# 实验（一） 操作系统的进程调度

19122557 王波

## 1. 目的

进程是操作系统最重要的概念之一，进程调度又是操作系统核心的主要内容。本实习要求学生独立地用高级语言编 写和调试一个简单的进程调度程序。调度算法可任意选择或自行设计。例如，简单轮转法和优先数法等。本实习可加深 对于进程调度和各种调度算法的理解。

## 2. 要求

（1） 设计一个有 n 个进程工行的进程调度程序。每个进程由一个进程控制块（PCB）表示。进程控制块通常应包含 下述信息：进程名、进程优先数、进程需要运行的时间、占用 CPU 的时间以及进程的状态等，且可按调度算法 的不同而增删。

（2） 调度程序应包含 2～3 种不同的调度算法，运行时可任意选一种，以利于各种算法的分析比较。

（3） 系统应能显示或打印各进程状态和参数的变化情况，便于观察诸进程的调度过程

## 3. 题目要求

本程序可选用优先数法或简单轮转法对五个进程进行调度。每个进程处于运行 R(run)、就绪 W(wait)和完成 F(finish) 三种状态之一，并假设起始状态都是就绪状态 W。为了便于处理，程序进程的运行时间以时间片为单位计算。各进程的优先数或轮转时间片数、以及进程需要运行的时间片数，均由伪随机数发生器产生。

## 4. 算法分析

### 1. 动态优先数法

1、进程就绪队列按优先数大小从高到低排列，链首进程首先投入运行

2、进程每执行一次，进程需要的时间片数减1、该进程的优先数减3

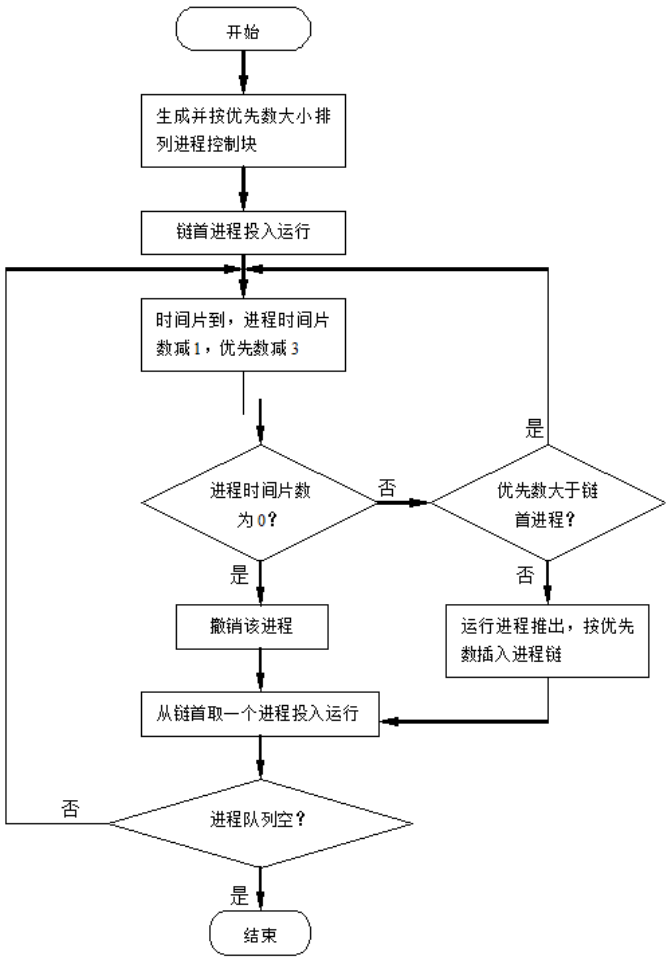
3、若该进程执行完毕，放入完成队列，继续取出就绪队列头进程投入运行

4、若未完成，则把该进程有序插入就绪队列，取出头进程投入运行

5、重复上述操作，当所有进程都运行完毕

每次取出优先级最高的进程投入运行，执行一次后接着仍是用该进程降低一级 后的优先数与就绪队列中链首进程的优先数进行比较，如果仍是该进程的优先数高或相同，便让该进程继续执行；否则，调度就绪队列的链首进程投入运行。原运行过的进程按其现 行优先数大小插入就绪队列，且改变它们对应的进程状态，一直到所有进程都运行完各自的时间数。

