**第5章课后作业**

1.括号检验：

输入一个代数表达式，表达式只能含有+,-,\*,/,(,),1,2,3,4,5,6,7,8,9,0字符且每个数字均小于10，设表达式除括号匹配有误外，再无其他错误。编写算法对输入的表达式进行检验，判断括号匹配是否正确。

正确的： 错误的：

1+2+4  (1+)2

(1+2)+4  (1+2(4+3))  
(1+2)  (1+2+3\*(4+5()))  
 1+2+3\*(4+5))

**源代码：**

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

struct s\_node

{

char data;

struct s\_node \*next;

};

typedef struct s\_node s\_list;

typedef s\_list \*link;

link stack = NULL;

void print\_stack()

{

link temp = NULL;

temp = stack;

if (temp == NULL)

printf("The stack is empty!!\n");

else

{

while (temp != NULL)//意思是如果temp=NULL的时候就不循环了

{

printf("[%d] ", temp->data);

temp = temp->next;

}

printf("\n");

}

}

int is\_empty()

{

if (stack == NULL)

return 1;

return 0;

}

void push(char value)

{

link newnode;

newnode = (link)malloc(sizeof(s\_list));

newnode->data = value;

newnode->next = stack;

stack = newnode;

}

int pop()

{

link top;

int temp;

if (stack != NULL)

{

top = stack;

stack = stack->next;

temp = top->data;

free(top);

return temp;

}

else

return -1;

}

int is\_op(char ch)

{

if (ch == '+' || ch == '-' || ch == '\*' || ch == '/')

return 1;

else

return 0;

}

int is\_num(char ch)

{

if (ch <= '9'&&ch >= '0')

return 1;

return 0;

}

void test(char \*buf)

{

int len;

int i;

len = strlen(buf);

for (i = 0; i<len; i++)

{

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

{

if (buf[i + 1] == ')')

{

printf("无效的表达式\n");

break;

}

else

push(buf[i]);

}

if (is\_op(buf[i]))

{

if (buf[i + 1] == ')')

{

printf("无效的表达式\n");

break;

}

}

if (is\_num(buf[i]))

{

if (buf[i + 1] == '(')

{

printf("无效的表达式\n");

break;

}

}

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

{

if (pop() == -1)

{

printf("无效的表达式\n");

break;

}

}

}

if (is\_empty())

{

printf("正确的表达式\n");

}

else

printf("无效的表达式\n");

}

int main()

{

char buf[100];

printf("输入代数表达式\n");

scanf("%s", buf);

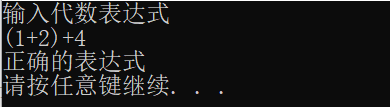
test(buf);

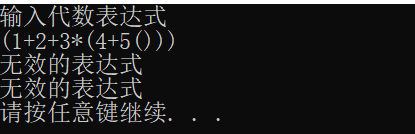
system("pause");

return 0;

}

**结果示例：**

****



2.有一个由数字1,2,3，……，9,组成的数字串（长度不超过200），问如何将M（M<=20）个加号插入到这个数字串中，使所形成的算术表达式的值最小。请编写算法解决这个问题。

注意：加号不能加在数字串的最前面或最末尾，也不应有两个或两个以上的加号相邻。M保证小于字符串的长度。

**源代码：**

#include<stdlib.h>

#include<stdio.h>

#define N 5 //N个数字

#define M 2 //M个加号

char buf[N];

int a[N];

char b[M+1][N];

int c[M+1];

int try(int t);

void swap(int t1,int t2);

int add();

void output();

int min=99999;

int main()

{

int i;

for(i=0;i<N;i++)

{

scanf("%c",&buf[i]);

}

a[0]=0;

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

{

a[i]=1;

}

for(;i<N;i++)

{

a[i]=0;

}

try(1);

output();

printf("%d\n",min);

system("pause");

return 0;

}

int try(int t)

{

int j;

int i;

int sum;

if(t>=N)

{

sum=add();

if(sum<min)

{

min=sum;

for(i=0;i<M+1;i++)

{

c[i]=atoi(b[i]);

}

}

}

else

{

for(j=t;j<N;j++)

{

{

swap(t,j);

try(t+1);

swap(t,j);

}

}

}

}

void swap(int t1,int t2)

{

int t;

t=a[t1];

a[t1]=a[t2];

a[t2]=t;

}

int add()

{

int sum=0;

int i=0;

int j;

int k=0;

int h=0;

for(i=0;i<M+1;i++)

for(j=0;j<N;j++)

b[i][j]='Q';

i=0;

j=0;

h=0;

k=0;

for(j=0;j<N;j++)

{

if(a[j]==1)

{

h=0;

i++;

b[i][h]=buf[j];

h++;

}

else

{

b[i][h]=buf[j];

h++;

}

}

for(i=0;i<M+1;i++)

