第二部分（操作系统大题集中营考点）
第二章处理器管理

1. 处理器低级调度算法详细介绍
（1）先来先服务算法　FCFS
调用后背队列中最先进入队列的一个或多个作业。属于非剥夺式调度
。
特点：利于长作业，不利于短作业。简单易实现。效率低。只顾等待
时间，不过执行时间。
（2）短作业 / 短进程优先调度算法　SJF
调用运行时间短的作业，属于非剥夺式调度。
特点：
降低平均等待时间，提过系统吞吐量。
对长作业不利。
（3）最高优先权调度算法
调度优先权高的作业，分为：
非抢占式：被调进程一直运行，直到结束或等待事件发生才主动放弃
CPU。
抢占式：运行中的进程将　CPU的使用权让给优先权高的

（4）最高响应比算法
系统响应时间作业等待时间　+作业要求时间
R= =

作业要求运行时间作业要求时间
属于非剥夺式调度。

（5）时间片轮转调度算法

进程在规定的时间内没有结束，系统将产生一个中断。
属于剥夺式算法

（6）最短剩余时间优先调度算法
短进程优先调度算法改造得到的剥夺式算法。
2. 实时调度
任务的空闲时间　=任务的截止时间　- 任务剩余执行时间　- 当前时间

8、在道数不受限制的多道程序系统中，作业进入系统的后备队列时立即进行作
业调度。现有　4 个作业进入系统，有关信息列举如下，作业调度和进程调度

均采用高优先级算法（规定数值越大则优先级越高）　　。

作业名进入后备队列的时间执行时间　　/min　优先数
Job1 8:00 60 1
Job2 8:30 50 2
Job3 8:40 30 4
Job4 8:50 10 3

试填充下表。（非抢占式，在同一时刻的作业，按优先数来，只要在执行就不能抢占）

作业名 | 进入后备队列
| 执行时间
| 开始执行
| 结束执行
| 周转时间
| 带权周转
| 的时间
| /min |　　时间
| 时间
| /min |　　时间
Job1| 8:00 | 60 | 8:00 | 9:00 | 60 | 1
Job2| 8:30 | 50 | 9:40 | 10:30 | 120 | 2.4
Job3| 8:40 | 30 | 9:00 | 9:30 | 50 | 1.67
Job4| 8:50 | 10 | 9:30 | 9:40 | 50 | 5
平均周转时间
T=(60+120+50+50)/4=70
带权平均周转时间
W=(1+2.4+1.67+5)/4=2.52

非抢占式详细分析：

J1 在 8:00 来了，就一直执行到　9 点，即使 J2，J3，J4 来了就算你们的优先数再多在高也不能
抢占，还不是要看我　J1 执行完了才能执行哈。待　J1 执行完了之后，看你们的优先数多就把 CPU让给谁，结果花落给 J3，让 J2 和 J4 苦等了几十分钟，　就这样推下去，　J3 完成了就是 J4

了。最后才把　CPU给 J2。终于大团圆结局！别忘了计算啊。

注意：高级调度就是作业调度，低级调度就是进程调度。

试填充下表。（抢占式，在同一时刻的多个作业，按优先数来，谁多谁就可以抢占）

作业名 | 进入后备队列
| 执行时间
| 开始执行
| 结束执行
| 周转时间
| 带权周转
| 的时间
| /min | 时间
| 时间
| /min | 时间
Job1| 8:00 | 60 | 8:00 | 10:30 | 150 | 2.5
Job2| 8:30 | 50 | 8:30 | 10:00 | 90 | 1.8
Job3| 8:40 | 30 | 8:40 | 9:10 | 30 | 1
Job4| 8:50 | 10 | 9:10 | 9:20 | 30 | 3
平均周转时间
T=(150+90+30+30)/4=75
带权平均周转时间
W=(2.5+1.8+1+3)/4=2.075

17、如果在限制为两道的多道程序系统中，有 4 个作业进入系统，其进入系统时间、估计运行时间列于下表中，系统采用 SJF 作业调度算法，采用 SRTF（最短剩余时间优先算法）进程调度算法，请填充下表。（分析指出：最短剩余时间优先是指在多个进程同一时刻所需要的剩余时间最多，比如在 10:05 有 J1 和 J2 转化为进程，而 J2 的剩余时间最短先执行，以此类推。）