程序框图如下：




## 2. 时间片轮转法

1、进程就绪队列按各进程进入的先后顺序排列，取出头进程投入运行

2、每执行一次，进程所需时间片数减1，占用CPU时间片数加1

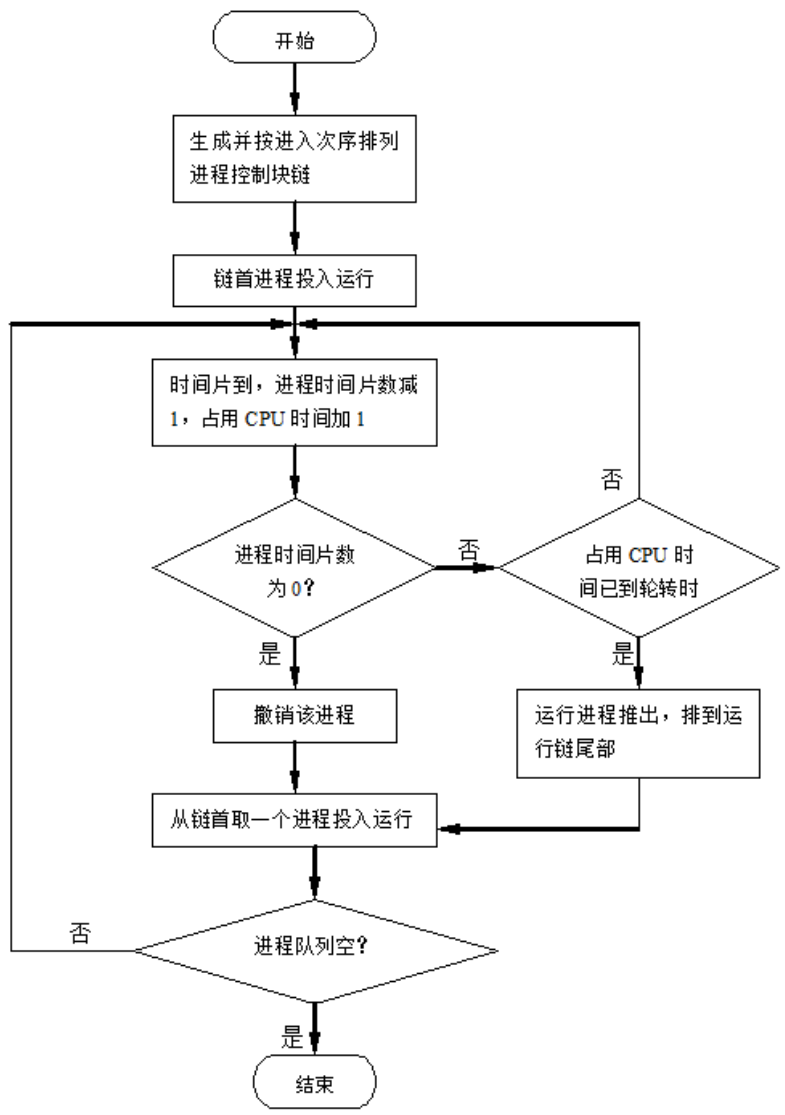
3、若进程未执行完毕，若占用CPU时间片数<轮转时间片，则该进程继续运行

4、否则则停止运行，投入就绪队列尾部，取出头进程运行

5、若进程执行完毕，则投入完成队列，从就绪队列取出头进程运行

进程每次所需处理机的轮转时间片数记入进程控制块中的轮转时间片数记录项。进程执行时，每运行一个时间片，进程还需要的时间片数减1，运行进程占用处理机的时间片数加1，然后比较占用CPU的时间片数是否与该进程的轮转时间片数相等，若相等则说明已达到轮转时间，应将现运行的进程排列就绪队列的末尾，调度队列上的首进程运行，且改变它们的进程状态，直至所有进程完成各自的时间片。

程序框图如下：



# 5. 算法实现

对于本次实验，我采用python语言进行，通过设置PCB类和PCBChain类来完成实验要求。

对于题目中要求的结构体实现如下：

```
class PCB(object):
      def __init__(self, ID=0, PRIORITY=0, CPUTIME=0, ALLTIME=0, STATE='W', NEXT=0, FORWARD=0):
            self.ID = ID
            self.PRIORITY = PRIORITY   # 优先级
            self.CPUTIME = CPUTIME   # CPU时间
            self.ALLTIME = ALLTIME   # 运行总时间
            self.STATE = STATE   # 进程状态
            self.NEXT = NEXT # 连接下一个进程的指针
            self.FORWARD = FORWARD
```

