**数据结构实验报告**

**实验三 跳舞舞伴问题**

学 生 姓 名 ：杨仕振

学 号 ：2015301500349

二○一六年十月

1. **简介**

假设在周末舞会上，男士们和女士们进入舞厅时，各自排成一队。跳舞开始时，依次从男队和女队的队头各出一人配成舞伴。若两队初始人数不相同，则较长的 那一队中未配对者等待着下一轮舞曲。现要求写一算法模拟上述舞伴配对问题。

提示：1.设置两个队列分别存放男士和女士入队者。

2.假设男士和女士的记录存放在一个数组中作为输入，然后依次扫描该数组的各元素，并根据性别来决定是进入男队还是女队。

3.跳舞者个人信息可用结构体表示，包括舞者姓名（字符数组）和性别（字符）。

**二、算法说明**

定义总的人数，配对的轮数，总队与男女队等变量，然后先初始化男女队，将每个人的姓名和性别输入到总队里，然后再根据性别将其分别入男队和女队，自定义匹配的次数，然后开始匹配。

先确定两队的人数，然后进行循环匹配，配对即：将两队各出队一人，打印配对结果，然后再将其入队。直到将其中较短的队伍遍历一次为止（等长则默认为女队），然后根据两队的人数，将本轮配对中未配对的人数和下一个等待配对的人显示出来。如此匹配下去，直到达到指定的配对轮数为止。

进队函数bool enQueue(CirQueue \*&q, person\_type p);

出队函数bool deQueue(CirQueue \*&q, person\_type &p);

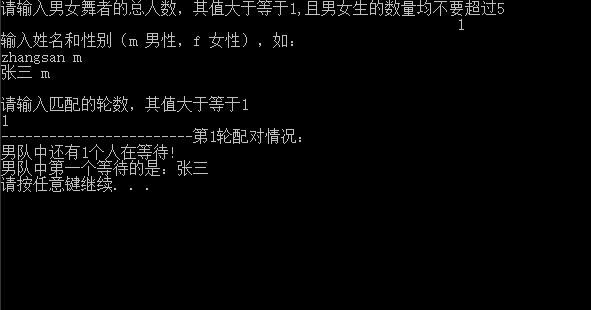
求首个队员的函数person\_type queue\_front(CirQueue \*&q)

上述三个函数采用了环形队列方式，并且没有在男女队结构中设置rear指针，而是通过front指针和队中元素个数count来进行出队和进队操作。

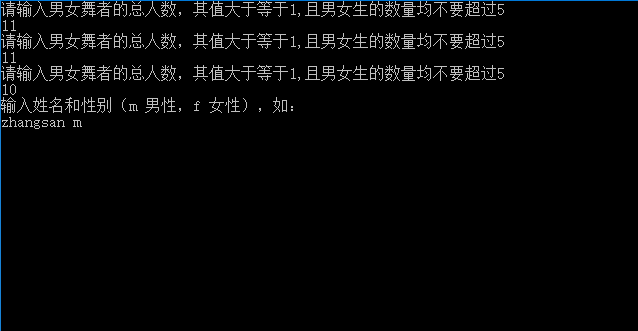
上述进、出队函数以及初始化函数void InitQueue(CirQueue \*&q);都借鉴了教材P85的例题中的算法。

