通信与信息工程学院

《数据结构》实验报告

|  |  |
| --- | --- |
| 班 级： | 电信1702 |
| 姓 名： | 秦伟钦 |
| 学 号： | 17407110815 |
| 指导教师： | 刘晓佩 |
| 实验时间： |  |

通信与信息工程学院

二〇一九年

**目录**

实验4 队列的各种基本运算及应用(2)-------------------------------------------------------------------------------------------页码1

**完成所有实验报告后，按照上述格式形成目录。**

实验四 队列的各种基本运算及应用

一、实验目的

1. 领会队列的循环存储结构和链式存储结构；

2. 掌握循环队列和链队列中各种基本运算算法设计。

二、实验环境

Visual C++ 6.0

三、实验内容

**问题描述**：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下一轮舞曲。

**要求写程序模拟舞伴配对问题**。具体功能如下：

初始化男队、女队；

依次将跳舞者按照性别分别入男队和女队。

判断队列是否为空；

输出配对的男女舞者信息；

若两队初始人数不相同，则较长的那一队中未配对者等待下一轮舞曲。

四、设计过程

#include<stdio.h>

#define MAXQSIZE 100

#define OK 1

#define ERROR 0

#define OVERFLOW -1

#include <cstdlib>

#include<iostream>

using namespace std;

typedef char QElemType;

typedef int Status;

typedef struct

{

char name[20];

char sex;

}person;

typedef struct

{

person\* dancer;

person\* base;

int front;

int rear;

}SqQueue;

Status InitQueue(SqQueue& Q)

{//构造一个空队列Q

Q.base = new person[MAXQSIZE];

if (!Q.base) exit(OVERFLOW);

Q.front = Q.rear = 0;

return OK;}

Status EnQueue(SqQueue& Q, person e)

{//插入元素e为Q的新的队尾元素

if ((Q.rear + 1) % MAXQSIZE == Q.front)

return ERROR;

Q.base[Q.rear] = e;

Q.rear = (Q.rear + 1) % MAXQSIZE;

return OK;

int QueueEmpty(SqQueue& Q)

{//判断队列是否为空 if (Q.front == Q.rear) return OK; else

return ERROR;

Status DeQueue(SqQueue& Q, person& e)

{//删除Q的队头元素，用e返回其值

if (Q.front == Q.rear) return ERROR;

e = Q.base[Q.front];

Q.front = (Q.front + 1) % MAXQSIZE;

return OK;

}

person GetHead(SqQueue Q)

{//返回Q的队列元素，不修改队头指针

if (Q.front != Q.rear)

return Q.base[Q.front];

}

void DancePartner(person dancer[], int num)

{//结构数组dancer中存放跳舞的男女，num是跳舞的人数

person p;

int i;

SqQueue Mdancers, Fdancers;

InitQueue(Mdancers);

InitQueue(Fdancers);

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

{

p = dancer[i];

if (p.sex == 'F') EnQueue(Fdancers, p);

else EnQueue(Mdancers, p);

}

cout << "The dancing partner are:\n";

while (!QueueEmpty(Fdancers) && !QueueEmpty(Mdancers))

{//依次输出男女舞伴的姓名

DeQueue(Fdancers, p);

cout << p.name << " ";

DeQueue(Mdancers, p);

cout << p.name << endl;

}

if (!QueueEmpty(Fdancers))

{

p = GetHead(Fdancers);

cout << "下次出队的先生是" << p.name << endl;

}

else if (!QueueEmpty(Mdancers))

{

p = GetHead(Mdancers);

cout << "下次出队的小姐姐是." << p.name << endl; }

int main()

{

int i, j

person dancer[20];

cout << "请输入跳舞的人数：";

cin >> j;

while (j <= 0)

{

cout << "输入错误，请重新输入跳舞的人数：";

cin >> j;

}

for (i = 1; i <= j; i++)

{

cout << "请输入第" << i << "舞者的名字:" << endl;

cin >> dancer[i - 1].name;

cout << "请输入第" << i << "个人的性别(F/M):" << endl;

cin >> dancer[i - 1].sex;

while (dancer[i - 1].sex != 'F' && dancer[i - 1].sex != 'M')

{

cout << "\*\*\*\*\*\*\*输入错误，请重新输入：\n";

cout << dancer[i - 1].sex;

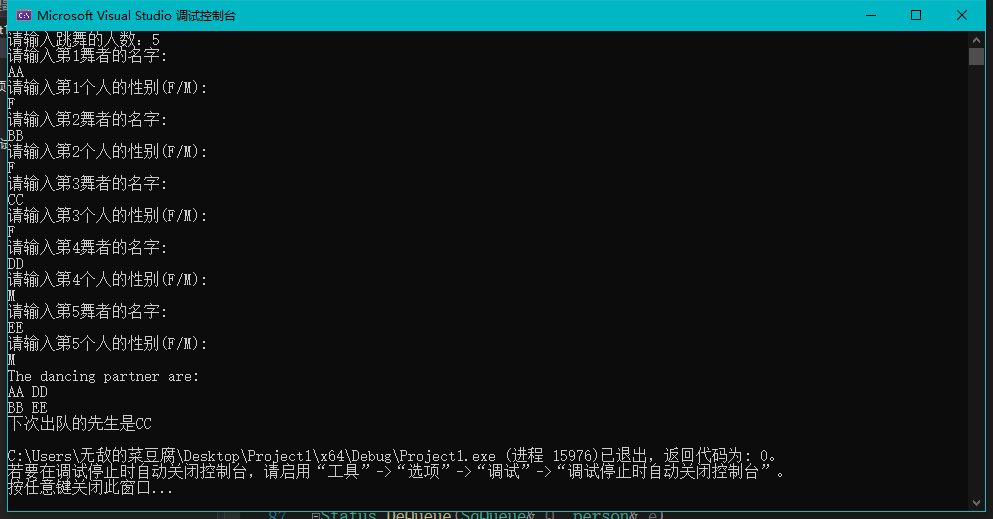
cout << "请输入第" << i << "个人的性别(F/M):" << endl;

cin >> dancer[i - 1].sex;

break; }

DancePartner(dancer, j);

}



**五、总结和体会：**

队列顺序特点是先进先出，所以我们也可以通过双链表的方式来实现队列内容，队列的优势很明显进行顺序组合时能很好的简化循环在算法的时间复杂度上仅有循环算法的低次方，且链队列对内存空间利用率高，需要多少开辟多少不会造成浪费，但是标准输入时可能造成一定的影响，修改标准输入或者输入流可以缓解内存警告。并且链队采用结构化设计，修改或增添函数功能十分方便，利于后期的维护开发。