对于题目中，动态优先数法实现如下：

```
      def __priority_running(self):
            # 时间片到了
            self.Chain[0].STATE = "R"
            self.Chain[0].ALLTIME -= 1
            self.Chain[0].PRIORITY -= 3
            if self.Chain[0].ALLTIME <= 0:
                  self.Chain.remove(self.Chain[0])
                  if self.Chain.__len__() >= 1:
                        self.Chain[0].STATE = "R"
                  return
            if self.Chain.__len__() >= 2 and self.Chain[0].PRIORITY < self.Chain[1].PRIORITY:
                  self.Chain[0].STATE = 'W'
                  self.Chain[1], self.Chain[0] = self.Chain[0], self.Chain[1]
                  self.Chain[0].STATE = 'R'
                  return
            return

      # 对队列中的排序算法实现
      def _order_by_priority(self):
            for j in range(0, self.Chain.__len__() - 1):
                  c = 0
                  for i in range(0, self.Chain.__len__() - 1 - j):
                        if self.Chain[i].PRIORITY < self.Chain[i + 1].PRIORITY:
                              self.Chain[i], self.Chain[i + 1] = self.Chain[i + 1], self.Chain[i]
                              c += 1
                        if c == 0:
                              break
```

首先，先把位于队列首部的状态变换为‘R’(表示正在运行)，随后对于该对象的属性所需时间和优先级进行修改，最后判断该对象是否完成，从队列中移除。特别的，当队列中，只剩余一个元素的时候，就直接进行运行，没有必要的优先级比较。

时间片轮转法实现如下：

```
      def __round_robin_running(self):
            # 时间片到了
            self.Chain[0].STATE = "W"
            self.Chain[0].CPUTIME += 1
            self.Chain[0].ALLTIME -= 1
            # self.Chain
            if self.Chain != [] and self.Chain[0].ALLTIME <= 0:
                  self.Chain.remove(self.Chain[0])
            if self.Chain != []:
                  self.Chain.append(self.Chain.pop(0))
                  self.Chain[0].STATE = "R"
            return
```

对于该部分的算法实现很简单，主要修改队列的先后顺序，党队列中任务运行完成后移除队列，而且，该部分的代码在运行过程中也需要动态优先数算法的相关部分参与，源码在试验最后给出。对于文中要求的伪随机数的给出也在实验中有所体现。

# 6. 实验总结

操作系统是计算机系统中必不可少的系统软件。它是计算机系统中各种资源的管理者和各种活动的组织者、指挥者。操作系统采用时间片法调度进程,使系统资源得到充分的利用,用户也可以花更少的时间完成更多的工作，这次模拟系统调度进程，让我们明白了系统时间片的调度方法和p,v原语操作情况，对操作系统理论的学习更加深一层.

# 7. 思考题

（1） 示例中的程序，没有使用指针型（pointer）数据结构，如何用指针型结构改写本实例，使更能体现 C 语言的特性。

答：在本次实验中，我所采取的是通过优先队列来实现优先级的先后顺序的，如果要使用pointer，则需要设置一个链表，而且在每一次插入删除链表的操作中，都需要对于该链表进行重新排序。

（2） 如何在程序中真实地模拟进程运行的时间片?

答：可以在每一次时间片运行结束的时候，对还未完成的任务，在队列中重新排序，去寻找下一个进程。

（3） 如果增加进程的“等待”状态，即进程因请求输入输出等问题而挂起的状态，如何在程序中实现?

