**第三章作业**

**第一题（3.1） 对一个栈的输入序列 a1,a2,a3,… ,an,，称由此栈依次出栈后所得到的元素序列为栈 的合法输出序列。例如，假设栈 S 的一个输入序列为 1,2,3,4,5，则可得到多个输出序列， 例如，1,2,3,4,5 就是一个合法的输出序列，同理，5,4,3,2,1 和 3,2,1,4,5 也分别是其合 法的输出序列。分别求解下列问题：**

**（1）判断序列 1,3,4,5,2 是否是合法的输出序列。**

算法分析：由于出栈时可能存在入栈还没有结束的情况，一个输出序列中的元素之后的元素，对于原序列1, 2,3,4,5来说，在原序列中相对位置比它小的，必须是逆序输出；在原序列中相对位置比它大的，输出顺序没有要求

算法实现：

#include <iostream>

using namespace *std*;

bool Stackright(int a[])

{

int min = 0;

for (int i = 0; i < 5; ++i)

{

min = a[i];

for (int j = i; j < 5; ++j)

{

if (a[j] < a[i])//在i元素之后找到比元素i小的元素

{

if (a[j] > min) //验证元素是否按照降序排列

return false;

else

min= a[j];

}

}

}

return true;

}

int main() {

*cout* << "请输入要判断的序列" << *endl*;

int y,a[5];

for (int i=0;i<5;i++)

{

*cin* >> y;

a[i] = y;

}

if (Stackright(a))

{

*cout* << "该序列合法" << *endl*;

}

else

{

*cout* << "该序列不合法" << *endl*;

}

return 0;

}

**（2）对输入序列 1,2,3,4,5，求出其所有的合法的输出序列。**

算法分析：

首先对序列进行全排列，需要用到全排列函数，之后运用上题所写的函数进行判断合法性，合法则输出序列

算法实现：

bool Stackright(int a[])

{

int min = 0;

for (int i = 0; i < 5; ++i)

{

min = a[i];

for (int j = i; j < 5; ++j)

{

if (a[j] < a[i])//在i元素之后找到比元素i小的元素

{

if (a[j] > min) //验证元素是否按照降序排列

return false;

else

min= a[j];

}

}

}

return true;

}

int main() {

int a[5] = { 1,2,3,4,5 };

while (*next\_permutation*(a, a + 5)) {

if (Stackright(a))

{

for (int i=0;i<5;i++)

{

*cout* << a[i];

}

*cout* << *endl*;

}

}

return 0;

}

**（3\*）设计算法以判断对输入序列 1,2,3,… ,n，序列 a1,a2,a3,… ,an是否是该栈的合法 的输出序列（假设输出序列在数组 A 中）。**

算法分析：

在上一题的基础上要增加一个判断两个序列元素是否相等的步骤，先判断两个序列长度是否相同，不同直接输出不合法，相同则对输出序列升序排序；再比较输入序列与排序后序列的元素是否相等，如果两个序列不相等，输出不合法，相等，则进行下一步判断输出序列是否合法。

算法实现：

#include <algorithm>

#include <iostream>

using namespace std;

typedef struct Stack

{

int data[100];

int top;

}Stack;

void InitStack(Stack& S)

{

S.top = -1;

}

void Push(Stack& S, int x)

{

S.data[++S.top] = x;

}

void Pop(Stack& S, int& x)

{

x = S.data[S.top--];

}

bool isEmpty(Stack& S)

{

if (S.top == -1)

return true;

else

return false;

}

bool judgenum(int a[],int alen,int b[],int blen)

{

if (alen!=blen)

{

return false;

}

else

{

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

{

if (a[i] != b[i])

{

return false;

}

}

}

}

bool Stackright( int a[])

{

int min = 0;

for (int i = 0; i < 5; ++i)

{

min = a[i];

for (int j = i; j < 5; ++j)

{

if (a[j] < a[i])//在i元素之后找到比元素i小的元素

{

if (a[j] > min) //验证元素是否按照降序排列

return false;

else

min = a[j];

