格式化输出（方法是找规律，找递推关系，用循环）

Eg1.输出：

/\*思路：找规律，有几个\*有几个空格，通过循环语句实现\*/

#include <stdio.h>

**int** main()

{

**int** i,n,m,k,j,x;

**for** (i=1;i<=4;i++)   //定义行数为i

    {

        n=9-2\*i;   //找到关系 行数ix2 + 每行\*的个数n ==9

        m=(7-n)/2;   //空格和\*的总和是7,左右各一半

**for**(k=0;k<m;k++)  //输出空格，注意第一行是0，所以k只能从0开始

        {

**printf**(" ");

         }

**for**(j=1;j<=n;j++)

        {

**printf**("\*");

         }

**for**(x=0;x<m;x++)  //输出空格对称的空格

        {

**printf**(" ");

         }

**printf**("\n");  // 一行结束了要换行

    }

**for**(i=5;i<=7;i++)   //处理下半部分

    {

        n=2\*i-7;

        m=(7-n)/2;

**for**(k=0;k<m;k++)  //以下格式跟上述差不多

        {

**printf**(" ");

         }

**for**(j=1;j<=n;j++)

        {

**printf**("\*");

         }

**for**(x=0;x<m;x++)

        {

**printf**(" ");

         }

**printf**("\n");

     }

**return** 0;

}

Jb的方法：

#include <stdio.h>

int main()

{

int i,j,k=0,s=7; //k控制空格，s控制\*

for(i=1;i<=7;i++)

{

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

printf(" ");

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

printf("\*");

printf("\n");

if(i<4)

{

k++;

s=s-2;

}

else

{

k--;

s=s+2;

}

}

return 0;

}

整数转成字符串（要记住的套路）

//用递归法将一个整数n转换成字符串。例如，输入483，应输出字符串“483”。N的位数不确定，可以是任意位数的整数

 #include <stdio.h>

**void** change(**int** n)   //函数的形参n 是需要转换成字符串的整数。

{

**int** i;

    i=n/10;     //声明一个变量i，将n整除10，获取最高位数字。

**if**(i!=0) change(i);  //判断i是否为0，如果不是则递归调用函数。如果是0，该数是一个一位数。

**putchar**(n%10+'0');  //输出最低位的数字，其中'0'是 ASCII码中表示数字0的字符，即字面值为 48。

    /\*putchar和getchar逐个输出输入字符数组的函数：name[i]=getchar();putchar(name[i]);

    scanf("%c",&name[i]);printf("%c",name[i]) 注意是%c   \*/

**putchar**(' ');    //最后输出一个空格，使输出的数宇之间有间隔

}

/\*因为递归的实现方式，每一次递归的输出都是从最高位开始的，而每次输出一个数字后都会输出一个

空格，所以最终的输出序列与原整数相反。 \*/

**int** main()

{

**int** n;

**scanf**("%d",&n);

    change(n);

**return** 0;

}

期末中档难度

/\*输入一个三行四列的矩阵，求出每一行和每一列中的最大值，将这些最大值中的最小值输出。

例如，矩阵

13 12 45 31

11 16 32  -9

10 58 31 29

行和列的最大值分别为 45,32,58,13,58,45,31

则输出为 13\*/

思路：构造一个大的数组把每行每列的最大值都塞到最外面，再比较

#include <stdio.h>

**int** main()

{

**int** i,j,min1,min2,max1,max2;

**int** a[4][5];

**for**(i=0;i<3;i++)

**for**(j=0;j<4;j++)

**scanf**("%d",&a[i][j]); //输入矩阵

**for**(i=0;i<3;i++)

    {

        max1=a[i][0]; //注意这个max1放置的位置

**for**(j=1;j<3;j++)

        {

**if**(max1<a[i][j])

            max1=a[i][j];

        }

        a[i][4]=max1;

    }

**for**(j=0;j<4;j++)

    {

        max2=a[0][j];

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

        {

**if**(max2<a[i][j])

            max2=a[i][j];

        }

        a[3][j]=max2;

}

min1=a[0][4];

**for**(i=0;i<3;i++)

    {

**if**(a[i][4]<min1)

        min1=a[i][4];

}

    min2=a[3][0];

**for**(j=0;j<4;j++)

    {

**if**(a[3][j]<min2)

        min2=a[3][j];