作业
| 进入系统时间
| 估计运行时间 /min | 开始运行时间
| 结束运行时间
| 周转时间 /min

Job1| 10:00 | 30 | 10:00 | 11:05 | 65

Job2| 10:05 | 20 | 10:05 | 10:25 | 20

Job3| 10:10 | 5 | 10:25 | 10:30 | 20

Job4| 10:20 | 10 | 10:30 | 10:40 | 20

平均周转时间

T=(65+20+20+20)/4=31.25

带权周转时间

W=(65/30+20/20+20/5+20/10)/4=2.2925

第三章同步、通信与死锁

经典问题源码分析：

1、生产者与消费者问题：

一个生产者与一个消费者共享一个缓冲池问题：

源码 :

```
1. int B;
2. semaphore empty;//      可以使用的空缓冲区数
3. semaphore full;//         缓冲区内可以使用的产品数
4. empty=1;//     缓冲区内允许放入一件产品
5. full=0;//        缓冲区内没有产品
6. cobegin
7. process producer(){ process consumer(){
8. while(true){ while(true) {
9. produce( ); P(full);
10. P(empty); take( ) from B;
11. append( ) to B; V(empty);
12. V(full); consume( );
13.} }
14.} }
```

多个生产者与多个消费者共享多个缓冲池问题：

源码：

```
1. item B[k];
2. semaphore empty; empty=k;//        可以使用的空缓冲区数
3. semaphore full; full=0;//              缓冲区内可以使用的产品数
```

4. semaphore mutex; mutex=1;//　　互斥信号量
5. int in=0;//　　放入缓冲区指针
6. int out=0;
// 取出缓冲区指针
7. cobegin
8. process producer_i ( ){ process consumer_j ( ){
9.
while(true) { while(true)
{
10. produce( );
P(full);
11. P(empty);
P(mutex);
12.
P(mutex); take( ) from
B[out];
13. append to B[in];
out=(out+1)%k;
14. in=(in+1)%k;
V(mutex);
15. V(mutex);
V(empty);
16. V(full);
consume( );
17.}
}
18.}
}
19. coend
2、读者与写者问题：
源码：

1. intreadcount=0;//　　读进程计数
2. semaphore writeblock,mutex;
3. writeblock=1;mutex=1;
4. cobegin
5. process reader_i( ){
6. P(mutex);
7. readcount++;
8. if(readcount==1)
9. P(writeblock);
10. }
11. V(mutex);

12.{ 读文件 };
13. P(mutex);
process writer_j( ){
P(writeblock);

{ 写文件 };
V(writeblock);
14. readcount--;
15. if(readcount==0)
16. V(writeblock);
17. V(mutex);
}
18. 一个经典同步问题：吸烟者问题　　(patil
, 1971
)。3 名吸烟者在一个房间内，还有一个香烟供应者。为了制造并抽掉香烟，每

个吸烟者需要三样东西：烟草、纸和火柴，供应者有丰富货物提供。三个吸烟

者中，第一个有自己的烟草，第二个有自己的纸和第三个有自己的火柴。供应

者随机地将两样东西放在桌子上，允许一个吸烟者进行对健康不利的吸烟。当

吸烟者完成吸烟后唤醒供应者，供应者再把两样东西放在桌子上，唤醒另一个

吸烟者。试采用信号量和　　PV操作来编写他们同步工作的程序。

解：分析：同时有效的只有一个生产者和一个消费者，所以这是一个生产者

者与一个消费者共享一个缓冲池问题。

Semaphore
empty=1,full1=full2=full3=0;//full1　　是拥有烟草的，　full2　是拥有纸的　,full3　是拥有火柴的
process producer(){
P(empty);
if(　放 paper
& match)

V(full1);//　　唤醒
else{if(　　放 paper
&烟草)V（full2　）;
elseV(full3)　　；}
process　烟草（）　{

P（full1 ）；
取
paper &match ；
V(empty);

卷烟；
吸烟；
}
process 火柴（） {
P（full1 ）；
取
paper & 烟草；
V(empty);

卷烟；
吸烟；

}
processpaper （） {
P（full1 ）；
取烟草
&match；
V(empty);

