第二章　实验三

模拟进程队列管理：入队与出队

0. 进程队列概述

- 进程队列的链接

  在多道程序设计的系统中往往会同时创建多个进程。在单处理器的情况下，每次只能让一个进程运行，其他的进程处于就绪状态或等待状态。为了便于管理，经常把处于相同状态的进程链接在一起，称“**进程队列**”。
- 由于进程控制块能标志进程的存在和动态刻画进程的特性，因此，**进程队列可以用进程控制块（PCB）的连接来形成**。
- 出队和入队：当发生的某个事件使一个进程的状态发生变化时，这个进程就要退出所在的某个队列而排入到另一个队列中去。一个进程从所在的队列退出的操作称为**出队**；一个进程排入到一个指定的队列的操作称为**入队**；系统中负责进程入队和出队的工作称为**队列管理**。
- 链接的方式有两种：**单向链接**和**双向链接**。
- 无论单向链接还是双向链接，解决入、出队问题，都是首先找到该队列的队首指针，沿链找出要入队的进程以及它要插入的位置，或找出要出队的进程，然后修改本进程指针（入队情况）和相邻进程的有关指针值即可。

1. 单链表--进程入队

- 动态地输入进程队列，如 1，3，4，2
- 入队进程 x（若有 x，则提示已存在）
- 进程入队（放置在队列尾巴）：1，3，4，2，x
- 使用 Micrsoft Visual Studio C++ 6.0 或 CodeBlocks 编程：程序 2_5_inqueue.cpp。完善如下程序代码：

```
#include <malloc.h>
#include <stdio.h>
#include <string.h>
#define NULL 0
typedef struct processpcb
 { int id;          /*进程控制块编号*/
    struct processpcb *next;
 }node;
 int n;
node *creat(void)     /*建立进程控制块队列表*/
 {  填补程序  }
node *append(node *head,node *q)   /*增加一个进程进入队列*/
 {  填补程序  }
void print (node *head)   /*输出链表*/
 {  填补程序  }
void main()
 {  填补程序：模拟建立进程控制块队列和入队过程  }
```

2. 单链表--进程出队

- 动态地输入进程队列，如 1，3，4，2
- 出队方式
  - 队首进程出队
  - 队中进程出队（选定进程）
  - 队尾进程出队
- 根据进程 id 选择一个进程出列（若不存在该进程，则提示不存在）
- 使用 Micrsoft Visual Studio C++ 6.0 或 CodeBlocks 编程：程序 2_6_outqueue.cpp。完善如下程序代码：

```
#include <malloc.h>
#include <stdio.h>
#include <string.h>
#define NULL 0
typedef struct processpcb
 { int id;   /*进程控制块编号*/
    struct processpcb *next;
 }node;
 int n;
node *creat(void)     /*建立进程控制块队列表*/
  { 填补程序 }
node *del(node *head,int pcb) /*根据进程 PCB 的 id 找到需要出列的进程*/
  { 填补程序 }
void print (node *head)   /*输出链表*/
  { 填补程序 }
void main()
  { 填补程序：模拟建立进程控制块队列和出队过程 }
```