# 实验二

## 1. 问题定义及需求分析

### 1.1 问题描述

##### 设有M个男羽毛球运动员和N个女羽毛球运动员，现进行男女混合双打组合K轮配对。男女运动员分别编号排队在等候队列，按顺序依次从男女运动员中各出队1人组合配对。本轮没成功配对者等待下一轮次配对。

### 1.2 实验要求

##### 设计程序模拟完成运动员组合配对过程。

###### （1）采用队列等数据结构。

###### （2）输出每轮的配对信息。

### 1.3 输入

##### 男女运动员人数，配对的总轮数，每名运动员的名字。

### 1.4 输出形式

##### 每一轮比赛的场次，每场比赛的男女运动员共4人的名字。

### 1.5 程序的功能

##### 输出配对情况。

### 1.6 测试数据

#### 1.6.1 输入

##### 5 4 3

##### Abe Bruce Charlie David Eric

##### Amy Betty Claire Dory

#### 1.6.2 输出

##### 第1轮配对:

###### 第1场比赛:

Team 1:男运动员: Abe|女运动员: Amy  
Team 2:男运动员: Bruce|女运动员: Betty

###### 第2场比赛:

Team 1:男运动员: Charlie|女运动员: Claire  
Team 2:男运动员: David|女运动员: Dory

###### 第3场比赛:

Team 1:男运动员: Eric|女运动员: Amy  
Team 2:男运动员: Abe|女运动员: Betty

##### 第2轮配对:

###### 第1场比赛:

Team 1:男运动员: Bruce|女运动员: Claire  
Team 2:男运动员: Charlie|女运动员: Dory

###### 第2场比赛:

Team 1:男运动员: David|女运动员: Amy  
Team 2:男运动员: Eric|女运动员: Betty

##### 第3轮配对:

###### 第1场比赛:

Team 1:男运动员: Abe|女运动员: Claire  
Team 2:男运动员: Bruce|女运动员: Dory

###### 第2场比赛:

Team 1:男运动员: Charlie|女运动员: Amy  
Team 2:男运动员: David|女运动员: Betty

###### 第3场比赛:

Team 1:男运动员: Eric|女运动员: Claire  
Team 2:男运动员: Abe|女运动员: Dory

## 2.概要设计

### 2.1 抽象数据类型定义

##### ADT Queue

##### {

##### char Data[max\_element\_count][max\_str\_length];//二维数组存储每运动员的名字

##### int front,end;//队列头元素的下标，队列尾元素的下标

##### }Queue;

##### void init(Queue\*);//初始化一个空队列

##### int push(Queue,char);//压元素入队列

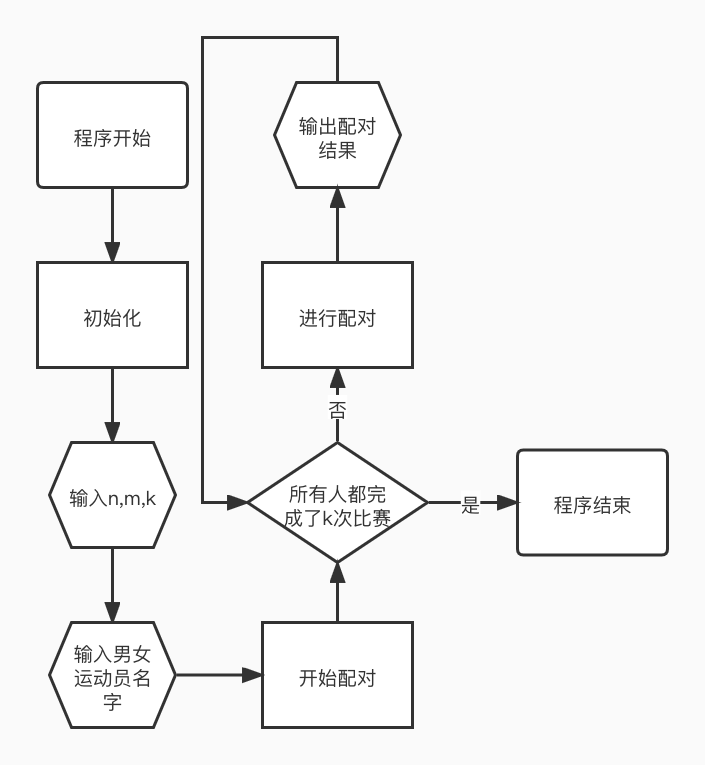