}

}

}

return true;

}

int main()

{

Stack S;

InitStack(S);

int alen,blen;

cin >>alen;

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

{

int x;

cin >> x;

Push(S, x);

}

int arr[100] ;

for ( int j = 0; j< alen; j++)

{

int y=0;

if (!isEmpty(S))

{

Pop(S,y);

arr[j] = y;

}

}

int b[4] = { 4,1,3,2 };

blen = sizeof(b) / sizeof(int);

if(!judgenum(arr, alen, b, blen))

{

cout << "不是" << endl;

}

else

{

if (!Stackright(b))

{

cout << "不是" << endl;

}

else

{

cout << "是"<<endl;

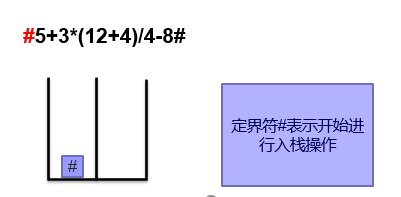
}

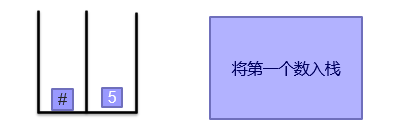
}

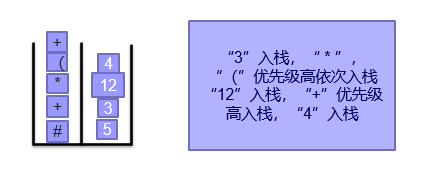
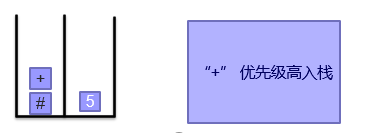
return 0;

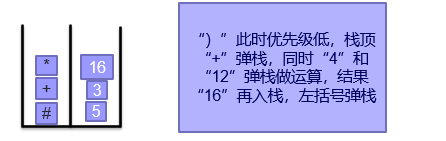
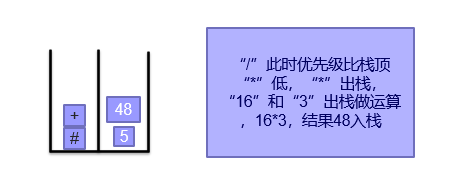
}

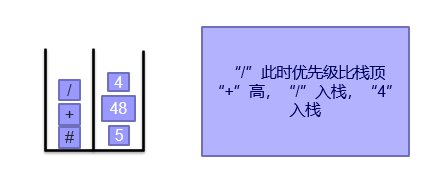
**第二题（3.5）对表达式 5+3\*(12+4)/4-8，依次画出在求解过程中的各步骤中的栈的状态。**

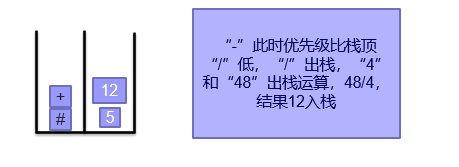


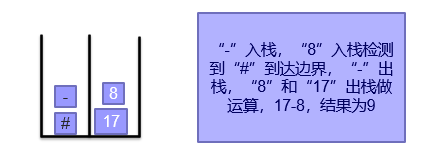
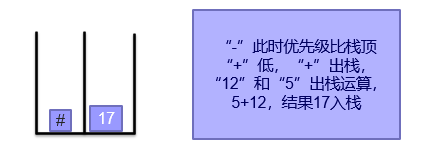












