姓名： 毛翊尧 班级： A1611 学号： 31 号

**实验二 栈和队列**

一、实验目的

1、掌握栈的结构特性及其入栈，出栈操作；

2、掌握队列的结构特性及其入队、出队的操作，掌握循环队列的特点及其操作。

二、实验预习

说明以下概念

1、顺序栈：数据顺序储存的栈。

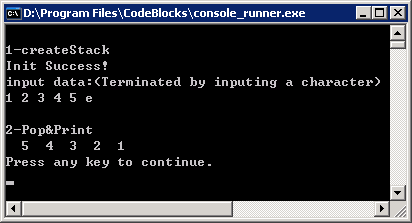
2、链栈：数据链式储存的栈。

3、循环队列：基于数组的储存方式表示，逻辑上，储存队列元素的表为一个环。

4、链队：是基于单链表储存表示的队列。

三、实验内容和要求

1、阅读下面程序，将函数Push和函数Pop补充完整。要求输入元素序列1 2 3 4 5 e，运行结果如下所示。



#include<stdio.h>

#include<malloc.h>

#define ERROR 0

#define OK 1

#define STACK\_INT\_SIZE 10 /\*存储空间初始分配量\*/

#define STACKINCREMENT 5 /\*存储空间分配增量\*/

typedef int ElemType; /\*定义元素的类型\*/

typedef struct{

ElemType \*base;

ElemType \*top;

int stacksize; /\*当前已分配的存储空间\*/

}SqStack;

int InitStack(SqStack \*S); /\*构造空栈\*/

int push(SqStack \*S,ElemType e); /\*入栈\*/

int Pop(SqStack \*S,ElemType \*e); /\*出栈\*/

int CreateStack(SqStack \*S); /\*创建栈\*/

void PrintStack(SqStack \*S); /\*出栈并输出栈中元素\*/

int InitStack(SqStack \*S){

S->base=(ElemType \*)malloc(STACK\_INT\_SIZE \*sizeof(ElemType));

if(!S->base) return ERROR;

S->top=S->base;

S->stacksize=STACK\_INT\_SIZE;

return OK;

}/\*InitStack\*/

int Push(SqStack \*S,ElemType e){

}/\*Push\*/

int Pop(SqStack \*S,ElemType \*e){

}/\*Pop\*/

int CreateStack(SqStack \*S){

int e;

if(InitStack(S))

printf("Init Success!\n");

else{

printf("Init Fail!\n");

return ERROR;

}

printf("input data:(Terminated by inputing a character)\n");

while(scanf("%d",&e))

Push(S,e);

return OK;

}/\*CreateStack\*/

void PrintStack(SqStack \*S){

ElemType e;

while(Pop(S,&e))

printf("%3d",e);

}/\*Pop\_and\_Print\*/

int main(){

SqStack ss;

printf("\n1-createStack\n");

CreateStack(&ss);

printf("\n2-Pop&Print\n");

PrintStack(&ss);

return 0;

}

算法分析：输入元素序列1 2 3 4 5，为什么输出序列为5 4 3 2 1？体现了栈的什么特性？

2、在第1题的程序中，编写一个十进制转换为二进制的数制转换算法函数（要求利用栈来实现），并验证其正确性。

实现代码：

验证：

3、阅读并运行程序，并分析程序功能。

#include<stdio.h>

#include<malloc.h>

#include<string.h>

#define M 20

#define elemtype char

typedef struct

{

elemtype stack[M];

int top;

}

stacknode;

void init(stacknode \*st);

void push(stacknode \*st,elemtype x);

void pop(stacknode \*st);

void init(stacknode \*st)

{

st->top=0;

}

void push(stacknode \*st,elemtype x)

{

if(st->top==M)

printf("the stack is overflow!\n");

else

{

st->top=st->top+1;

st->stack[st->top]=x;

}

}

void pop(stacknode \*st)

{

if(st->top>0) st->top--;

else printf(“Stack is Empty!\n”);

}

int main()

{

char s[M];

int i;

stacknode \*sp;

printf("create a empty stack!\n");

sp=malloc(sizeof(stacknode));

init(sp);

printf("input a expression:\n");

gets(s);

for(i=0;i<strlen(s);i++)

{

if(s[i]=='(')

push(sp,s[i]);

if(s[i]==')')

pop(sp);

}

if(sp->top==0)

printf("'('match')'!\n");

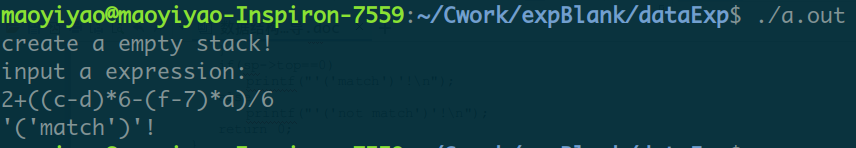
else

printf("'('not match')'!\n");

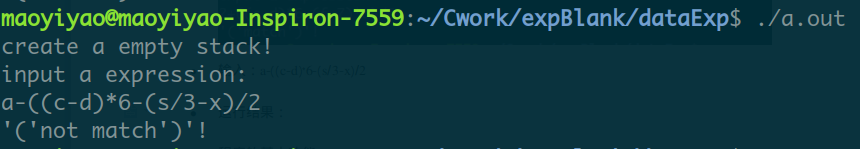
return 0;

}

输入：2+((c-d)\*6-(f-7)\*a)/6

运行结果：

输入：a-((c-d)\*6-(s/3-x)/2

运行结果：

程序的基本功能：

实验小结：