一．（1）

int Max(Linklist p)

{

if (!p->next)

return p->data;

else

{

int max = Max(p->next);

return p->data >= max ? p->data : max;

}

}

（2）

int Length(Linklist p)

{

if (!p->next)

return 1;

else {

return Length(p->next) + 1;

}

}

（3）

int Add(Linklist p,int n)

{

if (!p->next)

return p->data;

else {

double ave = Add(p->next, n - 1);

return (ave \* (n - 1) + p->data) / n;

}

}

二．判断入栈序列是否合法（注：这里写的程序只有输入正常的可能出栈序列时才能正确运行，如输入：7，4，3，5，6，2，1。如果输入：5，7，5则程序无法正确运行；默认入栈顺序为1-n）

#include<iostream>

using namespace std;

struct Stack {

int data;

int\* base, \* top;

};

void InitStack(Stack& s) {

s.base = new int[105];

s.top = s.base;

}

void Push(Stack& s, int value) {

\*(s.top++) = value;

}

void Pop(Stack& s) {

\*s.top--;

}

bool Empty(Stack s) {

if (s.base == s.top) {

return true;

}

return false;

}

int GetTop(Stack s) {

return \*(s.top - 1);

}

int main() {

int m = 0;

char c = 0;

int n = 0;

int arr[100] = { 0 };

for (m = 0;;m++)

{

scanf\_s("%d", &arr[m]);

c = getchar();

n++;

if (c != ',')

break;

}

Stack s;

InitStack(s);

for (int i = 0, j = 0;i < n;i++) {

Push(s, i + 1);

while (!Empty(s) && GetTop(s) == arr[j]) {

Pop(s);

j++;

}

}

if (Empty(s)) {

cout << "合法" << endl;

}

else {

cout << "不合法" << endl;

}

}

三．表达式求值

#include <stdio.h>

int a[10] = { 0,1,2,3,4,5,6,7,8,9 };

int operate[10];//操作

int b[10];

void print(int sum)

{

printf\_s("%d", b[1]);

int i, j;

for (i = 2, j = 1;i < sum + 1;i++)

{

while (operate[j] == 0)

j++;//如果操作符为空，则使j增加直至达到第一个操作符不为空的j的位置

if (operate[j] == 1)

printf\_s("+");

else

printf\_s("-");

j++;

printf\_s("%d", b[i]);

}

printf\_s("\n");

}

void judge()

{

int sum = 1;

int i, j = 1, now = 0;

b[1] = a[1];

for (i = 1;i < 9;i++)

{

if (operate[i] == 0)

b[sum] = b[sum] \* 10 + a[i + 1];

else

{

sum++;

b[sum] = a[i + 1];//‘+’或‘-’则把a[i+1]都当作个位数处理即可

}

}

now = b[1];

for (i = 2, j = 1;i < sum + 1;i++)

{

while (operate[j] == 0)

j++;

if (operate[j] == 1)

now += b[i];

else

now -= b[i];

j++;

}

if (now == 110)

print(sum);

}

void dfs(int k)

{

if (k == 9)

{

judge();

return;

}

int i;

for (i = -1;i < 2;i++)

{

operate[k] = i;

dfs(k + 1);

}

}

int main()

{

dfs(1);

return 0;

}