**第三题（3.13）**

**阅读下列程序，并写出其运行结果:**

**(1) void P(int W) {**

**if (W>0)**

**{ P(W-1);**

**cout<<W;**

**}**

**}**

**调用P（4）**

**结果为 1234**

**（2） void P(int W) {**

**if (W>0)**

**{ cout<<W;**

**P(W-1);**

**}**

**}**

**调用 P(4);**

**结果为 4321**

**（3） void P(int W) {**

**if (W>0)**

**{ cout<<W;**

**P(W-1);**

**cout<<W;**

**}**

**}**

**调用 P(4);**

**结果为43211234**

**（4）void P(int W)**

**{**

**if (W>0)**

**{**

**P(W-1);**

**P(W-1);**

**cout<<W;**

**}**

**}**

**调用 P(4);**

**结果为112112311211234**

**（5）int F(int N)**

**{**

**if ( N==0) F=0;**

**else if (N==1)**

**return 1;**

**else**

**return F(N-1)+2\*F(N-2);**

**}**

**调用 cout<<F(5);**

**结果为11**

**(6) void P(int N,int \*F)**

**{**

**if (N==0)**

**\*F=0;**

**else { P(N-1, \*F); F=F+N; }**

**}**

**调用 P(4,\*M); cout<<\*M**

**结果为10**

**第四题（ppt例3.10）**

**对下面的程序代码，画出调用F(5)的执行路线图，并给出执行结果。**

**int F( int n )**

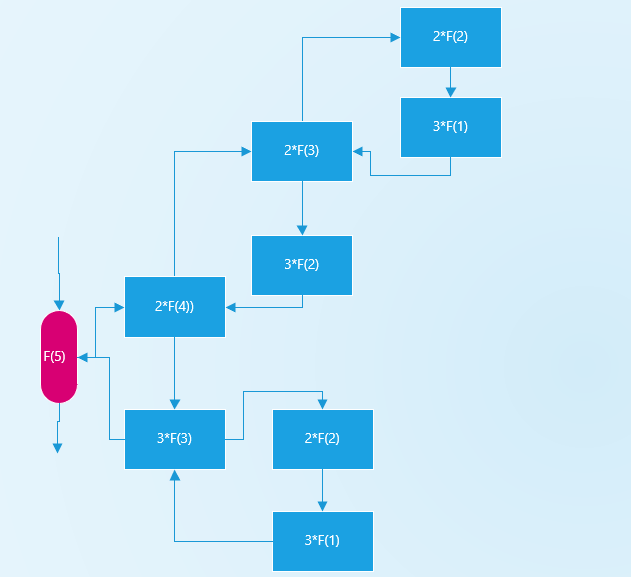
**{**

**if( n<=2 )**

**return 1;**

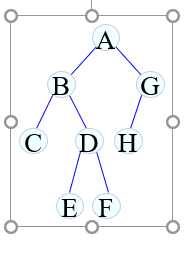
**return 2\*F(n-1) + 3\*F(n-2);**

**}**

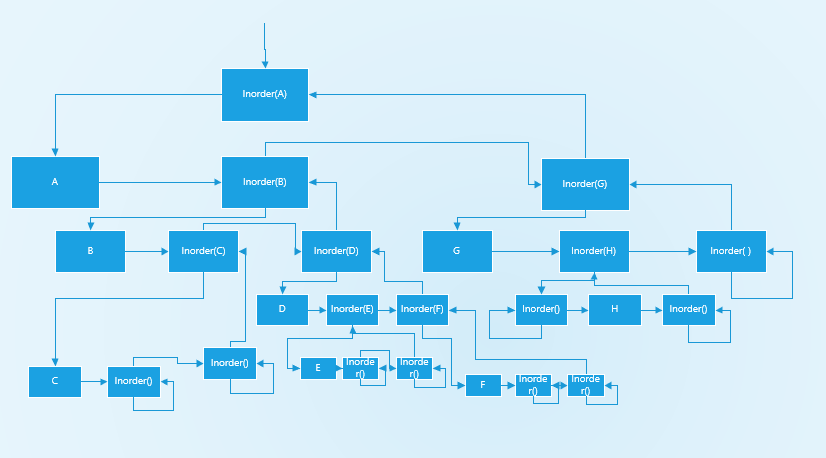


**结果为41**

**第五题（二叉树ppt88页图）按照ppt方法先序，中序遍历下面这棵二叉树，需画图**



**先序**



**中序**