表项有 9 位。

（3）512KB / 2KB = 256 = 2^8，所以有 2^8 个块，故每个页表项有 8 位，页表中的最多项数仍是 32.

4.5.2. 某分页系统的内存容量为 64KB，页面大小为 1KB，对一个 4 页大的作业，其 0、1、2、3 页分别被分配到内存的 2、4、6、7 块中。将十进制的逻辑地址 1023、2500、3500 转换为物理地址。

首先要计算出页号和页内地址。注意逻辑地址和物理地址都是以字节为单位。1KB = 1024B。

物理地址 = 物理块号 × 页面大小 + 页内地址。

（1）1023 / 1024 = 0，1023 % 1024 = 1023，因此页号为 0，页内地址为 1023,对应的物理块号为 2,所以物理地址为 2 × 1024 + 1023 = 3071.

（2）2500 / 1024 = 2， 2500 % 1024 = 452，因此页号为 2，页内地址为 452，对应的物理块号为 6，所以物理地址为 6 × 1024 + 542 = 6596.

（3）3500 / 1024 = 3， 3500 % 1024 = 428，因此页号为 3，页内地址为 428，对应的物理块号为 7，所以物理地址为 7 × 1024 + 428 = 7596.

4.5.3. 假设一个分页存储系统具有快表，多数活动页表项都可以存在于其中。若页表放在内存中，内存访问时间是 1ns，快表的命中率是 85%，快表的访问时间为 0.1ns，则有效存取时间为多少？

EAT = 0.1ns × 85% + (0.1 + 1) × (1 - 85%) + 1 = 1.25ns.

4.5.4. 某系统采用二级页表的分页存储管理方式，虚拟地址格式如下：

10 位	10 位	12 位
页目录号	页表索引	页内地址

（1）页面大小是多少？进程的虚拟地址空间大小最多为多少页？

（2）假定页目录项和页表项均占 4B，则进程的页目录和页表最多共占多少页？

（3）若在某指令周期内访问的虚拟地址为 0100 0000H（H 表示这是一个 16 进制数）和 0111 2048H，则进行地址转换时共访问多少个页表分页？

（1）页面大小是 2^{12}B = 4KB.进程的虚拟地址空间大小最多为 $2^{10} \times 2^{10} = 2^{20}$ 页。

（2）页目录所占页数为($2^{10} \times 4$) / 2^{12} = 1，页表所占页数为($2^{20} \times 4$) / 2^{12} = 1024，所以共占 1025 页。

（3）第一个地址的前 3 位 010 转换为二进制是 0000 0001 0000，前 10 位是 0000000100 = 4（十进制）；第二个地址的前 3 位 011 转换为二进制是 0000 0001 0001，前 10 位是 0000000100 = 4（十进制），所以访问的都是 4 号页表分页，因此，共访问一个页表分页。

4.5.5. 下列选项中，属于多级页表的优点的是（ D ）。

A. 加快地址转换速度

B. 减少缺页中断次数

C. 减少页表项所占字节数

D. 减少页表所占的连续内存空间

4.5.6. 某计算机采用二级分页存储管理方式，页面大小为 1KB，页

表项大小为 2B，逻辑地址空间大小为 2^{16} 页，则外层页表所包含的项数至少为（ B ）。

A. 64

B. 128

C. 256

D. 512

一个页表分页最多只能占一个页面，所以一个页表分页占一个页面时，外层页表的项数最少。1 页可以存放 2^{10} / 2 = 2^{9} 个页表项，所以需要 2^{16} / 2^{9} = 128 个页面来存放页表项，所以外层页表的表项数至少是 128，选 B。

