队列实验题

【实验目的】

1. 熟练掌握队列的存储结构及其基本操作的实现。
2. 利用队列先进先出的特点解决实际应用问题。
3. 掌握用各种线性结构来解决稍复杂问题。

【实验内容】

1. 题目内容：

编写一个程序，采用链队列，实现队列的下列基本运算：

(1)初始化队列

(2)入队列

(3)出队列

(4)取队头元素

(5)判队列是否为空

(6)遍历队列(将队列的数据元素从队头到队尾依次显示)

输入格式：

三行数据，第一行整数表示要入队的数据元素个数，第二行整数表示要入队的数据元素，两个整数之间以空格分隔，第三行整数表示要出队的元素个数。

输出格式：

三行数据，第一行整数表示目前队列是否为空，如果不为空，第二行表示目前队列的队头元素，第三行表示目前队列的数据元素(依次从队头到队尾)；如果为空，仅显示一行数据。

输入样例： 3 5 4 6 2 输出样例： N 6 6	输入样例： 5 3 2 1 7 9 2 输出样例： N 1 1 7 9	输入样例： 4 3 2 1 7 4 输出样例： Y

2. 题目内容：

编写一个程序，采用循环队列，实现队列的下列基本运算：

(1)初始化队列

(2)入队列

(3) 出队列

(4) 取队头元素

(5) 判队列是否为空

(6) 遍历队列(将队列的数据元素从队头到队尾依次显示)

输入格式：

三行数据，第一行整数表示要入队的数据元素个数，第二行整数表示要入队的数据元素，两个整数之间以空格分隔，第三行整数表示要出队的元素个数。

输出格式：

三行数据，第一行整数表示目前队列是否为空，如果不为空，第二行表示目前队列的队头元素，第三行表示目前队列的数据元素(依次从队头到队尾)；如果为空，仅显示一行数据。

输入样例： 3 5 4 6 2 **输出样例：** N 6 6	**输入样例：** 5 3 2 1 7 9 2 **输出样例：** N 1 1 7 9	**输入样例：** 4 3 2 1 7 4 **输出样例：** Y

3. 题目内容：

设停车场是一个可停放 n 辆汽车的狭长通道，且只有一个大门可供汽车进出。汽车在停车场内按车辆到达时间的先后顺序，依次由北向南排列(大门在最南端，最先到达的一辆车停放在车场的最北端)，若车场内已停满 n 辆汽车，则后来的汽车只能在门外的便道上等候，一旦有车开走，则排在便道上的第一辆车即可开入；当停车场内某辆车要离开时，在它之后进入的车辆必须先退出车场为它让路，待该辆车开出大门外，其他车辆再按原次序进入停车场，每辆停放在停车场的车在它离开停车场时必须按它停留的时间长短交纳费用。试为停车场编制按上述要求进行管理的模拟程序。

【基本要求】

以栈模拟停车场，以队列模拟车场外的便道，按照从终端读入的输入数据序列进行模拟管理。每一组输入数据包括三个数据项：汽车“到达”或“离去”信息、汽车牌照号码以及到达或离去的时刻。对每一组输入数据进行操作后的输出信息为：若是车辆到达，则输出汽车在停车场内或便道上的停车位置；若是车辆离去，则输出汽车在停车场内停留的时间和应交纳的费用(在便道上停留的时间不收费)。栈以顺序结构实现，队列以链表结构实现。

【测试数据】

设 n=2，输入数据为: (‘A’,1,5), (‘A’,2,10), (‘D’,1,15), (‘A’,3,20), (‘A’,4,25), (‘A’,5,30), (‘D’,2,35), (‘D’,4,40), (‘E’,0,0)。其中：‘A’ 表示到达(Arrival); ’D’表示离去(Departure); ‘E’表示输入结束(End)。

【实验提示】

需另设一个栈，临时停放为给要离去的汽车让路而从停车场退出来的汽车，也用顺序存储结构实现。输入数据按到达或离去的时刻有序。栈中每个元素表示一辆汽车，包含两个数据：汽车的牌照号码和进入停车场的时刻。

【实验参考】

```
int main()
{……
   InitStack(Sstation); /*初始化停车场*/
   InitStack(Stemp); /*初始化让路的临时栈*/
   InitQueue(Wait); /*初始化便道*/
   while(1)
   {
     printf("\n   A. The Car Arrive   D. The Car Leave    E. Exit System\n");
    scanf("%d",&ch);
    switch(ch)
    {
      case ‘A’: Arrival();   break; /*车辆到达*/
      case ‘D’: Leave();    break; /*车辆离开*/
      case ‘E’: return 0;            /*退出主程序*/
      default: break;
    }
  }
return 0;
}
/*-------------------------------------------------*/
Arrival()    //车辆到达
{   if   停车场未满       //判断栈是否满
    {
        将汽车 A 放在停车场;   //入栈
        输出汽车 A 在停车场的停车位置;
      }
    else
    {
        将汽车 A 放在便道;      //入队列
        输出汽车 A 在便道的停车位置;
      }
}
  /*-------------------------------------------------*/
  Leave()/*车辆离开
{ if   汽车 A 停在停车场内       //判断栈是否存在某一元素
```

```
    {
            将比汽车 A 后进停车场的所有汽车先出停车场,暂时存放在临时栈 Stemp;
        汽车 A 出停车场,并计算出汽车 A 在停车场的停留时间;
        临时栈 Stemp 中的所有汽车在按原次序回到停车场;
        将便道上的最前面的一辆汽车入停车场;
    }
    else
    {
        输入出错;
    }
}
  /*-----------------------------------------------*/
```

测试格式参考如下：