# 温州大学计算机与人工智能学院

## 程序设计课程设计　实验报告

| 实验名称 | 银行排队之四：按客户类型入队和出队 | | | | |
|---|---|---|---|---|---|
| 班　　级 | 23大数据1 | 姓　　名 | 徐王晶 | 学　　号 | 23211870102 |
| 实验地点 | 南5B105 | 实验时间 | 2023-12-26,19:37:10 | 指导老师 | 虞铭财 |

### 一、问题编号：

1168

地址：http://10.132.254.54/problem/1168/

### 二、问题描述：

注意：用链表完成
现在银行都有排队叫号系统，如果你到银行去办理业务，首先取得一个顺序号，并告诉你前面有多少人在等待。
现在请你设计、并实现该系统。

客户信息包括：取号的号码、客户银行卡号码、客户类型（包括普通客户和vip客户，分别用Ordinary和VIP表示）。
VIP用户有优先权，即VIP入队时，可以插入到　所有Ordinary类型用户的前面，已经排队的最后一个VIP用户的后面。

本题要求实现以下功能：
（1）客户进入银行并取号，该功能对应的命令为IN number type，number表示客户银行卡号码，type表示客户类型。
（2）在上一题基础上，按取号顺序列出当前未办理业务的客户信息，该功能对应的命令为LIST。
（3）客户办理完业务并离开银行，该功能对应的命令为OUT。
（4）退出系统，该功能对应的命令为QUIT。

### 三、输入说明：

输入有多行，每行表示一种操作。

本题实现的操作有：
IN guestnumber guesttype
其中IN 表示入队，guestnumber表示客户银行卡号码，guesttype有两种取值，分别为Ordinary和VIP，表示普通客户与VIP客户。

LIST
按取号的顺序输出队列中所有排队的客户信息。

QUIT
退出银行排队系统

请注意：输入的数据中，只有最后一个命令是QUIT命令。

## 四、输出说明：

输出有多行，对不同的命令给出不同的输出。

以下是具体的输出说明：
IN 客户银行卡号码  客户类型
该命令首先在单独的一行中输出"IN:"，然后再输出客户信息。
客户信息包括客户编号 客户银行卡号码  客户类型 该客户前面的客户数。客户编号由系统在客户取号的时候给定，从1开始顺序编号。
客户类型分为Ordinary和VIP。数据之间用一个空格分开。

请注意：
第一个客户入队后，则该用户的顺序号为1，前面有0个客户等待；
第二个客户入队后，则该用户的顺序号为2，前面有1个客户等待。
以此类推。

LIST
该命令首先在单独的一行中输出"LIST:"，后面有若干行先按客户类型（VIP优先），再按客户取号的顺序输出，每一行输出一个客户的信息，每行的输出格式为
客户编号 客户银行卡号码 客户类型

OUT
该命令首先在单独的一行中输出"OUT:"，接下来一行输出办理业务的客户信息，包括顺序号，客户银行卡号码和客户类型。
如果没有办理业务的客户，则输出"FAILED:"。

QUIT
在单独的一行中显示"GOOD BYE!"后结束程序。

## 五、输入样列：

```
IN 1000001 Ordinary
IN 2000003 VIP
IN 2000009 VIP
OUT
OUT
OUT
OUT
IN 1000007 Ordinary
IN 2000005 VIP
LIST
OUT
QUIT
```

## 六、输出样列：

```
IN:1 1000001 Ordinary 0
IN:2 2000003 VIP 0
IN:3 2000009 VIP 1
OUT:2 2000003 VIP
OUT:3 2000009 VIP
OUT:1 1000001 Ordinary
FAILED:
IN:4 1000007 Ordinary 0
IN:5 2000005 VIP 0
LIST:
5 2000005 VIP
4 1000007 Ordinary
OUT:5 2000005 VIP
GOOD BYE!
```