4.5.7. 某系统采用二级页表的分页存储管理方式，虚拟地址格式如下：

10 位	10 位	12 位
页目录号	页表索引	页内地址

虚拟地址为 2050 1225H 对应的目录号和页表索引分别是什么？A

A. 081H、101H

B. 081H、401H

C. 201H、101H

D. 201H、401H

2050 1225H 转换为二进制为

0010 0000 0101 0000 0001 0010 0010 0101，

前 23-32 位是 0010 0000 01，前面添两个 0，变为 0000 1000 0001 =

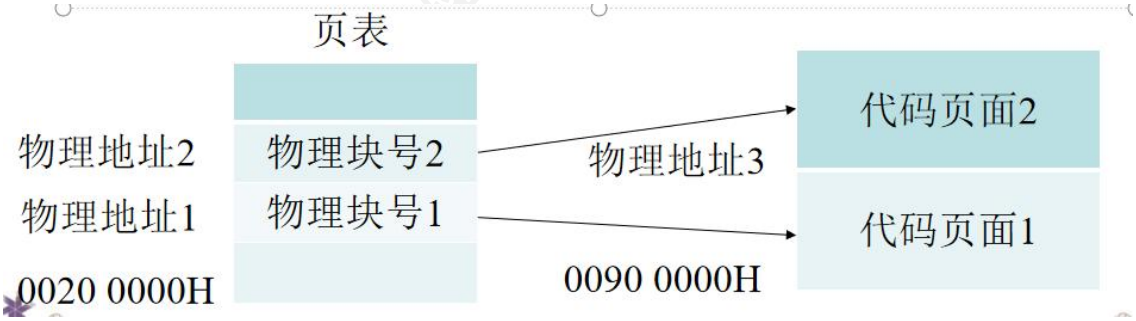
081H，13-22 位是 01 0000 0001，前面添两个 0，变为 0001 0000 0001 = 101H，选 A。

4.5.8. 某系统使用一级页表的分页存储管理方式，逻辑地址和物理地址都是 32 位，页表项大小为 4B，逻辑地址格式如下：

20 位	12 位
页号	页内地址

（1）页面大小是多少？页表最多占用多少字节？

（2）一个代码段的起始逻辑地址为 0000 8000H，其长度为 8KB，被装载到从起始物理地址为 0090 0000H 的连续内存空间中。页表从内存 0020 0000H 开始的物理地址处连续存放，如图所示（地址大小自下向上递增），请计算出该代码段对应的两个页表项的物理地址，以及这两个页表项中的物理块号和代码页面 2 的起始物理地址。



（1）页面大小是 2^{12}B = 4KB. 页表最多占用 2^{20} × 4B = 4MB。

（2）0000 8000H 的前 5 位 00008 转换为十进制是 8，所以页号为 8，物理地址 1 = 起始物理地址 + 页号 × 页表项大小 = 0020 0000H + 8 × 4 = 0020 0020H，物理地址 2 = 物理地址 1 + 页表项大小 = 0020 0020H + 4 = 0020 0024H。由于物理块大小和页面大小相同，所以 0090 000H 的前 5 位就是物理块号，因此，物理块号 1 是 00900H，物

理块号 2 是 00901H，物理地址 3 就是 00901H 号物理块的起始地址 00901000H。

4.6.1. 在分段管理中，（ A ）。

A. 以段为单位进行分配，每个段都是一个连续存储区

B. 段与段之间必定不连续

C. 段与段之间必定连续

D. 每个段都是等长的

4.6.2. 一个分段存储管理系统的地址长度为 32 位，其中段号占 8 位，则段长最大是（ C ）。

A. 2^8B

B. 2^{16}B

C. 2^{24}B

D. 2^{32}B

4.6.3. 某进程的段表内容如下所示：

当访问段号 2、段内地址为 400 的逻辑地址时，地址转换的结果是（ D ）。

A. 段缺失异常　　B. 得到内存地址 4400

C. 越权异常　　D. 越界异常

段号	段长	内存起始地址	状态
0	100	6000	在内存
1	200		不在内存

2	300	4000	在内存

4.6.4. 采用（ B ）不会产生内部碎片。

A. 分页存储管理

B. 分段存储管理

C. 随机存储管理

D. 段页式存储管理

4.6.5. 在内存管理中，内存利用率高且保护和共享容易的是（ D ）方式。

A. 分区存储管理

B. 分页存储管理

C. 分段存储管理

D. 段页式存储管理

4.6.6. 一个 OS 采用分段存储管理方式，支持的最大段长为 64KB，一个进程的段表如下所示（十进制）。请问：逻辑地址 47FD5H、003FFH 对应的物理地址是多少？

段号	段长	段起始地址
0	512	80K
1	20K	50K
2	12K	81K
3	3K	96K

4	32K	10K

（1）64KB = 2^{16}B。逻辑地址由 1 个 5 位十六进制数，所以逻辑地址最高位数代表段号，47FD5H 对应的的段号是 4，段内地址是 7FD5H，段长是 32K = 32768 = 2^{15}，下面把它转换为十六进制：

32768 % 16 = 0, 32768 / 16 = 2048;

2048 % 16 = 0, 2048 / 16 = 128;

128 % 16 = 0, 128 / 16 = 8;

所以 32768 对应的十六进制是 8000H，8000H > 7FD5H，没有越界。

起始地址是 10K = 2800H。

物理地址 = 起始地址 + 段内地址

所以物理地址为 2800H + 7FD5H = A7D5H。

（2） 003FFH 对应的段号是 0，段内地址是 03FFH，段长是 512，转换为十六进制是 200H，3FFH > 200H，因此会产生地址越界。