{

sum+=atoi(b[i]);

}

return sum;

}

void output()

{

int i;

for(i=0;i<M+1;i++)

{

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

if(i!=M)

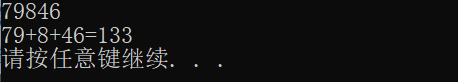
printf("+");

}

printf("=");

}

**结果示例：**

****

5.一个整数n(n<=100)可以有多种分划，分划整数之和为n。例如：

输入n=6

6

5 1

4 2

4 1 1

3 3

3 2 1

3 1 1 1

2 2 2

2 2 1 1

2 1 1 1 1

1 1 1 1 1 1

Total=11{表示分划数有11种}

求n的分划数。

**源代码：**

#include<stdio.h>

#include<stdlib.h>

int divinteger(int n,int m);

int main()

{

int n;

printf("n=");

scanf("%d",&n);

printf("有%d种划分\n",divinteger(n,n));

system("pause");

return 0;

}

int divinteger(int n,int m)

{

if(n<1||m<1)

{

printf("error");

}

else

if(n==1||m==1)

{

return 1;

}

else

if(n<m)

return divinteger(n,n);

else

if(n==m)

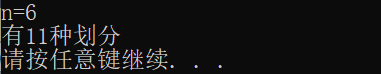
return 1+divinteger(n,n-1);

else

return divinteger(n,m-1)+divinteger(n-m,m);

}

**结果示例：**

****

11.设有A,B,C,D,E5人从事J1，J2,J3,J4,J5 5项工作，每人只能从事一项，他们的效益如图所示，求最佳安排使效益最高。

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | J1 | J2 | J3 | J4 | J5 |
| A | 10 | 11 | 10 | 4 | 7 |
| B | 13 | 10 | 10 | 8 | 5 |
| C | 5 | 9 | 7 | 7 | 4 |
| D | 15 | 12 | 10 | 11 | 5 |
| E | 10 | 11 | 8 | 8 | 4 |

**源代码：**

#include<stdio.h>

#include<stdlib.h>

#define N 5

int a[N];

int c[N\*2],d[N\*2];

int buf[N][N]={

{10,11,10,4,7},

{13,10,10,8,5},

{5,9,7,7,4},

{15,12,10,11,5},

{10,11,8,8,4}

};

int max=0;

int maxa[N];

void try(int t);

void swap(int t1,int t2);

int main()

{

int i;

for(i=0;i<N;i++)

a[i]=i;

for(i=0;i<N;i++)

{

c[i]=0;

c[N+i]=0;

d[i]=0;

d[N+i]=0;

}

try(0);

printf("max=%d\n",max);

printf("A B C D E\n");

for(i=0;i<N;i++)

printf("%d ",maxa[i]);

printf("\n");

system("pause");

return 0;

}

void try(int t)

{

int j;

if(t>=N)

{

int i;

int sum=0;

for(i=0;i<N;i++)

sum+=buf[i][a[i]];

if(sum>max)

{

max=sum;

for(i=0;i<N;i++)

maxa[i]=a[i];

}

}

else

{

for(j=t;j<N;j++)

{

swap(t,j);

if(c[t+a[t]]==0&&d[t-a[t]+N]==0)

{

c[t+a[t]]=1;

d[t-a[t]+N]=1;

try(t+1);

c[t+a[t]]=0;

d[t-a[t]+N]=0;

}

swap(t,j);

}

}

}

void swap(int t1,int t2)

{

int t;

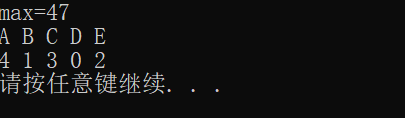
t=a[t1];

a[t1]=a[t2];

a[t2]=t;

}

**结果示例：**

****

12.一个正整数有可能可以被表示为n(n>=2)个连续正整数之和，如n=15时

15=1+2+3+4+5

15=4+5+6

15=7+8

请编写算法，根据输入的任何一个正整数，找出符合这种要求的所有连续正整数序列。

**源代码：**

#include<stdio.h>

#include<stdlib.h>

void try(int n);

int main()

{

int n;

scanf("%d",&n);

try(n);

system("pause");

return 0;

}

void try(int n)

{

int i;

int sum=0;

int first=1,end=1;

while(end<=n/2+2)

{

if(sum<n)

{

sum+=end;

end++;

}

else

if(sum==n)

{

for (i = first; i < end; i++)

{

printf("%d ", i);

}

printf("\n");

sum+=end;

end++;

}

else

if(sum>n)

{

first++;

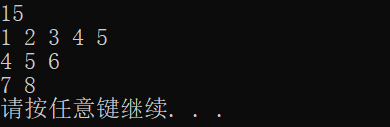
sum-=(first-1);

}

}

}

**结果示例：**

****