卷烟；
吸烟；

}

20. 有三组进程 Pi 、 Qj、 Rk，其中 Pi
、 Qj 构成一对生产者和消费者，共享一个由 M1 个缓区构成的循环缓冲池 buf1

。 Qj、 Rk 凡构成另一对生产者和消费者，共享一个由 M2

个缓冲区构成的循环缓冲池 buf2

。如果 Pi 每次生产一个产品投入 buf1,Qj 每次从中取两个产品组装成一个后并投

入 buf2， Rk 每次从中取三个产品包装出厂 . 试用信号量和 P

、 V 操作写出它们同步工作的程序。

解：分析：这是多个生产者与多个消费者共享多个缓冲区问题。　　(此

题考试概率为　0。)

```
int c1=c2=0;
Semaphore empty1=k,empty2=m,full1=full2=0,mutex1=mutex2=1;
process Pi(){
P（empty1）;
P(mutex1);
放产品；
c1++;
if(c1==2){c1=0;V(full1);}
V(mutex1);
}
process Qj(){
P(full1)　　;
P(mutex1);
取产品两个　;
V(mutex1);
V(empty1);
V(empty1);

P（empty2）;
P（mutex2）;
放产品；

c2++;

if(c2==3){c2=0;V(full2);}
V(mutex2);
}
Process Rk(){


P(full2)　　;

P(mutex1);

取产品　3 个;

V(mutex2);
V(empty2);
V(empty2);
```

V(empty2);
}

银行家算法是解决死锁的关键途径。

28. 设当前的系统状态如下，此时 Available(1,1,2) 。

进程
Claim Allocation
R1R2R3 R1R2R3
P1 322 100
P2 613 511
P3 314 211
P4 422 002

(1) 计算各个进程还需要的资源数 Cki-Aki 。
(2) 系统是否处于安全状态，为什么？
(3) 进程 P2 发出请求向量 request2(0,0,1), 系统能把资源分配给它吗？
(4) 若在进程 P2 申请资源后, P1 发出请求向量 request1(1,0,1),
系统能把资源分配给它吗？
(5) 若在进程 P1 申请资源后, P3 发出请求向量 request3(0,0,1),
系统能把资源分配给它吗？
解:(1)Need=Cki-Aki, 如图所示：

进程
Claim Allocation Need Availabler

R1R2R3 R1R2R3 R1R2R3 1 1 2
P1 322100 222

P2 613511 102
P3 314211 103
P4 422002 420

解:(2) 寻找一个安全序列如（ P2→P1→P3→P4），如图所示：
进程
Claim Allocation Need Availabler

R1R2R3 R1R2R3 R1R2R3 1 1 2
P1 322100 222 623
P2 613511 102 723

P3 314211 103 934

P4 422002 420 936

分析：可分配的是（1,1,2 ）, 符合条件的只有　P2, 从 P2 开始，算出下一个　Available=(1,1,2)+（5,1,1 ）=（6,2,3 ）。这时，符合条件都可以，所以随便选取一个进程来计算　Available　。依次类推，找到一个安全序列（　　P2→P1→P3→P4），所以系统是安全的。

解:(3)
寻找一个安全序列如（　P2→P1→P3→P4），因为　request2(0,0,2)　　向量的各个值都小于（ 1,1,2 ）, 由（ 2）知可以找到一个安全序列（　　P2→P1→P3→P4），系统是安全的，可以分配资源给它。

(4) 同理可证，系统不能分配资源给它，因为找不到一个安全序列。
(5) 同理可证，系统不能分配资源给它，因为找不到一个安全序列。
36. 独木桥问题　1：东西向汽车驶过独木桥，为了保证交通安全，只要桥上无车，则允许一方的汽车过桥，待其全部过完后，才允许另一方的汽车过桥。请用

信号量和　PV操作写出汽车过独木桥问题的同步算法。

解：分析：这是读者与写者的两类读者问题。

```
Semaphore lmutex=rmutex=1,bridge=0;
int lcar=rcar=0;

process lcar(){

P(lmutex);

lcar++;

P(bridge);

V(lmutex);

if (lcar==1){

过桥 ;P(lmutex);lcar--;}
if(lcar==0) {V(bridge)　　；V(lmutex);}
}
process rcar(){
P(rmutex);
rcar++;
```

P(bridge);
V(rmutex);
if (rcar==1){
过桥 ;P(rmutex);rcar--;}
if(rcar==0) {V(bridge)　　；V(rmutex);}
}

