实验项目名称： 栈和队列

**（所属课程：** 数据结构 **）**

**学 院： 计算机科学与信息工程学院 专业班级：网络工程18-1**

**姓 名： 邓二浦 学 号：18031110129**

**实验日期： 2019-10-11 实验地点：a06-203**

**合作者： 指导教师：孙高飞**

**本实验项目成绩： 教师签字： 日期：**

1. **实验目的**

本次实验的目的在于使学生深入了解栈和队列的特征，掌握在实际问题背景下的灵活运用。

1. **实验内容**

2.1 用顺序结构表示栈并实现栈的各种基本操作

2.2 使用栈检查括号匹配的检验

2.3采用链式结构表示队列并实现各种基本操作实现（选做）。

**三 、实验步骤与结果**

2.1实验步骤

* 建立main函数

int main()

{

Seqstack S;

int n,i,e;

initstack(&S);

printf("请输入要进栈元素的个数\n");

scanf("%d",&n);

printf("请输入要进栈的元素\n");

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

{

scanf("%d",&e);

push(&S,e);

}

printf("输出栈的元素\n");

printstack(&S);

printf("执行出栈操作\n");

pop(&S,&e);

printf("出栈后栈中的元素为\n");

printstack(&S);

printf("输出栈顶元素");

gettop(&S,&e);

printf("%d\n",e);

return 0;

}

* 按80页 录入SeqStack类型定义和相关常量定义。

typedef struct

{

int a[50];

int top;

}Seqstack;

* 录入并调试方法InitStack、GetTop、Push、Pop方法

void initstack(Seqstack \*S) //初始化顺序栈

{

S->top=-1;

}

int push(Seqstack \*S,int x) //进栈

{

if(S->top==max-1) return 0;

S->top++;

S->a[S->top]=x;

return 1;

}

int pop(Seqstack \*S,int \*x) //出栈

{

if(S->top==-1)

return 0;

else

{

\*x=S->a[S->top];

S->top--;

return 1;

}

}

int gettop(Seqstack \*S,int \*x) //读取栈顶元素

{

if(S->top==-1)

{

return 0;

}

else

{

\*x=S->a[S->top];

return 1;

}

}

* 编写方法输出栈中元素内容

void printstack(Seqstack \*S) //输出栈中的元素

{

int i;

for(i=0;i<=S->top;i++)

printf("%d ",S->a[i]);

printf("\n");

}

* 调用方法，出栈入栈若干元素，通过输出栈内容，观察栈内容变化

2.2实验思路

* 栈操作为字符，因此需要将StackElemType定义为char

typedef struct node

{

char date;

struct node \*next;

}Linkstack;

* 循环：录入并调试算法3.12

void pipei(char \*str)

{

Linkstack \*L;

int i;char ch;

L=(Linkstack \*)malloc(sizeof(struct node));

L->next=NULL;

for(i=0;str[i]!='\0';i++)

{

switch(str[i])

{

case'(':

case'{':

case'[':

push(L,str[i]);

break;

case')':

case']':

case'}':

if(isempty(L))

{printf("\n右括号多余");return;}

else

{

gettop(L,&ch);

if(match(ch,str[i]))

pop(L,&ch);

else

{printf("\n对应的左右括号不同类");return;}

}

}

}

if(isempty(L))

printf("括号匹配\n");

else

printf("括号不匹配\n");

}

2.3实验步骤

* 录入85页链队列定义

typedef struct node

{

int date;

struct node \*next;

}Linkqueuenode;

typedef struct

{

Linkqueuenode \*front;

Linkqueuenode \*rear;

}Linkqueue;

* 实现85页 算法3.8，3.9。

int enterqueue(Linkqueue \*Q,int x) //链队列入队操作

{

Linkqueuenode \*s;

s=(Linkqueuenode \*)malloc(sizeof(struct node));

if(s!=NULL)

{

s->date=x;

Q->rear->next=s;

Q->rear=s;

s->next=NULL;

return 1;

}

else

return 0;

}

int deletequeue(Linkqueue \*Q,int \*x) //链队出队操作

{

Linkqueuenode \*p;

if(Q->front==Q->rear)

return 0;

p=Q->front->next;

Q->front->next=p->next;

if(Q->rear==p)

Q->rear=Q->front;

\*x=p->date;

free(p);

return 1;

}

* 实现输出队列内元素方法

int printqueue(Linkqueue \*Q) //输出队列中的元素

{

Linkqueuenode \*p;

p=Q->front->next ;

if(Q->front==Q->rear)

{

printf("这是一个空队列\n");

return 0;

}

while(p!=NULL)

{

printf("%d ",p->date);

p=p->next;

}

printf("\n");

return 1;

}

* 入队列、出队列若干数据，观察队列内容变化

1. **实验总结**

栈的特点：先进后出

队列特点：先进先出

栈和队列都是特殊的线性表，是操作受限定的线性表，他们的共同点是操作的位置限制在表的端点；

栈具有LIFO的特性，限定元素的运算位置只在表的一端（栈顶）进行。

队列具有FIFO的特性，限定元素的运算位置分别在表的两端进行。