**Project 4 顺序栈与链栈**

学号 2014211116 姓名 骆金参 上交时间 2015-11-1

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 题号 | 程序逻辑(40) | 算法新颖性(20) | 代码规范 (20) | 实验报告(20) | 总分 |
| 得分 |  |  |  |  |  |

1. **问题描述**

顺序栈与链栈插入栈顶元素，删除栈顶元素，依次打印各元素。

**2. 算法描述**

（1）顺序栈插入元素e为新的栈顶元素

Status Push(SqStack \*S,SElemType e)

{

if(S->top == MAXSIZE -1)

{

return ERROR;

}

S->top++;

S->data[S->top]=e;

return OK;

}

（2）顺序链若栈不空，则删除S的栈顶元素，用e返回其值，并返回OK；否则返回ERROR

Status Pop(SqStack \*S,SElemType \*e)

{

if(S->top==-1)

return ERROR;

\*e=S->data[S->top];

S->top--;

return OK;

}

（3）顺序链从栈底到栈顶依次对栈中每个元素显示

Status StackTraverse(SqStack S)

{

int i;

i=0;

while(i<=S.top)

{

visit(S.data[i++]);

}

printf("\n");

return OK;

}

（4）栈链插入元素e为新的栈顶元素

Status Push(LinkStack \*S,SElemType e)

{

LinkStackPtr s=(LinkStackPtr)malloc(sizeof(StackNode));

s->data=e;

s->next=S->top;

S->top=s;

S->count++;

return OK;

}

（5）栈链若栈不空，则删除S的栈顶元素，用e返回其值，并返回OK；否则返回ERROR

Status Pop(LinkStack \*S,SElemType \*e)

{

LinkStackPtr p;

if(StackEmpty(\*S))

return ERROR;

\*e=S->top->data;

p=S->top;

S->top=S->top->next;

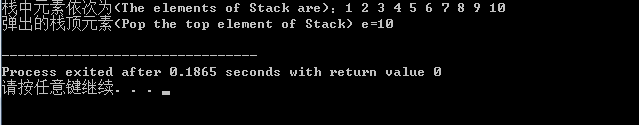
free(p);

S->count--;

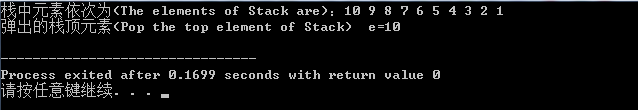
return OK;

}

**3. 测试结果**



**（顺序栈）**



**（链栈）**

**4. 分析与评论**

较为简单，但是还是有很多的细节没有处理好，例如两者在删除站定元素是是有区别的，需好好阅读教材。