**2017年内蒙古大学计算机复试笔试题目及源码解析**

2017-03-26 Category: [学习笔记](http://yangyingming.com/category/%E5%AD%A6%E4%B9%A0%E7%AC%94%E8%AE%B0/) Tags: [C/C++](http://yangyingming.com/tag/3/) [Leave a Comment](http://yangyingming.com/article/371/#comments)[8589 Read](http://yangyingming.com/article/371/)[Edit](http://yangyingming.com/admin/blog/article/371)

题目是考完了去网吧回忆出来的，除了一些细节，大体意思基本和原题一样。如果你看着没毛病，那说明我语言描述能力还行，哈哈。

代码也是我在网吧现敲出来的，由于源码在网吧 vc++6.0 和 codeblocks 以及 dev c 等IDE环境中编译运行有问题（可能网吧做了限制？百思不得姐），所以用 ssh 远程连接了我的 VPS，在 linux 环境下用 g++ 编译运行测试的。

以下输入输出测试样例均在 g++ 编译器下测试成功。

**第一题**

**题目**

输入一组由正整数组成的数字序列，序列大小不超过500，遇到-1结束输入，判断这个数字序列是否是升序。如果是升序，输出“YES”，如果不是升序，输出“NO”。

**源码**

#include<iostream>

using namespace std;

int main()

{

int a[600] = {0};

int i;

for(i=0;i<600;i++){ //输入数字序列

cin>>a[i];

if(a[i]==-1)

break;

}

for(i=1;i<600;i++){ //判断是否升序

if(a[i]==-1)

break;

if(a[i]<=a[i-1])

break;

}

if(a[i]==-1) //输出判断结果

cout<<"YES"<<endl;

else

cout<<"NO"<<endl;

return 0;

}

**输入样例**

1 2 3 -1

1 3 2 -1

1 2 3 3 5 8 7 10 -1

**输出样例**

YES

NO

NO

**注：每组样例请分开测试，即一次只运行一组。下同。**

**第二题**

**题目**

输入一个正整数n，判断不超过n的所有数中与7有关的数有多少个。如果某个数是7或者是7的倍数，或者这个数的某一位是7，那么这个数便与7有关。举个例子，如果输入的n是20，那么20以内与7有关的数便是：7、14、17，最终输出结果是3。

**源码**

#include<iostream>

using namespace std;

int getAnswer(int n) //计算与7有关的数字个数

{

int i,cnt=0;

for(i=1;i<=n;i++){ //循环n以内的所有数

if(i%7==0) //如果当前数是7的倍数，计数+1

cnt++;

else{

int t = i;

while(t){

if(t%10==7){ //如果当前数的某一位是7，计数+1

cnt++;

break;

}

t/=10;

}

}

}

return cnt;

}

int main()

{

int n;

cin>>n;

cout<<getAnswer(n)<<endl;

return 0;

}

**输入样例**

5

10

20

100

**输出样例**

0

1

3

30

**第三题**

**题目**

输入一个正整数，判断这个数是否是“回文数”。所谓“回文数”，便是正着念和反着念都一样的数。比如，1、11、121、1221、12321，都是回文数。如果这个输入的数是回文数，那么输出“YES”，如果不是回文数，输出“NO”。

**源码**

#include<iostream>

#include<string.h>

using namespace std;

int isHuiwen(char num[]) //判断是否是回文数

{

int i,len = strlen(num);

for(i=0;i<len;i++) //前后依次对比每一位数字，遇到不相等的情况，退出循环

if(num[i]!=num[len-i-1])

break;

if(i<len)

return 0;

else

return 1;

}

int main()

{

char num[1000]={0};

cin>>num;

if(isHuiwen(num)) //输出判断结果

cout<<"YES"<<endl;

else

cout<<"NO"<<endl;

return 0;

}

**输入样例**

1

11

121

1221

12321

123

1313

123456789876543210

**输出样例**

YES

YES

YES

YES

YES

NO

NO

NO

**第四题**

**题目**

输入两个字符串，str1 和 str2，将字符串 str2 逆序之后连接到字符串 str1 后面，最后输出str1。举个例子，如果输入的字符串 str1 是"abc", 字符串 str2 是"123"，那么逆序连接之后，输出的结果为"abc321"。

**源码**

#include<iostream>

#include<string.h>

using namespace std;

void mycat(char str1[],char str2[])

{

int i = strlen(str2)-1;

int j = strlen(str1);

for(;i>=0;i--) //逆序连接

str1[j++] = str2[i];

str1[j] = '\0';

}

int main()

{

char str1[1100] = {0};

char str2[1100] = {0};

cin>>str1; //输入两个字符串

cin>>str2;

mycat(str1,str2); //逆序连接

cout<<str1<<endl; //输出连接后的结果

return 0;

