**错题本**

[1： 4](#_Toc512864666)

[参考答案 4](#_Toc512864667)

[2. 6](#_Toc512864668)

[参考答案 6](#_Toc512864669)

[赵一凡的解法： 7](#_Toc512864670)

[3.钟面角度问题 8](#_Toc512864671)

[CSDN&自解1： 8](#_Toc512864672)

[近乎标准解法1： 9](#_Toc512864673)

[参考答案： 9](#_Toc512864674)

[4. 程序：计算本金和复利之和 11](#_Toc512864675)

[参考答案 11](#_Toc512864676)

[5. 13](#_Toc512864677)

[自己做： 13](#_Toc512864678)

[6. 14](#_Toc512864679)

[代码一： 15](#_Toc512864680)

[代码2——do while： 16](#_Toc512864681)

[代码3 ——while 17](#_Toc512864682)

[7. 19](#_Toc512864683)

[自解： 19](#_Toc512864684)

[8.（关于ascii码） 20](#_Toc512864685)

[自解 20](#_Toc512864686)

[9. 21](#_Toc512864687)

[自解&反思： 21](#_Toc512864688)

[10.汉诺塔问题 23](#_Toc512864689)

[课堂笔记： 23](#_Toc512864690)

[参考答案： 24](#_Toc512864691)

[11. 25](#_Toc512864692)

[错误解法1： 25](#_Toc512864693)

[标答 26](#_Toc512864694)

[12.（不使用函数计算阶层） 27](#_Toc512864695)

[标答： 27](#_Toc512864696)

[13. 身高预测 28](#_Toc512864697)

[解析： 28](#_Toc512864698)

[14.【自例】利用goto达成循环 30](#_Toc512864699)

[自解： 30](#_Toc512864700)

[15.利用简单的递归来反向输出字符串（非指针） 32](#_Toc512864701)

[解答1（对scanf的深一步理解）//来源网络： 32](#_Toc512864702)

[解法2&做完18后的体悟！！ 33](#_Toc512864703)

[错解： 33](#_Toc512864704)

[思路： 34](#_Toc512864705)

[错解修正： 34](#_Toc512864706)

[结论&待解： 35](#_Toc512864707)

[16：复杂的汉诺塔问题 36](#_Toc512864708)

[17. 用递归法将一个整数n转换成字符串 39](#_Toc512864709)

[解答： 39](#_Toc512864710)

[18. 简单的递归反序输出正整数 40](#_Toc512864711)

[自解： 40](#_Toc512864712)

[体悟： 40](#_Toc512864713)

[错解1： 41](#_Toc512864714)

[错解 2： 41](#_Toc512864715)

[19.数组查找实例（浙大MOOC） 43](#_Toc512864716)

[注意：知识点 43](#_Toc512864717)

[笔记页：浙大指针 44](#_Toc512864718)

[20.在有序数列中插入数据 45](#_Toc512864719)

[CSDN解答： 45](#_Toc512864720)

[21. 分组排序 47](#_Toc512864721)

[参考答案 47](#_Toc512864722)

[//思维误区！ 48](#_Toc512864723)

[22.四舍五入 49](#_Toc512864724)

[自解答案： 49](#_Toc512864725)

[23.数据的顺时针排序 51](#_Toc512864726)

[24.常用小代码：找出一组数据中最大的数，并输出其下标 52](#_Toc512864727)

[25.进阶：同时输出最大和最小值 52](#_Toc512864728)

[代码优化（一次循环完成） 53](#_Toc512864729)

[23.连续读入数据的小段代码举例 54](#_Toc512864730)

[一：万能型，‘\n’为停止条件 54](#_Toc512864731)

[二：能用特征值作为停止条件 54](#_Toc512864732)

[24. 用数组完成解密工作 55](#_Toc512864733)

[CSDN1：完成hello,world!没毛病 55](#_Toc512864734)

[参考答案： 56](#_Toc512864735)

[25.判断元音出现的次数 58](#_Toc512864736)

[答案： 58](#_Toc512864737)

[26.cy问的一个题 60](#_Toc512864738)

[自解： 60](#_Toc512864739)

[非常精简的标答： 61](#_Toc512864740)

[27. 检查数中重复出现的数字 62](#_Toc512864741)

[自解： 62](#_Toc512864742)

[体悟 63](#_Toc512864743)

[28. 求并输出3~6位自方幂数 64](#_Toc512864744)

[参考答案 64](#_Toc512864745)