37. 独木桥问题　2：在独木桥问题　1 中，限制桥面上最多可以有　　k 辆汽车通过　. 试采用信号量和　PV操作写出汽车过独木桥问题的同步算法。
解：分析指出：这是读者与写者的多个读者问题；

Semaphorelmutex=rmutex=1,bridge=0　，　mutex=k;
int lcar=rcar=0;
process lcar(){
P(lmutex);
lcar++;
P(bridge);
V(lmutex);
if (lcar==1){
P(mutex)　；
过桥 ;
V(mutex);
P(lmutex);lcar--;}
if(lcar==0) {V(bridge)　　；V(lmutex);}

}
process rcar(){
P(rmutex);
rcar++;
P(bridge);
V(rmutex);
if (rcar==1){
P(mutex)　；过桥；
V(mutex);P(rmutex);rcar--;}
if(rcar==0) {V(bridge)　　；V(rmutex);}
}

44. 桌上有一只盘子，　每次只能放入一只水果。　爸爸专向盘子里放苹果　（Apple ）, 妈妈专向盘子放橘子，一个儿子专等吃盘子里的橘子，一个女儿专等吃盘子
里的苹果。写出爸爸妈妈儿子女儿正确同步互斥的算法。
解：分析：同时有效的是一个生产者和一个消费者，所以这是一个生产者与一个消费者共享一个缓冲池问题。爸爸与妈妈是互斥关系，儿子与妈妈是同步关

系，女儿与爸爸是同步关系，儿子与女儿无关系。

Semaphoreempty1=1,full1=full2=0;//full1　　代表女儿，　full2　代表儿子

process Father(){

P(empty);

放苹果；

V(full1);//　　唤醒女儿

}

process Mother(){

P(empty);

放橘子；

V(full2);//　　唤醒儿子

}

process Daugther(){

P(full1)　　；

取苹果；

V(empty)　；

吃苹果；

}

process Son(){

P(full2)　　；

取橘子；

V(empty)　；

吃橘子；

}

45.

桌上有一只盘子，最多可以容纳两个水果，每次仅能放入或取出一个水果。爸爸专向盘子里放苹果（　Apple），妈妈专向盘子放橘子，一个儿子专等吃盘子里的橘子，一个女儿专等吃盘子里的苹果。写出爸爸妈妈儿子女儿正确同步互斥

的算法。

解：分析：这是多个生产者与多个消费者共享多个缓冲池问题。

Semaphore empty1=k,empty2=m,full1=full2=0,mutex1=mutex2=1;

//full1　　代表女儿，　full2　代表儿子

process Father(){

P(empty1);
P(mutex1);

放苹果；

V(mutex1);

V(full1);//　　唤醒女儿

}
process Mother(){
P(empty2);
P(mutex2)

放橘子；

V(mutex2);

V(full2);//　　唤醒儿子

}
process Daugther(){

P(full1)　；
P(empty1)　；
取苹果；
V(empty1)　；
V(mutex1)　；

吃苹果；

}
process Son(){

P(full2)　；
P(empty2)　；
取橘子；
V(empty2)　；
V(mutex2)　；
吃橘子；

}

第四章存储管理

页面置换算法详细介绍：

1) 最佳页面替换算法　OPT
淘汰将来再也不访问或长时间不访问的页面。

2) 先进先出页面替换算法　FIFO
选择驻留主存时间最长的页面进行置换。
3) 最近最少用页面替换算法　LRU
淘汰最近未使用的页面。
每个页面设置一个多位计数器，又叫最不常用页面替换算法　LFU。每当访问一页时，就使它对应的计数器加１。

4) 第二次机会页面替换算法　SCR
5) 时钟页面替换算法　Clock
当页面被访问时，计数器加　1，淘汰时，从当前位置开始循环扫描队列，若页面的访问位位　1，则将它重新置　0，在检查下一个，若访问位为　0，则它就是将被淘汰的那个页面
6、某计算机有　cache、内存、辅存来实现虚拟存储器。如果数据在　cache 中，访问它需要　20ns；如果在内存但不在　cache，需要　60ns 将其装入缓存，然后才能访问；如果不在内存而在辅存，需要　12us 将其读入内存，然后，用　60ns 再读入　cache，然后才能访问。假设　cache 命中率为　0.9，内存命中率为　0.6，则数据平均访问时间是多少（　ns)。

解：ta 为平均访问时间，　tc 为缓存时间，　tm 为主存时间，　th 为辅存时间　,h 为命

中率。命中： tc+tm ；不命中 :tc+2tm ；

ta=hctc+(1-hc)[hm(tc+tm)+(1-hm)(2tc+2tm+th)]

代入数据计算得 ta=506ns 。

解析：先判断是否越界，如没有就计算其物理地址，反之，中断。

如在 [0,430] 中，因为 430<600，没有发生越界中断，所以物理地址

为 219+430=649.