}

**输入样例**

abc 123

12345 12345

/\*\*\* /\*\*\*

**输出样例**

abc321

1234554321

/\*\*\*\*\*\*/

**第五题**

**题目**

首先输入两个正整数，m(2<=m<=36)和n(2<=m<=36)，然后输入一个m进制的数字串，将这个数字串转换为n进制并输出。如果进制超过9，那么用小写字母'a'到'z'依次表示10~35的数字。（假设数字串对应的十进制数用int存储不溢出）

**源码**

#include<iostream>

#include<string.h>

using namespace std;

int toTen(int m,char s[]) //将m进制的数字串转换为十进制并返回

{

int i,d = 1,num = 0;

for(i=strlen(s)-1;i>=0;i--){ //循环该数字串，将每一位上的位值和位权相乘，累加结果便为对应的十进制数

int t;

if('a'<=s[i] && s[i]<='z')

t = int(s[i]-'a'+10);

else

t = int(s[i]-'0');

num+=t\*d;

d\*=m;

}

return num;

}

void toN(int num,int n,char ans[]) //将十进制数num转换为n进制数，并存储在ans数组中

{

int i=0;

char temp[1100] = {0};

while(num){ //辗转相除法

int yu = num%n;

num/=n;

if(yu>9)

temp[i++] = char(yu-10+'a');

else

temp[i++] = char(yu+'0');

}

int len = i;

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

ans[len-i-1] = temp[i];

ans[len] = '\0';

}

int main()

{

int m,n;

char s[1100] = {0};

char ans[1100] = {0};

cin>>m>>n; //输入m和n，以及m进制的数字串

cin>>s;

int num = toTen(m,s); //将m进制的数字串转换为十进制数

toN(num,n,ans); //将十进制数转换为n进制数

cout<<ans<<endl; //输出转换后的结果

return 0;

}

**输入样例**

10 2

5

10 2

64

10 2

1024

16 2

ffff

16 8

ffff

16 10

ffff

20 10

g

**输出样例**

101

1000000

10000000000

1111111111111111

177777

65535

16

**第六题**

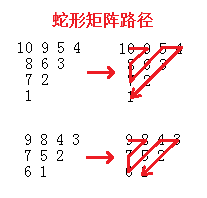
**题目**

输入一个正整数n(n<=30)，输出n所对应的蛇形矩阵。举两个例子：

n=10时，蛇形矩阵为：   
10 9 5 4   
8 6 3   
7 2   
1

n=9时，蛇形矩阵为：   
9 8 4 3   
7 5 2   
6 1

具体的蛇形矩阵变化路径见下图：

[](http://cdn.yangyingming.com/uploads/markdownx/2017/7/34aa4fd4-e8e0-4f56-886f-a46ec69e4fd0.png)

所以只要根据路径的变化规律依次生成蛇形矩阵中的数字，然后输出矩阵就可以了。

生成矩阵的主要 **思路** 是按顺序将递减的数字依次填到相应的位置，解题的关键是如何确认下一步的矩阵位置。在我的算法里，我用变量 s 控制路径拐弯之后到下一次拐角之间的路径长度，变量 i 控制下一个位置的行号，s-i 控制下一个位置的列号，d 控制路径方向（向下走还是向上走），n 控制下一个要填的数字。有了下一个矩阵的位置和数字，将数字递减到 1 即可生成蛇形矩阵。

一个 **小窍门** 是在矩阵中标出每个位置的行号和列号 (i,j) ，然后观察 i 和 j 的变化规律，想出办法用几个变量去控制这个规律即可。

**源码**

#include<iostream>

using namespace std;

void display(int n,int a[][10]) //输出n对应的蛇形矩阵

{

int s =0,i=0,d=-1;

while(n>=1){ //生成蛇形矩阵，存储在a[][10]数组中

int cnt = 0;

while(cnt<=s){

a[i][s-i] = n--;

cnt++;

i+=d;

}

s++;

d = -d;

i = d>0?0:s;

}

int j;

for(i=0;i<10;i++){ //输出n对应的蛇形矩阵

if(a[i][0]<=0)

break;

for(j=0;j<10;j++)

if(a[i][j]>0)

cout<<a[i][j]<<' ';

cout<<endl;

}

}

int main()

{

int n,a[10][10] = {0};

cin>>n;

display(n,a);

return 0;

}

**输入样例**

10

9

30

**输出样例**

10 9 5 4

8 6 3

7 2

1

9 8 4 3

7 5 2

6 1

30 29 25 24 16 15 3 2

28 26 23 17 14 4 1

27 22 18 13 5

21 19 12 6

20 11 7

10 8

9