答：可以对输入输出设置一个临界资源，当一个进程在使用该临界资源的时候，其他进程必须等待，可以在结构体中增加一个on模块，初始化on为false，当使用资源的时候为true，使用完毕为false。

# 8. 源码及结果展示

```
import random



class PCB(object):
      def __init__(self, ID=0, PRIORITY=0, CPUTIME=0, ALLTIME=0, STATE='W', NEXT=0, FORWARD=0):
            self.ID = ID
            self.PRIORITY = PRIORITY   # 优先级
            self.CPUTIME = CPUTIME   # CPU时间
            self.ALLTIME = ALLTIME   # 运行总时间
            self.STATE = STATE   # 进程状态
            self.NEXT = NEXT
            self.FORWARD = FORWARD


class PCBChain(object):
      def __init__(self):
            self.RUN = None
            self.HEAD = None
            self.TAIL = None
            self.Chain = []
            self.current_alg = None
            self.__verify = PCB()
            self.WAITING_QUEUE = []
            self.FINISH = False

      # def __gen_header(self,pcb1,pcb2):
      #       pcb1.FOWARD = None
      #       pcb1.NEXT = pcb2
      #       pcb2.FORWARD = pcb1
      #       self.HEAD= pcb1
```

```
# def connect(self,pcb1,pcb2):
#       CURRENT_NODE = self.HEAD
#       if self.HEAD is None:
#             self.__gen_header(pcb1,pcb2)
#             return 1
#       else:
#             while CURRENT_NODE is None:
#                   CURRENT_NODE=CURRENT_NODE.NEXT
#             CURRENT_NODE.NEXT =

def append(self, pcb):
      '''
     后来觉得写的有问题，就优先级用了顺序表的方法，时间片用了链表的方法
     :param pcb:
     :return:
     '''
      if pcb.__class__ == self.__verify.__class__:
            self.Chain.append(pcb)
      else:
            print("type error")

def ChangeAlg(self, alg):
      # print(self.Chain)
      if self.Chain != []:
            if alg in ["priority", "round robin"]:
                  self.current_alg = alg
            else:
                  print("type error")
      else:
            print("Append first")

def prepare(self):
      # print(self.current_alg)
      if self.current_alg == "priority":
            self._order_by_priority()
            # 设置初始指针
            # self.RUN = self.Chain[0]
            # self.HEAD = self.Chain[1]
            # self.TAIL = self.Chain[-1]
      self.Chain[-1].NEXT = None
      for i in range(self.Chain.__len__() - 1):
            self.Chain[i].NEXT = self.Chain[i + 1]


            self.Chain[0].STATE = "R"

      def running(self):
            print("RUNNING PROC     WATING QUEUE")
            print("{}                        {}".format("".join([str(word.ID) + " " for word in self.Chain])
[:2],
                                                      "".join([str(word.ID) + " " for word in
self.Chain])[2:]))
            print("==========================")
            print("ID                        {}".format("".join([str(word.ID) + " " for word in self.Chain])))
            print("PRIORITY              {}".format("".join([str(word.PRIORITY) + " " for word in
self.Chain])))
            print("CPUTIME               {}".format("".join([str(word.CPUTIME) + " " for word in
self.Chain])))
            print("ALLTIME               {}".format("".join([str(word.ALLTIME) + " " for word in
self.Chain])))
            print("STATE                 {}".format("".join([str(word.STATE) + " " for word in
self.Chain])))
```

```
            # print("NEXT                    {}".format("".join([str(word.NEXT.ID) + " " for word in
self.Chain if word.NEXT != None])) + "0")

            if self.current_alg == "priority":
                  self.__priority_running()