[29. 二维数组达成闰年问题 66](#_Toc512864746)

[答案 66](#_Toc512864747)

[30.思考：出现了意料之外的数字大概就是超出数组范围了 68](#_Toc512864748)

[解： 68](#_Toc512864749)

[31.指针入门 69](#_Toc512864750)

[参考答案 69](#_Toc512864751)

[32.指针法倒序输出输入的字符串 70](#_Toc512864752)

[C语言论坛答案： 70](#_Toc512864753)

[33. 指针的循环控制条件 71](#_Toc512864754)

[法一： 71](#_Toc512864755)

[法二： 71](#_Toc512864756)

[解析 72](#_Toc512864757)

[34. 字符串：数组名相当于个const指针 73](#_Toc512864758)

[错解： 74](#_Toc512864759)

[正解： 74](#_Toc512864760)

[35. 编程判断输入的一个字符串是否是“回文” 76](#_Toc512864761)

[代码寄存： 76](#_Toc512864762)

[36.自制的字符连接函数 77](#_Toc512864763)

[代码寄存： 77](#_Toc512864764)

1：编写一个程序，输入年份和月份，判断该年是否是闰年，并根据给出的月份判断是什么季节和该月有多少天？（闰年的条件是年份能被4整除但不能被100整除，或者能被400整除；规定3~5月为春季，6~8月为夏季，9~11月为秋季，1、2和12月为冬季）。

\*\*输入格式要求："%d,%d" 提示信息："Please enter year,month:"

\*\*输出格式要求："%d is leap year\n" "%d is not leap year\n" "The season is spring\n" "The season is summer\n" "The season is autumn\n" "The season is winter\n" "The number of days of this month is %d\n"

程序运行示例如下：

实例1：

Please enter year,month:2012,11

2012 is leap year

The season is autumn

The number of days of this month is 30

实例2：

Please enter year,month:2013,12

2013 is not leap year

The season is winter

The number of days of this month is 31

## 参考答案

|  |
| --- |
| #include <stdio.h>  **int** main()  {  **int** y, m, leap, season, days;  **printf**("Please enter year,month:");  **scanf**("%d,%d", &y, &m);       //1  **if** ((y % 4 == 0 && y % 100 != 0) || (y % 400 == 0))   //2          leap = 1;  **else**          leap = 0;  **if** (m >= 3 && m <= 5)  //1          season = 1;  **else** **if** (m >= 6 && m <= 8)          season = 2;  **else** **if** (m >= 9 && m <= 11)          season = 3;  **else**          season = 4;  **switch** (m)  //1      {  **case** 1:  **case** 3:  **case** 5:  **case** 7:  **case** 8:  **case** 10:  **case** 12:          days = 31;  **break**;             //1  **case** 4:  **case** 6:  **case** 9:  **case** 11:          days = 30;  **break**;  **case** 2:  **if** (leap == 1)              days = 29;  **else**              days = 28;      }  **printf**("%d %s leap year\n", y, (leap == 1 ? "is" : "is not"));  **printf**("The season is ");  **switch** (season)    //1      {  **case** 1:  **printf**("spring\n");  **break**;  **case** 2:  **printf**("summer\n");  **break**;  **case** 3:  **printf**("autumn\n");  **break**;  **case** 4:  **printf**("winter\n");      }  **printf**("The number of days of this month is %d\n", days);  **return** 0;  } |

2.编写一个程序打印如下对称图形（行数由键盘输入1-9范围的值），例如下面是输入数字4时的情形:

首先提示输入："Enter n(1-9):\n"

输入格式："%d"

然后输出结果，例如：

Enter n(1-9):

4↙

4444444

33333

222

1

222

33333

4444444

## 参考答案

|  |
| --- |
| #include <stdio.h>    **int** main()  {  **int** i, j, k;  **int** n;                  /\*定义变量n，表示行数\*/    **printf**("Enter n(1-9):\n");  **scanf**("%d", &n);  **for** (i = 0; i < 2 \* n - 1; i++)//限制行      {          k = i;  **if** (i >= n)          {              k = 2 \* n - 2 - i;//下半漏斗的函数关系。    /\*当i >=n时，令k=2\*n-2-i，打印相应对称图形\*/          }  **for** (j = 0; j < k; j++)//列小于行时打印空格——后面没有空格          {  **printf**(" ");                /\*打印空格\*/          }  **for** (j = 0; j < 2 \* (n - k) - 1; j++)          {  **printf**("%d", n - k);            /\*打印数字\*/          }  **printf**("\n");      }    } |