**三、测试结果**

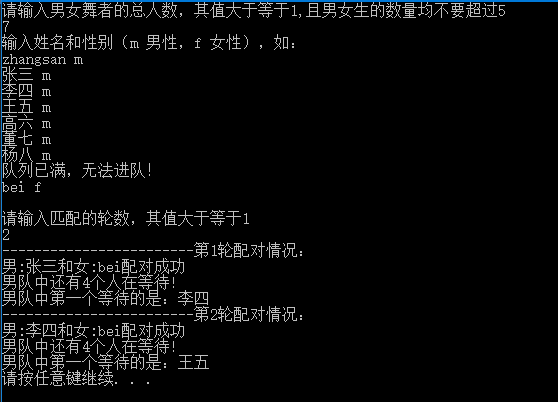
当其中一支队伍为空时：



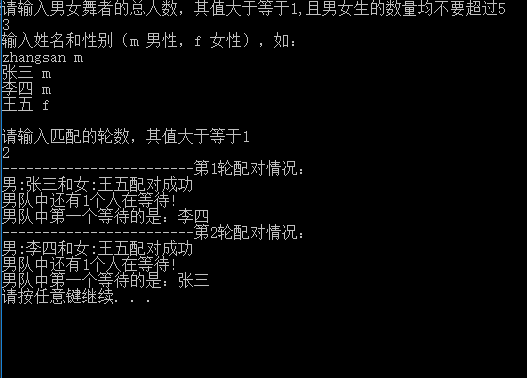
当其中总的人数超过上限时（此处限定总人数不得超过10）：



当其中一支队伍人数超过上限时而总人数未超过上限时：



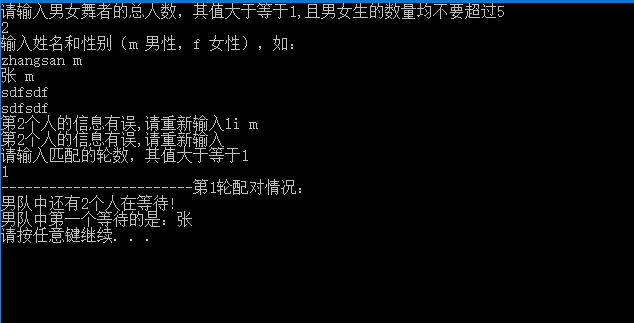
当一切正常时：



**四、分析与探讨**

该程序的时间复杂度取决于dancepartners函数中两重for循环的复杂度，为O(n2)。该程序的空间复杂度为O(1)。

总的来说，程序基本实现了所需要的功能，但是在输入个体信息的时候，如果不严格按照“姓名 性别”的方式输入的话，不能识别并拒绝数值的录入，这一点还有待修正。（如下图）



**附录：源代码**

#include<stdio.h>

#include<stdlib.h>

#include<malloc.h>

#define Queuesize 5

typedef struct{

char name[20];

char sex;

}person\_type; //定义一个舞者

typedef struct{

person\_type queue[Queuesize];

int front;

int count;

}CirQueue; //定义不同性别的队

void InitQueue(CirQueue \*&q)//初始化男女队列

{

q = (CirQueue\*)malloc(sizeof(CirQueue));

if (!q) {

printf("创建失败，程序终止！");

exit(-1);

} //如果内存申请失败，则退出程序。

q->front = q->count = 0;

}

bool enQueue(CirQueue \*&q, person\_type p) //进队

{

int rear;//临时的rear指针

if (q->count == Queuesize) {

printf("队列已满，无法进队！"); //若队列已满，则无法进队

return false;

}

else {

rear = (q->front + q->count) % Queuesize; //确定临时队尾的位置

rear = (rear + 1) % Queuesize;

q->queue[rear]=p;

q->count++;

return true;

}

}

bool deQueue(CirQueue \*&q, person\_type &p) //出队

{

if (q->count == 0) {

printf("队列已空，无法出队！");//若队中没有元素，则不可出队

return false;

}

else {

q->front = (q->front + 1) % Queuesize;//修改出队后队头的位置

p = q->queue[q->front];

q->count--;

return true;

}

}

person\_type queue\_front(CirQueue \*&q) //求队中的第一个人

{

if (q->count == 0) {

printf("队列为空，操作失败");

exit(-1);

} //队空则退出程序

return q->queue[(q->front+1)%Queuesize];

}

void dancepartners(CirQueue \*&female, CirQueue \*&male, int round)

{

person\_type p;

person\_type q;

int m = male->count;//求男队人数

int n = female->count;//求女队人数

for (int j = 1; j <= round; j++)

{

printf("------------------------");

printf("第%d轮配对情况：\n", j);

for (int i = 1; i <= (n<m ? n : m); i++)

{

deQueue(male, p);

deQueue(female, q);

printf("男:%s和女:%s配对成功\n", p.name, q.name);

enQueue(male, p);

enQueue(female, q);

}

if (m>n) {

printf("男队中还有%d个人在等待!\n", m - n);

printf("男队中第一个等待的是：%s\n", queue\_front(male).name);

}

else if (m<n) {

printf("女队中还有%d个人在等待!\n", n - m);

printf("女队中第一个等待的是：%s\n", queue\_front(female).name);

}

else {

printf("没有人剩余!\n");

}

}

}

int main()

{

int round;//定义配对的轮数

int i, num;//定义计数变量和总的人数

CirQueue \*male, \*female;//定义男队

person\_type dancers[2\*Queuesize];//定义包含所有人的队列

InitQueue(male);

InitQueue(female);//初始化两个队列

do {

printf("请输入男女舞者的总人数，其值大于等于1,且男女生的数量均不要超过%d\n",Queuesize);

scanf\_s("%d", &num);

}while (num<1 || num>2\*Queuesize);

printf("输入姓名和性别（m 男性，f 女性），如：\nzhangsan m\n");

for (i = 0; i<num; i++)

{

scanf\_s("%s %c",&dancers[i].name,20,&dancers[i].sex);

if (dancers[i].sex != 'f'&&dancers[i].sex != 'm')

{

printf("第%d个人的信息有误,请重新输入", (i--) + 1);

continue;

}

switch (dancers[i].sex)

{

case 'm':enQueue(male,dancers[i]); break;

case 'f':enQueue(female,dancers[i]);break;//通过选择，将不同的人进入不同的队

}

}

do{

printf("\n请输入匹配的轮数，其值大于等于1\n");

scanf\_s("%d", &round);

}while (round<1);

dancepartners(female, male, round);

system("PAUSE");

return 0;

}