##### int pop(Queue,char);//弹元素出队列

##### int empty(Queue\*);//清空队列

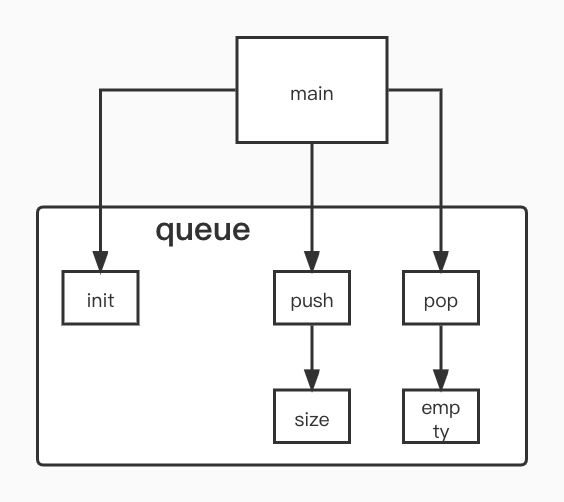
##### int size(Queue\*);//求队列的人数

##### }

### 2.2 主程序流程



### 2.3 模块调用关系



## 3.详细设计

### 3.1 定义数据类型及存储结构

typedef struct  
{  
 char Data[max\_element\_count][max\_str\_length];//二维数组存储每运动员的名字  
 int front,end;//队列头元素的下标，队列尾元素的下标  
}Queue;  
void init(Queue\*);//初始化一个空队列  
int push(Queue\*,char\*);//压元素入队列  
int pop(Queue\*,char\*);//弹元素出队列  
int empty(Queue\*);//清空队列  
int size(Queue\*);//求队列的人数

### 3.2 每个函数及操作的代码

##### 初始化一个空队列

void init(Queue\* Q)  
{  
 Q->front=1;  
 Q->end=0;  
 //printf("initializing queue...\n");  
}

##### 压元素入队列

int push(Queue\* Q,char\* str)  
{  
 if(Q->end+1>=max\_element\_count)//如果队列已满，则返回ERROR  
 return printf("\nERROR: cannot push because queue is full!\n"),ERROR;  
 ++Q->end;  
 for(int i=0;i<=strlen(str);++i)//将名字字符串压入队列  
 Q->Data[Q->end][i]=str[i];  
 return /\*printf("%s:push success.\n",Q->Data[Q->end]),\*/OK;  
}

##### 弹元素出队列

int pop(Queue\* Q,char\* str)  
{//如果队列为空，返回ERROR  
 if(empty(Q))return printf("ERROR: cannot pop because queue is empty!\n"),ERROR;  
 for(int i=0;i<=strlen(Q->Data[Q->front]);++i)//将名字字符串弹出队列  
 str[i]=Q->Data[Q->front][i];  
 ++Q->front;  
 return /\*printf("%s,pop success.\n",str),\*/OK;  
}

##### 清空队列

int empty(Queue\* Q)  
{  
 if(Q->front>Q->end)return /\*printf("queue is empty.\n"),\*/1;//判断队列是否为空  
 return /\*printf("queue is not empty.\n"),\*/0;  
}