## 赵一凡的解法：

#include <stdio.h>

#include <stdlib.h>

int main()

{

int n,i,m,j,k;

printf("Enter n(1-9):\n");

scanf("%d",&n);

for(i=1;i<=2\*n-1;i++)

{

if(i<=n)

{

for(m=1;m<=2\*n-i;m++)

{

if(m<=i-1)

printf(" ");

else

printf("%d",n+1-i);

}

printf("\n");

}

if (i>n)

{

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

{

if(j<2\*n-i)

printf(" ");

else

printf("%d",i-n+1);

}

printf("\n");

}

}

return 0;

}

# **3.钟面角度问题**

背景：钟面上的时针和分针之间的夹角总是在 0 度~ 359 度之间。举例来说，在十二点的时候两针之间的夹角为 0 度，而在六点的时候夹角为 180 度，在三点的时候为 90 度。本题要解决的是计算 0:00 到 12:00之间任意一个时间的夹角。

输入：要求输入必须为 小时h分钟m的格式，如12h00m, 5h43m等，其他形式的输入都不接收，如 12,00 或5h43等输入都不被接收，而且需要对“小时”和“分钟”进行数值有效判断（小时在[0,12],分钟在[0.60]之间），不满足以上要求时，需要重新输入。

在程序中首先打印：Please input time(e.g: 5h43m)，然后输入时间。

格式为：

"Please input time(e.g: 5h43m)\n"

"%d%c%d%c"

输出：对应每组测试数据，用常用格式显示时间以及这个时候时针和分针间的最小夹角，精确到小数点后一位。

格式为：

"At %d:%02d the angle is %.1f degrees.\n"

如：

输入“12h00m”时，输出 At 12:00 the angle is 0.0 degrees.

输入“5h43m” 时，输出 At 5:43 the angle is 86.5 degrees.

输入“5h5m” 时，输出 At 5:05 the angle is 122.5 degrees.

提示1：以表中心到12点的连线为基准，分针每走1分钟是6度，时针与基准的夹角每个小时也是30度，从整点开始，每过1分钟时针再增加0.5度。要求结果角度为正值，即最终要取绝对值（fabs函数）

提示2：二者之间角度不应大于180度，如果大于，应用360度减去该角度。

## CSDN&自解1：

#include <stdio.h>

#include <math.h>

int main()

{

int h,m;

float t,a,b;

char c1,c2;

printf("Please input time(e.g: 5h43m)\n");

scanf("%d%c%d%c",&h,&c1,&m,&c2 ); //输入时分

a=(h+(float)m/60)\*30.0; //时分融合

b=m\*6.0;

t=fabs(a-b);

if(t>180)

t=360.0-t;

printf("At %d:%.2d the angle is %.1f degrees.\n",h,m,t);

return 0;

}

//这个解法没有进行输入的数据的判定！！

## 近乎标准解法1：

#include <stdio.h>

#include <math.h>

void main()

{

    int h,m,i=0;

    float alpha=0;

    printf("Please input time(e.g: 5h43m)\n");

        scanf("%dh%dm",&h,&m);

    for(;i==0;)

    {

        if(h<12 && h>0 && m>0 && m<60)

        {    i=1;

            alpha=fabs(30\*h-5.5\*m);

            if(m<10)

            printf("At %d:0%d the angle is %.1f degrees.",h,m,alpha);

            else

            printf("At %d:%d the angle is %.1f degrees.",h,m,alpha);

        }

        else

        {

            printf("Please input time(e.g: 5h43m)");

            scanf("%dh%dm",&h,&m);

        }

}

}

## 参考答案：

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

**int** main()

{

**int** hour, min;

**int** ret;

**char** p, q;

**double** angle;

**do**

    {

        hour = -1;

        min = -1;

        p = 'a';

        q = 'a';

**printf**("Please input time(e.g: 5h43m)\n");

        ret = **scanf**("%d%c%d%c", &hour, &p, &min, &q);//2

        //getchar();

    }

**while** (ret != 4 || p != 'h' || q != 'm' || (hour > 12 || hour < 0) || (min > 60 || min < 0));//4

    angle = **fabs**(5.5 \* min - 30 \* hour);//2

**if** (angle > 180)

    {

        angle = 360 - angle;//2

    }

**printf**("At %d:%02d the angle is %.1f degrees.\n", hour, min, angle);

}

# **4.** 程序：计算本金和复利之和

编写程序打印出一个表格，这个表格显示了在几年时间内100美金投资在不通利率上的价值。用户将输入利率和要投资的年数。假设整合利息一年一次，表格将显示出一年间在此输入利率下和后边四个更高利率下投资的价值。

