**湖 州 学 院**

**学生实验报告**

课程名称： 数据结构

实验项目： 实验二 栈和队列

**湖州学院教务处印制**

**实验二 栈和队列**

**一、 实验目的**

掌握栈和队列两种数据结构的定义、存储方式、基本操作的实现以及具体应用。

**二、实验内容**

**实验2.1 括号匹配问题**

1.问题描述：给定一串字符（以’#’结束，不超过100个字符），可能包括三种括号：小括号、中括号和大括号，各种括号之间允许任意的嵌套，编程检查这一串字符中的( ) ,[ ],{ }是否匹配。

输入格式:

输入一字符串（以’#’结束，不超过100个字符），可能包括三种括号：小括号、中括号和大括号，各种括号之间允许任意的嵌套。

输出格式:

如果括号配对，输出配对成功信息，否则输出配对不成功信息。

2.测试数据

(1) ([]{})

(2) {[}]

(3) {{}

(4) ())

说明：可参考课本74-75页 案例3.2。

**实验2.2** **舞伴问题**

1.问题描述：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下一轮舞曲。现要求设计一算法模拟上述舞伴配对问题。

2.测试数据

让用户从键盘输入若干名进入舞厅的人的信息。每个人的信息包含两个部分：姓名和性别。

说明：可参考课本78-79页 案例3.3。

**三、两个实验的源程序清单及实例的运行结果（源程序中应该附有必要的注释）。**

1.源程序清单：可手写，也可附打印稿（A4纸打印）。

括号匹配问题

#include<iostream>

#define OK 1

#define ERROR 0

using namespace std;

typedef char SElemType;

typedef int Status;

#define MAXSIZE 100

typedef struct //栈的顺序存储结构

{

SElemType\* base;

SElemType\* top;

int stacksize;

}SqStack;

Status InitStack(SqStack& S)//构造一个空栈

{

S.base = new SElemType[MAXSIZE];

if (!S.base)

return ERROR;

S.top = S.base;

S.stacksize = MAXSIZE;

return OK;

}

Status Push(SqStack& S, SElemType e) //入栈

{

if (S.top - S.base == S.stacksize)

return ERROR;

\*S.top++ = e;

return OK;

}

Status Pop(SqStack& S, SElemType& e) //出栈

{

if (S.top == S.base)

return ERROR;

e = \*--S.top;

return OK;

}

SElemType GetTop(SqStack S) //取栈顶元素

{

if (S.top != S.base)

return \*(S.top - 1);

}

Status StackEmpty(SqStack S) //判断元素

{

if (S.top == S.base)

return OK;

else

return ERROR;

}

void Matching()

{

SqStack S;

InitStack(S);

int flag = 1,count=1;

char ch,x;

cin >> ch;

while (ch != '#' && flag && count <= 100) {

switch (ch) {

case '{':

case '[':

case '(':

Push(S, ch);

break;

case ')':

if (!StackEmpty(S) && GetTop(S) == '(')

Pop(S, x);

else flag = 0;

break;

case ']':

if (!StackEmpty(S) && GetTop(S) == '[')

Pop(S, x);

else flag = 0;

break;

case '}':

if (!StackEmpty(S) && GetTop(S) == '{')

Pop(S, x);

else flag = 0;

break;

}

cin >> ch;

count++;

}

if (StackEmpty(S) && flag)

cout << "匹配成功" << endl;

else

cout << "匹配失败" << endl;

}

int main()

{

Matching();

return 0;

}

舞伴问题

#include<iostream>

#define OK 1

#define ERROR 0

using namespace std;

#define MAXSIZE 100

typedef int Status;

typedef struct {

char name[20]; //姓名

char sex; //性别，'F'表示女性，'M'表示男性

} Person;

typedef struct {

Person\* base;

int front;

int rear;

}SqQueue;

SqQueue Mdancers, Fdancers;

Status InitQueue(SqQueue& Q)

{

Q.base = new Person[MAXSIZE];

if (!Q.base) return ERROR;

Q.front = Q.rear = 0;

return OK;

}

int QueueEmpty(SqQueue& Q) {

if (Q.front == Q.rear) //队空

return OK;

else

return ERROR;

}

Status EnQueue(SqQueue& Q, Person e)

{

if ((Q.rear + 1) % MAXSIZE == Q.front) return ERROR;

Q.base[Q.rear] = e;

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

return OK;

}

Status DeQueue(SqQueue& Q, Person& e)

{

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

e = Q.base[Q.front];

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

return OK;

}

Person GetHead(SqQueue Q) {

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。

InitQueue(Mdancers); //男士队列初始化

InitQueue(Fdancers); //女士队列初始化

Person p;

for (int i = 0; i <num; i++) //依次将跳舞者根据其性别入队

{

p = dancer[i];

if (p.sex == 'F')

EnQueue(Fdancers, p); //插入女队

else

EnQueue(Mdancers, p); //插入男队

}

cout << "The dancing partners are:" << endl;

while (!QueueEmpty(Fdancers) && !QueueEmpty(Mdancers)) {//依次输出男女舞伴的姓名

DeQueue(Fdancers, p); //女士出队

cout << p.name << " "; //输出出队女士姓名

DeQueue(Mdancers, p); //男士出队

cout << p.name << endl; //输出出队男士姓名

}

if (!QueueEmpty(Fdancers)) { //女士队列非空，输出队头女士的姓名

p = GetHead(Fdancers); //取女士队头

cout << "The first man to get a partner is: " << endl;

cout << p.name << endl;

}

else if (!QueueEmpty(Mdancers)) { //男士队列非空，输出队头男士的姓名

p = GetHead(Mdancers); //取男士队头

cout << "The first woman to get a partner is: " << p.name << endl;

}

}

int main()

{

int num;

cout << "输入人数 ";

cin >> num;

Person dancer[MAXSIZE];

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

{

cout << "请输入第" << i + 1 << "个舞者：";

cin >> dancer[i].name >> dancer[i].sex;

}

DancePartner(dancer, num);

return 0;

}

2.程序的运行结果：所有功能的运行结果都要截屏并打印出来。

括号匹配问题

图片包含 文本

描述已自动生成文本

中度可信度描述已自动生成

文本

低可信度描述已自动生成图形用户界面, 文本

描述已自动生成

舞伴问题

图片包含 图示

描述已自动生成图片包含 文本

描述已自动生成

**四、实验小结**

1.说明遇到的问题及解决方法；

文本, 信件, 白板

描述已自动生成

解决办法：定义SqStack S

图形用户界面, 应用程序

描述已自动生成

图形用户界面, 文本

描述已自动生成

解决办法：将QElemType类型改为Person类型



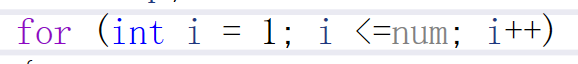
卡通人物

低可信度描述已自动生成

文本

低可信度描述已自动生成

解决办法：改变循环条件；



改为：



2.关于程序的特色和改进设想；

3.经过此实验所得到的收获,其他需要说明的情况。

通过本次实验，加深了对线性结构栈和队列的理解，学会了定义栈和队列的存储结构，加强对栈和队列操作机制的理解，掌握栈后进先出和队列先进先出的基本操作，解决了在调用栈和队列时不会引用及错误调用的问题。