15、在一分页存储管理系统中，逻辑地址长度为 16 位，页面大小为 4096 字节，现有一逻辑地址为 2F6AH，且第 0、1、2 页依次存在物理块 10、12、14 号中，问相应的物理地址为多少？

解：因为逻辑地址长度为 16 位，而页面大小为 4096（12 位）字节，所以，前面的 4 位表示页号。把 2F6AH 转换成二进制为： 00
10 1 1 11 0110 1010 ，可知页号为 2。故放在 14 号物理块中，写成十六进制为： EF6AH（页内地址 =块内地址）。

注意：若改变页面大小为 8kb（13 位）呢，页号就为 3, 其他的类似，最后的结果为 18F6AH.

若改变一逻辑地址为 9000 呢，请看图：
物理地址 =块号 *块长 +块内地址（块长 =页长）

So,PA=14*4096+808=58152.

31. 设程序大小为 460 个字，考虑如下访问序列：
55,20,108,180,79,310 ， 170,255,246,433,488,369
(1) 设页面大小为 100 个字，试给出访问序列的页面走向。
(2) 假设程序可用主存为 200 个字，采用 FIFO 先进先出、最近不常使用算法 LRU求命中率。
解： (1) 用访问序列除以 100 取整，页面走向为 0,0,1,1,0 ， 3,1,2,2,4 ,4,3.

(2)m=2 采用 FIFO 算法，一直往下挤，命中一次记一次。 LRU 算法，一直往下挤，命中一次记一次但要提在第一位：请看图

：

m=2FIFO LRU

0 0 1 1 0 3 1 2 2 4 4 3

0 0 1 1 0 3 1 2 2 4 4 3

0 0 1 1 1 3 3 2 2 4 4

3

0 0 1 1 0 3 1 2 2 4 4 3

0 0 0 1 1 3 3 2 2

4

0 0 1 0 3 1 1 2 2 4

红色代表命中：

命中率： 6/12=50%

红色代表命中：

命中率： 6/12=50%

47 题中的 m=4,FIFO、 LRU算法与上面类似。

第五章设备管理

算法回顾详细介绍：

移臂调度策略算法（核心）

注意磁头方向

1.

电梯调度算法（一级一级地向磁头方向扫描，不到最后一个，然后反向扫描）；

2.

最短查找时间优先算法 SSTF(先执行查找时间最短的那个磁盘请求)；

3.

先来先服务算法 FCFS（先来的先执行请求） ；

4.

扫描算法 SCAN(类似电梯调度算法，但是要扫描最后一个) ；

5.

分步扫描算法

6. 循环扫描算法

1

旋转型设备上信息的优化分布能减少为若干个拍服务的总时间．设磁鼓上分为 2
0V 个区，每区存放一个记录，磁鼓旋转一周需 20
毫秒，读出每个记录平均需用 1 毫秒，读出后经 2
毫秒处理，再继续处理下一个记录。在不知当前磁鼓位置的情况下：

(1 ）顺序存放记录 1、、、记录 20 时，试计算读出并处理 20
个记录的总时间； (2 ）给出优先分布 20 个记录的一种方案， 使得所花的总处理时间减少， 且
计算出这个方案所花的总
时间。
答：（1）定位第 1 个记录需 t 定=10m
s。读出第 1 个记录，处理花 2ms，这时已到了第 4 个记录，再转过 18 个记录（花 18ms）才
能找到记录 2，所以，读出并处理 20 个记录的总时间： 10 + 3 + (l + 2 + 18) *
19 = 13 + 2 1 * 19 =412ms ；（2）如果给出优先分布 20
个记录的方案为： 1 , 8 , 15 , 2 , 9 , 16 , 3 , 10 , 17 , 4 ,

11, 18, 5 , 12 , 19 , 6,13,20,7, 14 。当读出第 1
个记录，花 2ms 处理后，恰好就可以处理记录 2，省去了寻找下一个记录的时间，读出并处
理 20 个记录的总时间：

10+3+3*19=13+57=70ms .