\*\*输入格式要求："%d" 提示信息："Enter interest rate:" "Enter number of year:"

\*\*输出格式要求："\nyears" "%6d%%" "%3d " "%7.2f"

下面是程序运行时的情况：

Enter intereset rate: 6

Enter number of years: 5

years 6% 7% 8% 9% 10%

1 106.00 107.00 108.00 109.00 110.00

2 112.36 114.49 116.64 118.81 121.00

3 119.10 122.50 125.97 129.50 133.10

4 126.25 131.08 136.05 141.16 146.41

5 133.82 140.26 146.93 153.86 161.05

## 参考答案

|  |
| --- |
| #include <stdio.h>    #define NUM\_RATES (sizeof(value) / sizeof(value[0]))  #define INITIAL\_BALANCE 100.00    **int** main()  {  **int** i, low\_rate, num\_years, year;  **float** value[5];    **printf**("Enter interest rate:");  **scanf**("%d", &low\_rate);  **printf**("Enter number of year:");  **scanf**("%d", &num\_years);    **printf**("\nyears");  **for** (i = 0; i < NUM\_RATES; i++)      {  **printf**("%6d%%", low\_rate + i);          value[i] = INITIAL\_BALANCE;      }  **printf**("\n");    **for** (year = 1; year <= num\_years; year++)      {  **printf**("%3d   ", year);  **for** (i = 0; i < NUM\_RATES; i++)          {              value[i] += (low\_rate + i) / 100.0 \* value[i];  **printf**("%7.2f", value[i]);          }  **printf**("\n");      }    **return** 0;  } |

5. 程序：确定消息的长度。

为了说明字符的读取方式，下面编写一个程序来计算消息的长度。在用户输入消息后，程序显示的长度如下：

Enter a message: Brevity is the soul of wit.

Your message was 27 character(s) long.

消息的长度包括空格和标点符号，但是不包括消息结尾处的换行符。

\*\*输入格式要求：提示信息："Enter a message:"

\*\*输出格式要求："Your message was %d character(s) long.\n"

|  |
| --- |
|  |

## 自己做：

#include <stdio.h>

#include <stdlib.h>

int main()

{

int counter=0;

printf("Enter a message:");

while (getchar()!='\n')

{

counter++;

}

printf("Your message was %d character(s) long.\n",counter);

return 0;

}

//直接输入就行，没必要非要scanf

6. 检测输入数据中奇数和偶数的个数

从键盘输入一系列正整数，输入-1表示输入结束（-1本身不是输入的数据）。编写程序判断输入数据中奇数和偶数的个数。如果用户输入的第一个数据就是-1，则程序输出"over!"。否则。用户每输入一个数据，输出该数据是奇数还是偶数，直到用户输入-1为止，分别统计用户输入数据中奇数和偶数的个数。

程序运行结果示例1：

Please enter the number:

1↙

1:odd

5↙

5:odd

8↙

8:even

9↙

9:odd

12↙

12:even

17↙

17:odd

-1↙

The total number of odd is 4

The total number of even is 2

程序运行结果示例2：

Please enter the number:

-1↙

over!

The total number of odd is 0

The total number of even is 0

输入格式: "%d"

输出格式：

输入提示信息："Please enter the number:\n"

用户输入的第一个数据就是-1，输出格式："over!\n"

奇数的输出格式："%d:odd\n"

偶数的输出格式："%d:even\n"

输入数据中奇数的个数统计："The total number of odd is %d\n"

输入数据中偶数的个数统计："The total number of even is %d\n"

## 代码一：

#include <stdio.h>

#include <stdlib.h>

**int** main()

{

**int** a,o=0,e=0,b;

printf("Please enter the number:**\n**");

scanf("%d",&a);

**if** (a==-1)

{

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

printf("The total number of odd is 0**\n**The total number of even is 0**\n**");

**return** 0;

}

**else**

{

**if** (a%2==0)

{

printf("%d:even**\n**",a);

e++;

}

**else**

{

printf("%d:odd**\n**",a);

o++;

}

**while** (1)

{

scanf("%d",&b);

**if** (b==-1)

**break**;

**if** (b%2==0)

{

printf("%d:even**\n**",b);

e++;

}

**else**

{

printf("%d:odd**\n**",b);

o++;