## 七、解答内容：

**所用语言：**

**源代码：**

```c
#include <stdio.h>
#include <string.h>
#include <stdlib.h>

struct Data
{
    char id[20];
    char cla[20];
    int num;
    struct Data *next;
    struct Data *prev;
};

struct Queue
{
    struct Data *ohead;
    struct Data *otail;
    struct Data *vhead;
    struct Data *vtail;
    int ocount;
    int vcount;
    int onum;
};

void In(struct Queue *queue, int num);
void List(struct Queue *queue);
void Out(struct Queue *queue);

int main(void)
{
    struct Queue queue;
    queue.ohead = NULL;
    queue.otail = NULL;
    queue.vhead = NULL;
    queue.vtail = NULL;
    queue.ocount = 0;
    queue.vcount = 0;
    queue.onum = 0;

    int num = 0;
    char op[20];
    while (scanf("%s", op), strcmp(op, "QUIT") != 0)
    {
        if (strcmp(op, "IN") == 0)
        {
            num++;
            In(&queue, num);
        }
        else if (strcmp(op, "LIST") == 0)
        {
            List(&queue);
        }
        else if (strcmp(op, "OUT") == 0)
        {
            Out(&queue);
        }
    }
    printf("GOOD BYE!\n");
    return 0;
}
```

```
062. void In(struct Queue *queue, int num)
063. {
064.     struct Data *data = (struct Data *)malloc(sizeof(struct Data));
065.     scanf(
066.         "%s %s",
067.         data -> id,
068.         data -> cla
069.     );
070.     data -> num = num;
071. 
072.     if (strcmp(data -> cla, "VIP") == 0)
073.     {
074.         if (queue -> vhead == NULL)
075.         {
076.             queue -> vhead = data;
077.             queue -> vtail = data;
078.             data -> prev = NULL;
079.             data -> next = NULL;
080.         }
081.         else
082.         {
083.             data -> prev = queue -> vtail;
084.             queue -> vtail -> next = data;
085.             queue -> vtail = data;
086.             data -> next = NULL;
087.         }
088. 
089.         printf(
090.             "IN:%d %s %s %d\n",
091.             data -> num,
092.             data -> id,
093.             data -> cla,
094.             data -> num - 1 - (queue -> vcount) - (queue -> onum)
095.         );
096.     }
097.     else
098.     {
099.         (queue -> onum) += 1;
100.         if (queue -> ohead == NULL)
101.         {
102.             queue -> ohead = data;
103.             queue -> otail = data;
104.             data -> prev = NULL;
105.             data -> next = NULL;
106.         }
107.         else
108.         {
109.             data -> prev = queue -> otail;
110.             queue -> otail -> next = data;
111.             queue -> otail = data;
112.             data -> next = NULL;
113.         }
114. 
115.         printf(
116.             "IN:%d %s %s %d\n",
117.             data -> num,
118.             data -> id,
119.             data -> cla,
120.             data -> num - 1 - (queue -> vcount) - (queue -> ocount)
121.         );
122.     }
123. }
124. 
125. void Out(struct Queue *queue)
126. {
127.     if (queue -> vhead == NULL && queue -> ohead == NULL)
128.     {
129.         printf("FAILED:\n");
130.     }
131.     else
132.     {
133.         if (queue -> vhead != NULL)
134.         {
135.             printf(
136.                 "OUT:%d %s %s\n",
137.                 queue -> vhead -> num,
138.                 queue -> vhead -> id,
139.                 queue -> vhead -> cla
140.             );
141.             struct Data *data = queue -> vhead;
142.             queue -> vhead = queue -> vhead -> next;
143.             free(data);
144.             (queue -> vcount) += 1;
145.         }
146.         else
147.         {
148.             printf(
149.                 "OUT:%d %s %s\n",
150.                 queue -> ohead -> num,
151.                 queue -> ohead -> id,
152.                 queue -> ohead -> cla
```

```
            );
            struct Data *data = queue -> ohead;
            queue -> ohead = queue -> ohead -> next;
            free(data);
            (queue -> ocount) += 1;
        }
    }
}

void List(struct Queue *queue)
{
    printf("LIST:\n");
    struct Data *data = queue -> vhead;
    while (data != NULL)
    {
        printf(
            "%d %s %s\n",
            data -> num,
            data -> id,
            data -> cla
        );
        data = data -> next;
    }
    data = queue -> ohead;
    while (data != NULL)
    {
        printf(
            "%d %s %s\n",
            data -> num,
            data -> id,
            data -> cla
        );
        data = data -> next;
    }
}
```

## 八、判题结果

**AC - 答案正确**

**判题结果补充说明：**

test id:2031,result:AC, usedtime:0MS, usedmem:864KB,score:100