            elif self.current_alg == "Round Robin".lower():
                  self.__round_robin_running()

            if self.Chain == []:
                  self.FINISH = True

            print("==========================")

      def __priority_running(self):
            # 时间片到了
            self.Chain[0].STATE = "R"
            self.Chain[0].ALLTIME -= 1
            self.Chain[0].PRIORITY -= 3
            if self.Chain[0].ALLTIME <= 0:
                  self.Chain.remove(self.Chain[0])
                  if self.Chain.__len__() >= 1:
                        self.Chain[0].STATE = "R"
                  return
            if self.Chain.__len__() >= 2 and self.Chain[0].PRIORITY < self.Chain[1].PRIORITY:
                  self.Chain[0].STATE = 'W'
                  self.Chain[1], self.Chain[0] = self.Chain[0], self.Chain[1]
                  self.Chain[0].STATE = 'R'
                  return
            return

      def __round_robin_running(self):
            # 时间片到了
            self.Chain[0].STATE = "W"
            self.Chain[0].CPUTIME += 1
            self.Chain[0].ALLTIME -= 1
            # self.Chain
            if self.Chain != [] and self.Chain[0].ALLTIME <= 0:
                  self.Chain.remove(self.Chain[0])
            if self.Chain != []:
                  self.Chain.append(self.Chain.pop(0))
                  self.Chain[0].STATE = "R"
            return

      def _order_by_priority(self):
            for j in range(0, self.Chain.__len__() - 1):
                  c = 0
                  for i in range(0, self.Chain.__len__() - 1 - j):
                        if self.Chain[i].PRIORITY < self.Chain[i + 1].PRIORITY:
                              self.Chain[i], self.Chain[i + 1] = self.Chain[i + 1], self.Chain[i]
                              c += 1
                        if c == 0:
                              break


# class TestAlg():
#       def __init__(self, matirx):
#             '''
#             matrix is a mat with
#             :param matirx:
#             '''
#             try:
```

```
#                     np.array()


def testing_mode_1():
      # print("TYPE THE ALGORITHM:")
      # 对于文中要求伪随机数体现
      pcb1 = PCB(ID=1, PRIORITY=random.randint(0, 9), ALLTIME=3)
      pcb2 = PCB(ID=2, PRIORITY=random.randint(0, 9), ALLTIME=4)
      pcb3 = PCB(ID=3, PRIORITY=random.randint(0, 9), ALLTIME=6)
      pcb4 = PCB(ID=4, PRIORITY=random.randint(0, 9), ALLTIME=3)
      pcb5 = PCB(ID=5, PRIORITY=random.randint(0, 9), ALLTIME=4)
      alg = input("TYPE THE ALGORITHM:").lower()
      # elif self.current_alg == "Round Robin".lower():
      pcb_chain = PCBChain()
      pcb_chain.append(pcb1)
      pcb_chain.append(pcb2)
      pcb_chain.append(pcb3)
      pcb_chain.append(pcb4)
      pcb_chain.append(pcb5)
      # alg = "Round Robin".lower()
      # alg = "priority"
      if alg == "priority".lower():
            print("====================")
            pcb_chain.ChangeAlg(alg)
            pcb_chain.prepare()
            while pcb_chain.FINISH == False:
                  pcb_chain.running()
            # print(pcb_chain.current_alg)
      elif alg == "Round Robin".lower():
            print("====================")
            pcb_chain.ChangeAlg(alg)
            pcb_chain.prepare()
            while pcb_chain.FINISH == False:
                  pcb_chain.running()
      print("SYSTEM FINISH")


if __name__ == '__main__':
      testing_mode_1()
```

