顺序栈：

代码：

（截图在代码后面）

#include <iostream>

using namespace std;

#define ERROR 0

#define OK 1

#define MAXSIZE 10

typedef struct

{

int \*base;

int front;

int back;

int stacksize;

} Stack;

int Create\_stack(Stack \*S)

{

S->base = new int[MAXSIZE];

if(!S->base)

{

cout << "创建顺序栈失败" << endl;

return ERROR;

}

S->back = S->front = 0;

S->stacksize = MAXSIZE;

cout << "创建成功" << endl;

return OK;

}

int Push\_front(Stack \*S , int e) //入栈，e表示需要入栈的元素

{

if(S->front - S->back == S->stacksize)

{

cout << "栈满" << endl;

return ERROR;

}

S->base[S->front++] = e;

return OK;

}

int Pop\_front(Stack \*S) //删除栈顶元素

{

if(S->front - S->back == 0)

{

cout << "空栈，没有可删除元素" << endl;

return ERROR;

}

S->base[--S->front] = 0;

cout << "删除成功" << endl;

return OK;

}

int Get\_front(Stack \*S , int &e) //取栈顶元素

{

if(S->front - S->back == 0)

{

cout << "空栈，无法取出栈顶元素" << endl;

return ERROR;

}

--S->front;

e = S->base[S->front]; //取出元素

S->base[S->front] = 0; //重置取出位置的值为0

}

int Show\_stack(Stack \*S) //遍历顺序栈

{

if(S->front - S->back == 0)

{

cout << "空栈，无法遍历栈内元素" << endl;

return ERROR;

}

int length = S->front - S->back;

cout << "从栈顶到栈底依次为：" << endl;

for(int i = length-1 ; i >= 0 ; --i) //实现遍历操作

{

cout << S->base[i] << "\t";

}

cout << endl;

return OK;

}

int Empty\_stack(Stack \*S)

{

if(S->front - S->back == 0)

{

cout << "空栈，无需置空操作" << endl;

return ERROR;

}

int length = S->front - S->back;

for(int i = length-1 ; i >= 0 ; --i)

{

S->base[i] = 0;

}

S->front = S->back; //初始化栈内标记

cout << "置空完毕！" << endl;

}

int Ten\_to\_eight(Stack \*S , int &e)

{

if(S->front - S->back > 0)

{

cout << "当前栈不为空，无法数制转换，请先置空栈" << endl;

return ERROR;

}

int temp = e;

e = 0;

int yu = 0;

while(temp) //转换后入栈

{

yu = temp%8;

Push\_front(S , yu);

temp /= 8;

}

int length = S->front - S->back;

int t = 0;

while(length--) //出栈输出

{

e \*= 10;

Get\_front(S , t);

e += t;

}

cout << "数制转换后为：" << e << endl;

}

void menu()

{

cout << "1、初始化栈" << endl; //finish

cout << "2、插入元素（入栈）" << endl; //finish

cout << "3、删除栈顶元素" << endl; //finish

cout << "4、取栈顶元素" << endl; //finish

cout << "5、遍历顺序栈" << endl; //finish

cout << "6、置空顺序栈" << endl; //finish

cout << "7、数制转换" << endl;

cout << "8、退出程序" << endl;

cout << "-------------------------------->>" << endl;

}

int main()

{

Stack \*stack = new Stack;

stack == NULL;

char option;

int num = 0;

while(1)

{

menu();

cout << "请选择要执行的操作：";

cin >> option;

fflush(stdin);

if(option < '1' || option > '8')

{

cout << "输入不合法" << endl;

continue;

}

switch(option)

{

case '1':

Create\_stack(stack);

break;

case '2':

cout << "请输入需要入栈的数字；";

cin >> num;

Push\_front(stack , num);

break;

case '3':

Pop\_front(stack);

break;

case '4':

Get\_front(stack , num);

cout << "取出的值为：" << num << endl;

break;

case '5':

Show\_stack(stack);

break;

case '6':

Empty\_stack(stack);

break;

case '7':

cout << "请输入需要转换的数字（十进制转八进制）；";

cin >> num;

Ten\_to\_eight(stack , num);

break;

case '8':

cout << "退出成功" << endl;

exit(1);