##### 求队列元素个数

int size(Queue\* Q)  
{  
 return Q->end-Q->front+1;  
}

## 调试分析

### 4.1 遇到的问题及分析

#### 4.1.1 问题

##### 一开始理解错了题意，误以为要求输出男女运动一一配对后的结果即可，错误的原因是没有考虑循环的轮数，每一轮循环进行的比赛有多场；

##### 在读懂题目，摸清过程之后，又遇到每一轮中的比赛场数不定的情况，原因是输入的轮数k的奇偶性会影响结束条件。

#### 4.1.2 分析

##### 设置第i轮的终止条件为：男队的运动员或者女队的运动员中的每个人都至少比赛了i次。

### 4.2 算法时空分析

#### 4.2.1 时间

#### 4.2.2 空间

#### 4.2.3 改进

### 4.3 经验和体会

##### 在动手实现一个模拟现实生活的例子的时候，要先去了解相关事例，然后抽离出模型来，在计算机上进行模拟。

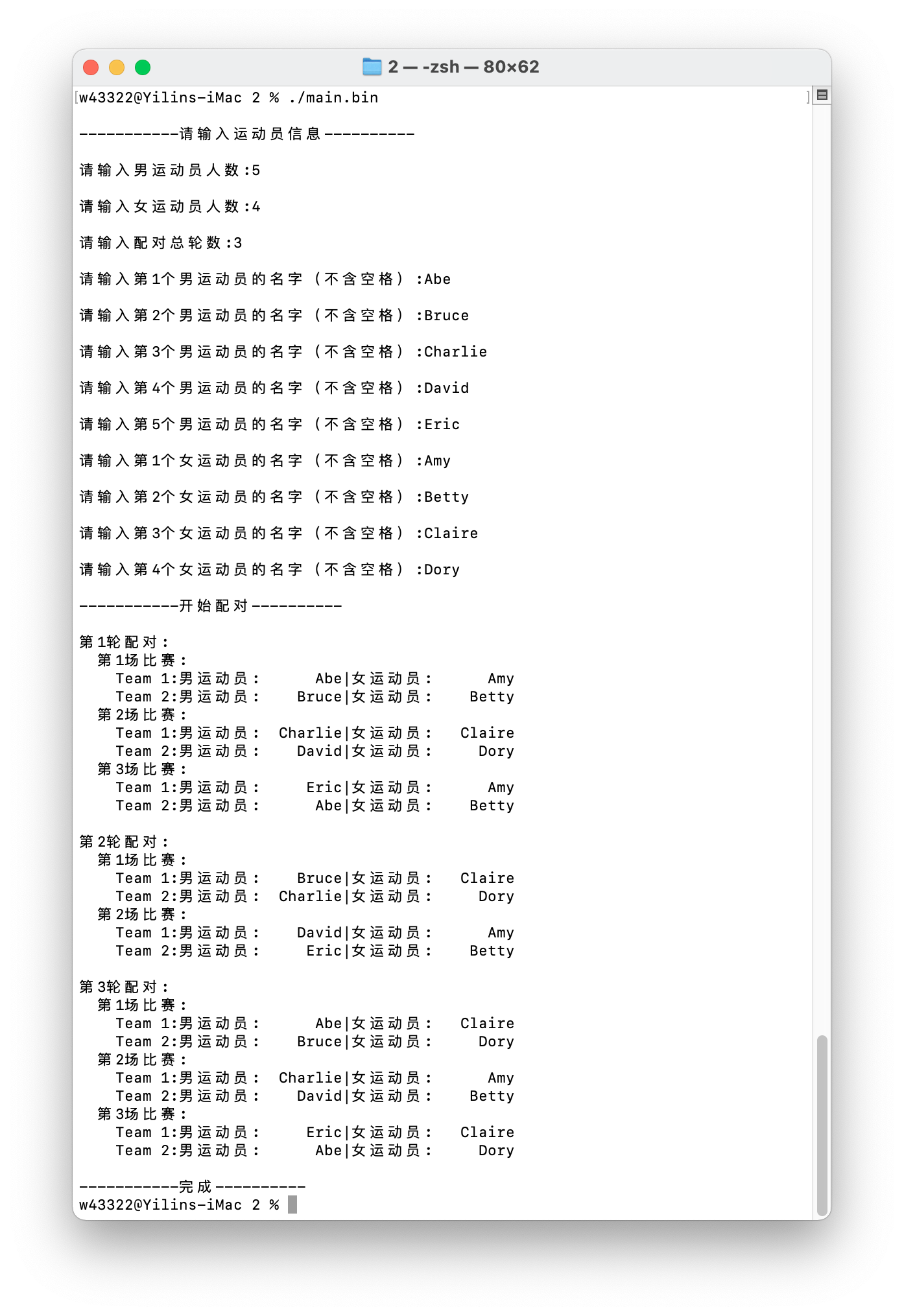
## 5.使用说明

##### (1)输入男女两队的人数和比赛的轮数

##### (2)按照提示输入男女运动员的名字

##### (3)程序根据以上输入的数据可自动输出匹配完成的信息

## 6.测试结果



## 7.附录

### 7.1 个人负责的部分

##### 主函数。

### 7.2 整个程序

#### 7.2.0 Makefile

main : main.c queue.c  
 gcc -Wall -O2 -o main.bin main.c queue.c

#### 7.2.1 universal\_header.h

#ifndef UNIVERSAL\_H  
#define UNIVERSAL\_H  
  
#include<stdio.h>  
#include<stdlib.h>  
#include<string.h>  
#define max\_element\_count 100  
#define max\_str\_length 10  
#define ERROR 1  
#define OK 0  
  
#define max(a,b) (a>b?a:b)  
  
#endif //UNIVERSAL\_H

#### 7.2.2.1 main.h

#ifndef MAIN\_H  
#define MAIN\_H  
  
#include"universal\_header.h"  
#include"queue.h"  
  