动态优先级结果展示:

```
E:\Anaconda3\envs\machine\python.exe E:/Users/id-none/Desktop/OS_homework-master/调度算法/index.py
TYPE THE ALGORITHM:priority
====================
RUNNING PROC     WATING QUEUE
1                          2 3 4 5
============================
ID                       1 2 3 4 5
PRIORITY               4 3 3 9 3
CPUTIME                 0 0 0 0 0
ALLTIME                 3 4 6 3 4
STATE                     R W W W W
============================
RUNNING PROC     WATING QUEUE
2                          1 3 4 5
============================
ID                       2 1 3 4 5
PRIORITY               3 1 3 9 3
CPUTIME                 0 0 0 0 0
ALLTIME                 4 2 6 3 4
STATE                     R W W W W
```

```
============================
RUNNING PROC     WATING QUEUE
1                          2 3 4 5
============================
ID                       1 2 3 4 5
PRIORITY               1 0 3 9 3
CPUTIME                 0 0 0 0 0
ALLTIME                 2 3 6 3 4
STATE                     R W W W W
============================
RUNNING PROC     WATING QUEUE
2                          1 3 4 5
============================
ID                       2 1 3 4 5
PRIORITY               0 -2 3 9 3
CPUTIME                 0 0 0 0 0
ALLTIME                 3 1 6 3 4
STATE                     R W W W W
============================
RUNNING PROC     WATING QUEUE
1                          2 3 4 5
============================
ID                       1 2 3 4 5
PRIORITY               -2 -3 3 9 3
CPUTIME                 0 0 0 0 0
ALLTIME                 1 2 6 3 4
STATE                     R W W W W
============================
RUNNING PROC     WATING QUEUE
2                          3 4 5
============================
ID                       2 3 4 5
PRIORITY               -3 3 9 3
CPUTIME                 0 0 0 0
ALLTIME                 2 6 3 4
STATE                     R W W W
============================
RUNNING PROC     WATING QUEUE
3                          2 4 5
============================
ID                       3 2 4 5
PRIORITY               3 -6 9 3
CPUTIME                 0 0 0 0
ALLTIME                 6 1 3 4
STATE                     R W W W
============================
RUNNING PROC     WATING QUEUE
3                          2 4 5
============================
ID                       3 2 4 5
PRIORITY               0 -6 9 3
CPUTIME                 0 0 0 0
ALLTIME                 5 1 3 4
STATE                     R W W W
============================
RUNNING PROC     WATING QUEUE
3                          2 4 5
============================
ID                       3 2 4 5
PRIORITY               -3 -6 9 3
CPUTIME                 0 0 0 0
ALLTIME                 4 1 3 4
```

```
STATE                    R W W W
============================
RUNNING PROC     WATING QUEUE
3                          2 4 5
============================
ID                       3 2 4 5
PRIORITY               -6 -6 9 3
CPUTIME                 0 0 0 0
ALLTIME                 3 1 3 4
STATE                      R W W W
============================
RUNNING PROC     WATING QUEUE
2                          3 4 5
============================
ID                       2 3 4 5
PRIORITY               -6 -9 9 3
CPUTIME                 0 0 0 0
ALLTIME                 1 2 3 4
STATE                      R W W W
============================
RUNNING PROC     WATING QUEUE
3                            4 5
============================
ID                       3 4 5
PRIORITY               -9 9 3
CPUTIME                 0 0 0
ALLTIME                 2 3 4
STATE                      R W W
============================
RUNNING PROC     WATING QUEUE
4                            3 5
============================
ID                       4 3 5
PRIORITY               9 -12 3
CPUTIME                 0 0 0
ALLTIME                 3 1 4
STATE                      R W W
============================
RUNNING PROC     WATING QUEUE
4                          3 5
============================
ID                       4 3 5
PRIORITY               6 -12 3
CPUTIME                 0 0 0
ALLTIME                 2 1 4
STATE                      R W W
============================
RUNNING PROC     WATING QUEUE
4                            3 5
============================
ID                       4 3 5
PRIORITY               3 -12 3
CPUTIME                 0 0 0
ALLTIME                 1 1 4
STATE                      R W W
============================
RUNNING PROC     WATING QUEUE
3                            5
============================
ID                       3 5
PRIORITY               -12 3
CPUTIME                 0 0
```

```
ALLTIME                1 4
STATE                    R W
============================
RUNNING PROC     WATING QUEUE
5
============================
ID                       5
PRIORITY              3
CPUTIME                0
ALLTIME                4
STATE                    R
============================
RUNNING PROC     WATING QUEUE
5
============================
ID                       5
PRIORITY              0
CPUTIME                0
ALLTIME                3
STATE                    R
============================
============================
RUNNING PROC     WATING QUEUE
5
============================
ID                       5
PRIORITY             -3
CPUTIME                0
ALLTIME                2
STATE                    R
============================
RUNNING PROC     WATING QUEUE
5
============================
ID                       5
PRIORITY             -6
CPUTIME                0
ALLTIME                1
STATE                    R
============================
SYSTEM FINISH

进程已结束,退出代码0
```