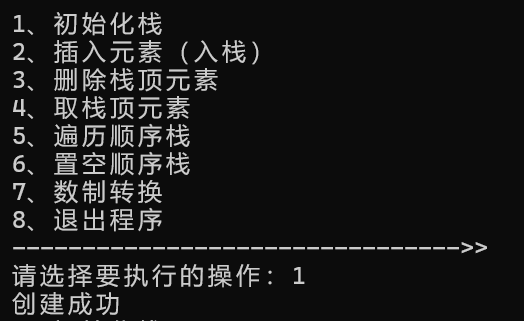
break;

}

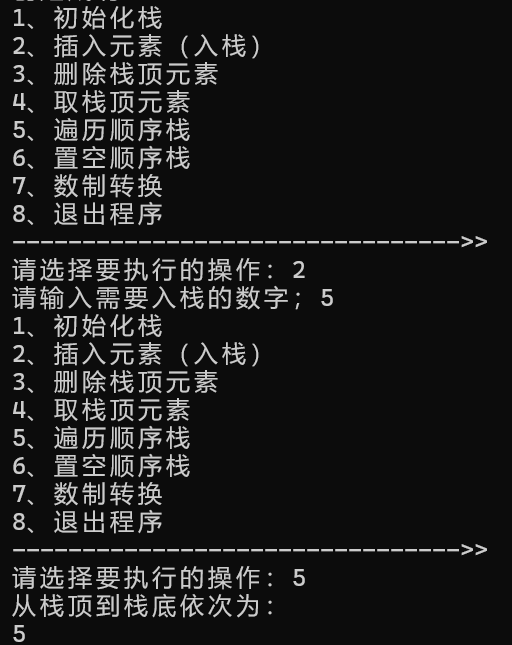
}

}

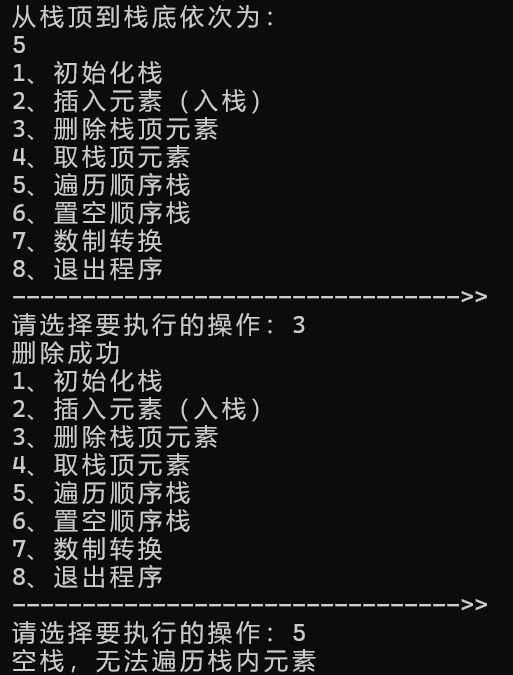
初始化栈



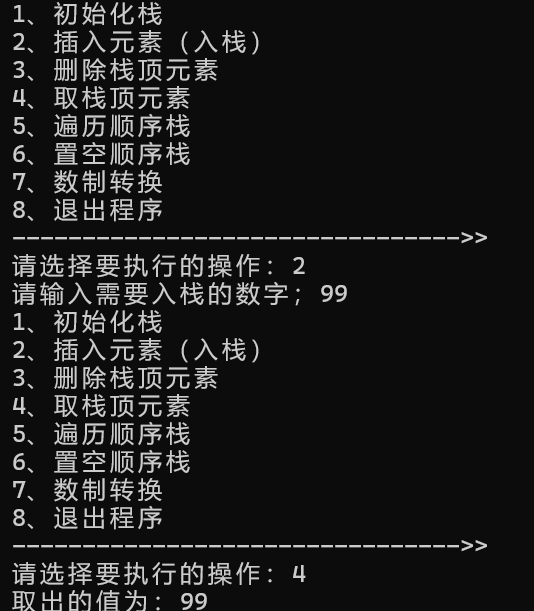