（t1=1+2=3ms;t2=18*1+3=21ms,t 顺=t1+19t2+t 定=412ms）
t 优化=20t1+t 定=70ms.(每隔两个放一个，节约时间开销)。

7、假定磁盘有 200 个柱面，编号 O
-199 ，当前存取臂的位置在 143
号柱面上，并刚刚完成了 125
号柱面的服务请求，如果请求队列的先后顺序是： 86
, 147 , 91 , 177 , 94
, 150 , 102 , 175 , 130

；试问：为完成上述请求，下列算法存取臂移动的总量是多少？并算出存取臂
移动的顺序。

(1 ）先来先服务算法 FCFS;(2 ）最短查找时间优先算 SSTF；(3 ）扫描算法 SCAN;
(4 ）电梯调度算法。
解：(l ）先来先服务算法 FCFS 存取臂移动的总量为 565，依次为
143-86-147-91-177-94-150-102-175
130。(2 ）最短查找时间优先算法 SSTF为 162，依次为 143-147-150

-130-102-94-91-86-175-177 。(3）扫描算法 SCAN存取臂移动的总量为 169，依次为 143

-147 -150 -175 -177 -199 -130 -102 -94

91 -86 。(

4）电梯调度存取臂移动的总量为 125，依次为 143

147 -150 -175 -177 -130-102 -94 -91 -86 。

第六章文件管理

16. 如果一个索引节点为 128B，指针长 4B，状态信息占用 68B，而每块大小为 8KB。问在索引节点中有多大空间给指针？使用直接、一次间接、二次间接和三次间接指针分别可表示多大的文件？

解：由于索引节点为 128B，而状态信息占用 68B，故索引节点中用于磁盘指针的空间大小为：

128-68

= 60 字节。

直接指针项数为： 60

/ 4 -3 = 12 个。每块大小为 8KB.

直接指针时： 12

*8KB= 96KB;

一次间接指针时 :(8KB/4)*8KB

=16MB;

二次间接指针时 :(8KB/4)*(8KB/4)*8KB=32GB;

三次间接指针时 :(8KB/4)*(8KB/4)*(8KB/4)

*8KB=16TB.

7. 在一个操作系统中，inode 节点中分别含有 10 个直接地址的索引和一、二、三级索引。若设每个盘块有 512B 大小，每个盘块中可存放 128 个盘块地址，则一个 1MB的文件占用多少间接盘块？一个 25MB的文件占用多少间接盘块？

解： 1MB/512B=2KB

25MB/512B=50KB

第一章操作系统概论

8. 若内存中有 3 道程序 A、B、C，优先级从高到低为 A、B 和 C，它们单独运行时的 CPU和 I/O 占用时间为：
如果三道程序同时并发执行，调度开销忽略不计，但优先级高的程序可中断优
先级低的程序，优先级与 I/O
设备无关。试画出多道运行的时间关系图，并问最早与最迟结束的程序是哪个
？每道程序执行到结束分别用了多少时间？计算三个程序全部运算结束时的 CPU
利用率？答：画出三个作业并发执行的时间图：

(l ）最早结束的程序为 B，最后结束的程序为 C。
(2 ）程序 A 为 250ms
。程序 B 为 220ms。程序 C 为 310ms。
(3) CPU 利用率为（ 310
-120) /
310 = 61.3 % 有两个程序， A 程序按顺序使用： (
CPU)10秒、（设备甲） 5
秒、（CPU） 5 秒、（设备乙） 10 秒、（CPU） 10
秒。 B 程序按顺序使用： （设备甲） 10 秒、（CPU） 10 秒、（设备乙） 5 秒、 (
CPU)5秒、（设备乙） 10 秒。在顺序环境下先执行 A，再执行 B

，求出总的 CPU利用率为多少？答：程序 A 执行了 40 秒，其中 CPU用了 25
秒。程序 B 执行了 40 秒，其中 CPU用了 15 秒。两个程序共用了 80 秒， CPU化
40 秒。故 CPU利用率为 40/80
=50%。

相信自己，路就在脚下！