#endif //MAIN\_H

#### 7.2.2.2 main.c

#include"main.h"  
int main()  
{  
 int m,n,k;  
 Queue M,N;  
 init(&M);  
 init(&N);  
 printf("\n-----------请输入运动员信息----------\n");  
 printf("\n请输入男运动员人数:");  
 scanf("%d",&n);  
 printf("\n请输入女运动员人数:");  
 scanf("%d",&m);  
 printf("\n请输入配对总轮数:");  
 scanf("%d",&k);  
 for(int i=1;i<=n;++i)  
 {  
 printf("\n请输入第%d个男运动员的名字（不含空格）:",i);  
 char name[max\_str\_length];  
 scanf("%s",name);  
 push(&M,name);  
 }  
 for(int i=1;i<=m;++i)  
 {  
 printf("\n请输入第%d个女运动员的名字（不含空格）:",i);  
 char name[max\_str\_length];  
 scanf("%s",name);  
 push(&N,name);  
 }  
 printf("\n-----------开始配对----------\n");  
 int max\_cnt=max(size(&M),size(&N));//男运动员多还是女运动员多？这个变量存最多的数量。  
 int tot=0;//每个运动员都要至少比赛K次。  
 for(int i=1;i<=k;++i)  
 {  
 printf("\n第%d轮配对:\n",i);  
 for(int j=1;tot<max\_cnt\*i;++j)//进入循环的条件：这一轮配对中，还有人没比赛i次。  
 {  
 char name\_m1[max\_str\_length],name\_n1[max\_str\_length];  
 pop(&M,name\_m1);push(&M,name\_m1);  
 pop(&N,name\_n1);push(&N,name\_n1);  
 char name\_m2[max\_str\_length],name\_n2[max\_str\_length];  
 pop(&M,name\_m2);push(&M,name\_m2);  
 pop(&N,name\_n2);push(&N,name\_n2);  
  
 printf(" 第%d场比赛:\n",j);  
 printf(" Team 1:男运动员:%9s|女运动员:%9s\n",name\_m1,name\_n1);  
 printf(" Team 2:男运动员:%9s|女运动员:%9s\n",name\_m2,name\_n2);  
 tot+=2;  
 }  
 }  
 printf("\n-----------完成----------\n");  
 return 0;  
}

#### 7.2.3.1 queue.h

#ifndef QUEUE\_H  
#define QUEUE\_H  
  
#include"universal\_header.h"  
typedef struct  
{  
 char Data[max\_element\_count][max\_str\_length];//二维数组存储每运动员的名字  
 int front,end;//队列头元素的下标，队列尾元素的下标  
}Queue;  
void init(Queue\*);//初始化一个空队列  
int push(Queue\*,char\*);//压元素入队列  
int pop(Queue\*,char\*);//弹元素出队列  
int empty(Queue\*);//清空队列  
int size(Queue\*);//求队列的人数  
  
#endif //QUEUE\_H

#### 7.2.3.2 queue.c

#include"queue.h"  
void init(Queue\* Q)  
{  
 Q->front=1;  
 Q->end=0;  
 //printf("initializing queue...\n");  
}  
int push(Queue\* Q,char\* str)  
{  
 if(Q->end+1>=max\_element\_count)//如果队列已满，则返回ERROR  
 return printf("\nERROR: cannot push because queue is full!\n"),ERROR;  
 ++Q->end;  
 for(int i=0;i<=strlen(str);++i)//将名字字符串压入队列  
 Q->Data[Q->end][i]=str[i];  
 return /\*printf("%s:push success.\n",Q->Data[Q->end]),\*/OK;  
}  
int pop(Queue\* Q,char\* str)  
{//如果队列为空，返回ERROR  
 if(empty(Q))return printf("ERROR: cannot pop because queue is empty!\n"),ERROR;  
 for(int i=0;i<=strlen(Q->Data[Q->front]);++i)//将名字字符串弹出队列  
 str[i]=Q->Data[Q->front][i];  
 ++Q->front;  
 return /\*printf("%s,pop success.\n",str),\*/OK;  
}  
int empty(Queue\* Q)  
{  
 if(Q->front>Q->end)return /\*printf("queue is empty.\n"),\*/1;//判断队列是否为空  
 return /\*printf("queue is not empty.\n"),\*/0;  
}  
int size(Queue\* Q)  
{  
 return Q->end-Q->front+1;  
}