}

**if**(min1>min2) **printf**("%d",min2);

**else** **printf**("%d",min1);

**return** 0;

}

方法2：

1.设定一维数组存储每行每列的最大值

2. 采用类似选择排序的思路寻找最大值或最小值

|  |
| --- |
| 折半查找  /\*有n个数（n<20）,已按从大到小顺序存放在一个数组中，输入一个数，要求用折半查找法找出该数是数组中的第几个元素的值  （从1开始）。如果不在数组中输出0。  Input：  第一行为数组元素的个数n。  第二行是n个数组元素的值，整数。  第三行是要查找的值。  Output：  查找的值在数组中的位置（从序号1开始），找不到时显示0。  【输入输出样例】  输入：  10  10 9 8 7 6 5 4 3 2 1  5  输出:  6    思路：  （1）数列必须先排序；（这个题目已经排好了）  （2）找到区间中点，比大小，进而循环迭代区间边界  （3）循环终止条件：区间右边界值大于左边界\*/    #include <stdio.h>  **int** main()  {  **int** n,a[100],find,left,right,mid; // 数组长度、数组、要查的数、折半查找所需变量  **int** i; // 循环计数器  **int** pos; // 查找结果    **scanf**("%d", &n);  **for** (i=0;i<n;i++)  **scanf**("%d",&a[i]);  **scanf**("%d",&find);        // 折半查找      left=0;      right=n-1;      pos=0; // 初始值为0，表示没有找到  **while**(left<=right)      {          mid=(left+right)/2; // 取中间位置  **if**(a[mid]==find)          { // 找到了              pos=mid+1;  **break**;          }  **else** **if**(a[mid]>find)          { // 在左半边继续查找              left=mid+1;          }  **else**          { // 在右半边继续查找              right=mid-1;          }      }        // 输出结果  **printf**("%d\n",pos);    **return** 0;  } |

窗体顶端

窗体底端

窗体顶端

窗体底端

易错点：

#include <stdio.h>

#include <stdlib.h>

int main()

{

    float x,y;

    scanf("%f",&x);

    if(x<1)

        y=x;

    else if(1<=x<10) //只能写成x>=1&&x<=10

        y=2\*x-1;

    else y=3\*x-11;

    printf("%f",y);

}

#include <stdio.h>

#include <stdlib.h>

#define PI 3.14159

int main()

{

float v,r;

scanf("%f",&r);

v=4/3\*PI\*r\*r\*r; //斜杠是整除，如果要算出小数，应该用4.0/3

printf("%6.2f\n",v); //保留两位小数，加上小数一共六位

return 0;

}

#include <stdio.h> //单个字符要用’单引号A’不能用”A”

#include <stdlib.h>

int main()

{ char a;

  int b;

  printf("请输入成绩：");

  scanf("%d",&a);

  if(a<60)

    a="E";

  else  b=a/10;

  switch(b)

......

注意 等于是==

Flag在循环里面？外面？

/\*请编写一个用户自定义函数 int searchletter(char\*str)，统计字符 str中的大写英文字母的个数函数的返回值为这个字符串 str 中大写英文字母的个数。(同时请写一个主函数调用这个searchletter函数)\*/

#include <stdio.h>

#include <string.h>

int searchletter(char \*str)

{

int len,sum=0,i;

len=strlen(str);

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

{

if(\*(str+i)>='A'&&\*(str+i)<='Z') sum=sum+1;

}

return sum;

}

int main()

{

int i,sum;

char \*p,str[1000];

p=str;

scanf("%s",str);

sum=searchletter(p);

printf("%d",sum);

return 0;

}

#include <stdio.h>

int gbs(int m,int n)

{

int t,r;

int m0,n0;

m0=m;

n0=n;

if(m<n)

{

t=m;

m=n;

n=t;

}

r=m%n;

while(r!=0)

{

m=n;

n=r;

}

return m0\*n0/n;

}

int main()

{

int i,a[5],m,n;

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

scanf("%d",&a[i]);

for(i=0;i<4;i++)

{

m=a[i];

n=a[i+1];

printf("%d",gbs(m,n));

printf("\n");

}

return 0;

}

计算pi

#include <stdio.h>

double pi(double e)

{

double k,sum;

int i;

sum=1;

k=1;

for(i=1;i<10000;i++)

{

k=k\*i\*1.0/(2\*i+1);

if(k<e) break;

else sum=sum+k;

}

return sum;

}

int main()

{

double e,sum;

scanf("%lf",&e);

sum=pi(e);

printf("%lf",sum\*2);

return 0;

}