插入元素（入栈）



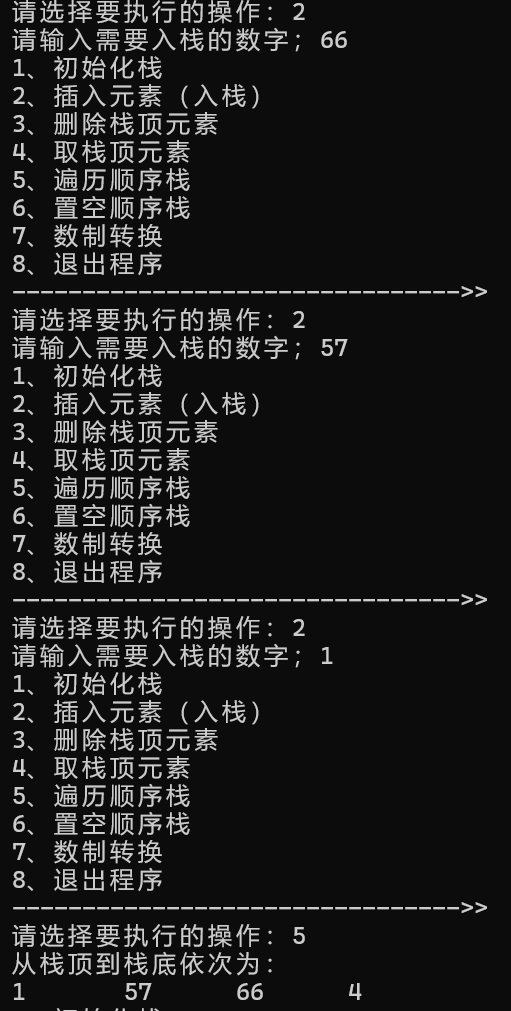
删除栈顶元素



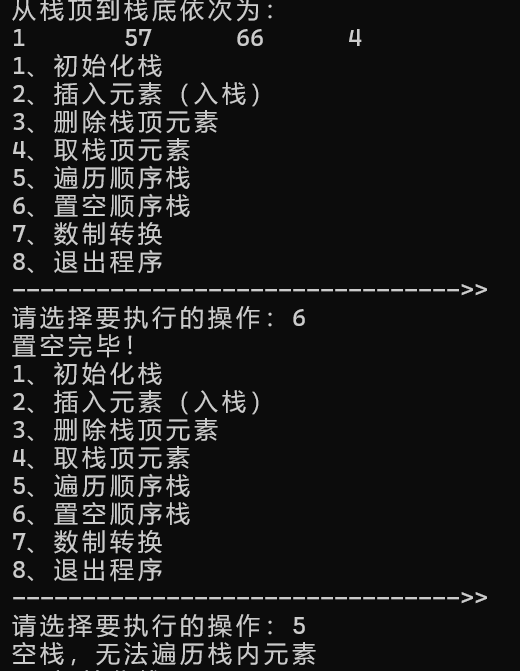
取栈顶元素



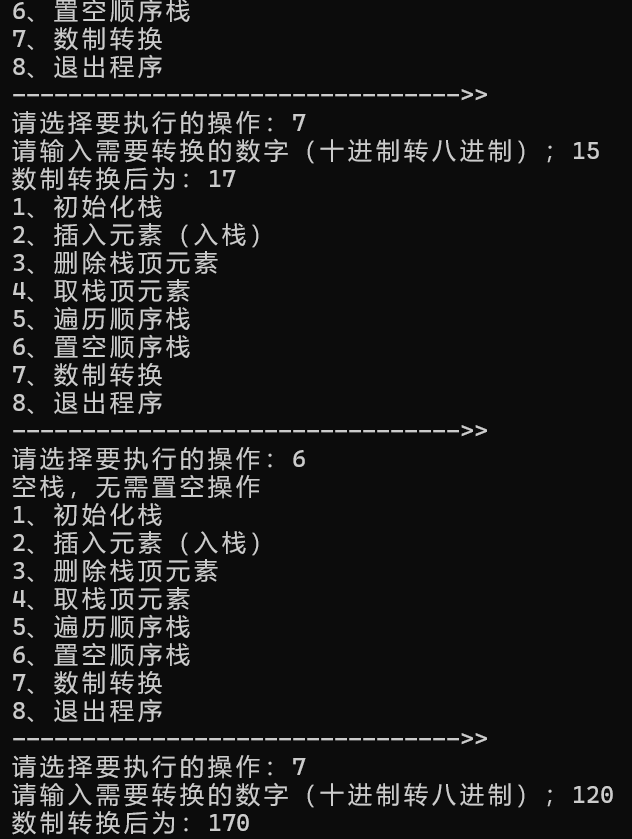
遍历顺序栈



置空顺序栈



数制转换



错误报告：