时间片轮转法结果展示:

```
E:\Anaconda3\envs\machine\python.exe E:/Users/id-none/Desktop/OS_homework-master/调度算法/index.py
TYPE THE ALGORITHM:Round Robin
====================
RUNNING PROC     WATING QUEUE
1                        2 3 4 5
============================
ID                     1 2 3 4 5
PRIORITY            0 6 7 5 3
CPUTIME              0 0 0 0 0
ALLTIME              3 4 6 3 4
STATE                  R W W W W
============================
RUNNING PROC     WATING QUEUE
2                        3 4 5 1
============================
ID                     2 3 4 5 1
PRIORITY            6 7 5 3 0
```

```
CPUTIME                0 0 0 0 1
ALLTIME                4 6 3 4 2
STATE                    R W W W W
============================
RUNNING PROC     WATING QUEUE
3                          4 5 1 2
============================
ID                       3 4 5 1 2
PRIORITY               7 5 3 0 6
CPUTIME                 0 0 0 1 1
ALLTIME                 6 3 4 2 3
STATE                    R W W W W
============================
RUNNING PROC     WATING QUEUE
4                          5 1 2 3
============================
ID                       4 5 1 2 3
PRIORITY               5 3 0 6 7
CPUTIME                 0 0 1 1 1
ALLTIME                 3 4 2 3 5
STATE                    R W W W W
============================
RUNNING PROC     WATING QUEUE
5                          1 2 3 4
============================
ID                       5 1 2 3 4
PRIORITY               3 0 6 7 5
CPUTIME                 0 1 1 1 1
ALLTIME                 4 2 3 5 2
STATE                    R W W W W
============================
RUNNING PROC     WATING QUEUE
1                          2 3 4 5
============================
ID                       1 2 3 4 5
PRIORITY               0 6 7 5 3
CPUTIME                 1 1 1 1 1
ALLTIME                 2 3 5 2 3
STATE                    R W W W W
============================
RUNNING PROC     WATING QUEUE
2                          3 4 5 1
============================
ID                       2 3 4 5 1
PRIORITY               6 7 5 3 0
CPUTIME                 1 1 1 1 2
ALLTIME                 3 5 2 3 1
STATE                    R W W W W
============================
RUNNING PROC     WATING QUEUE
3                          4 5 1 2
============================
ID                       3 4 5 1 2
PRIORITY               7 5 3 0 6
CPUTIME                 1 1 1 2 2
ALLTIME                 5 2 3 1 2
STATE                    R W W W W
============================
RUNNING PROC     WATING QUEUE
4                          5 1 2 3
============================
ID                       4 5 1 2 3
```

```
PRIORITY              5 3 0 6 7
CPUTIME                1 1 2 2 2
ALLTIME                2 3 1 2 4
STATE                     R W W W W
============================
RUNNING PROC     WATING QUEUE
5                           1 2 3 4
============================
ID                       5 1 2 3 4
PRIORITY              3 0 6 7 5
CPUTIME                1 2 2 2 2
ALLTIME                3 1 2 4 1
STATE                     R W W W W
============================
RUNNING PROC     WATING QUEUE
1                           2 3 4 5
============================
ID                       1 2 3 4 5
PRIORITY              0 6 7 5 3
CPUTIME                2 2 2 2 2
ALLTIME                1 2 4 1 2
STATE                     R W W W W
============================
RUNNING PROC     WATING QUEUE
3                           4 5 2
============================
ID                       3 4 5 2
PRIORITY              7 5 3 6
CPUTIME                2 2 2 2
ALLTIME                4 1 2 2
STATE                     R W W W
============================
RUNNING PROC     WATING QUEUE
4                           5 2 3
============================
ID                       4 5 2 3
PRIORITY              5 3 6 7
CPUTIME                2 2 2 3
ALLTIME                1 2 2 3
STATE                     R W W W
============================
RUNNING PROC     WATING QUEUE
2                           3 5
============================
ID                       2 3 5
PRIORITY              6 7 3
CPUTIME                2 3 2
ALLTIME                2 3 2
STATE                     R W W
============================
RUNNING PROC     WATING QUEUE
3                           5 2
============================
ID                       3 5 2
PRIORITY              7 3 6
CPUTIME                3 2 3
ALLTIME                3 2 1
STATE                     R W W
============================
RUNNING PROC     WATING QUEUE
5                           2 3
============================
```

```
ID                    5 2 3
PRIORITY              3 6 7
CPUTIME                2 3 4
ALLTIME                2 1 2
STATE                    R W W
============================
RUNNING PROC     WATING QUEUE
2                          3 5
============================
ID                    2 3 5
PRIORITY              6 7 3
CPUTIME                3 4 3
ALLTIME                1 2 1
STATE                    R W W
============================
RUNNING PROC     WATING QUEUE
5                          3
============================
ID                    5 3
PRIORITY              3 7
CPUTIME                3 4
ALLTIME                1 2
STATE                    R W
============================
RUNNING PROC     WATING QUEUE
3
============================
ID                    3
PRIORITY              7
CPUTIME                4
ALLTIME                2
STATE                    R
============================
RUNNING PROC     WATING QUEUE
3
============================
ID                    3
PRIORITY              7
CPUTIME                5
ALLTIME                1
STATE                    R
============================
SYSTEM FINISH

进程已结束,退出代码0
```