}

}

}

printf("The total number of odd is %d**\n**",o);

printf("The total number of even is %d**\n**",e);

**return** 0;

}

## 代码2——do while：

#include <stdio.h>

#include <stdlib.h>

int main()

{

int a,o=0,e=0;

printf("Please enter the number:\n");

do{

scanf("%d",&a);

if (a%2==0)

{

printf("%d:even\n",a);

e++;

}

if (a%2==1)

{

printf("%d:odd\n",a);

o++;

}

} while(a!=-1);

if(a==-1) printf("over!\n");//这一行理论上有问题，它应该会导致最后多一个OVER，但是在我的CB里没表现出来。

printf("The total number of odd is %d\n",o);

printf("The total number of even is %d\n",e);

return 0;

}

## 代码3 ——while

#include <stdio.h>  
  
#include <stdlib.h>  
  
  
  
int main()  
  
{  
  
    int a=1,o=0,e=0;  
  
    printf("Please enter the number:\n");  
  
    //do  
    while (a!=-1)  
        {  
  
        scanf("%d",&a);  
  
        if (a%2==0)  
  
            {  
  
                printf("%d:even\n",a);  
  
                e++;  
  
            }  
  
        if (a%2==1)  
  
            {  
  
                printf("%d:odd\n",a);  
  
                o++;  
  
            }  
  
    }  
    //while(a!=-1);  
  
    if(a==-1) printf("over!\n");  
  
    printf("The total number of odd is %d\n",o);  
  
    printf("The total number of even is %d\n",e);  
  
    return 0;  
  
}

7. 两个正整数的最大公约数是能够整除这两个整数的最大整数。

请按如下函数原型编写计算最大公约数的函数Gcd()。

int Gcd(int a, int b);

在主函数中从键盘任意输入两个整数,调用函数Gcd()计算这两个数的最大公约数返回给主函数，然后在主函数中输出其最大公约数。

\*\*输入提示信息: "Input a,b:\n"

\*\*输入格式: "%d,%d"

\*\*输出格式:

如果a或b不是正整数，输出: "Input number should be positive!\n"

否则输出: "Greatest Common Divisor of %d and %d is %d\n"

注：（1）不能使用指针、结构体、共用体、文件、goto、枚举类型进行编程。

（2）用纯C语言编程，所有变量必须在第一条可执行语句前定义。

## 自解：

#include <stdio.h>

#include <stdlib.h>

int Gcd(int a, int b);

int main()

{

int m,n,v;

printf("Input a,b:\n");

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

if (m!=(int )m||n!=(int )n||m<=0||n<=0)//判断是否为整数的方法，很好吧？

printf("Input number should be positive!\n");

else{

v=Gcd(m,n);

printf("Greatest Common Divisor of %d and %d is %d\n",m,n,v);}

return 0;

}

int Gcd(int a, int b)

{

int i,q;

q=a>b?b:a;//将两个正整数中较小的数赋给q

for(i=q;i>0&&(b%i||a%i);i--);//遍历从q到0的所有数，如果能同时被两个正整数整除，则为两个数的最大公约数

return i;

}

8.（关于ascii码） 通过键盘输入字符，将输入的字符分为控制、数字、大写字母、小写字母和其他字符等五类。

\*\*输入格式要求："%c" 提示信息："Enter a charactor:"

\*\*输出格式要求："\nThe charactor is a control charactor.\n"

"\nThe charactor is a digit charactor.\n" "\nThe charactor is a capital charactor.\n"

"\nThe charactor is a lower charactor.\n" "\nThe charactor is a other charactor.\n"

程序运行示例1如下：

Enter a charactor:1

The charactor is a digit charactor.

程序运行示例2如下：

Enter a charactor:a

The charactor is a lower charactor.

自解：

void main()

{

char a;

printf("Enter a charactor:");

scanf("%c",&a);

if(a>=65&&a<=90)

printf("\nThe charactor is a capital charactor.\n" );

else if(a>=97&&a<=122)

printf("\nThe charactor is a lower charactor.\n" );

else if(a>=48&&a<=57)

printf("\nThe charactor is a digit charactor.\n");

else if ((a>=0&&a<=32)||a==127)

printf("\nThe charactor is a control charactor.\n");

else

printf("\nThe charactor is a other charactor.\n");

}

9. 如果一个正整数（大于1）等于其各个数字的立方和，则该数称为阿姆斯特朗数（亦称为自恋性数）。如407=4^3+0^3+7^3就是一个阿姆斯特朗数。试编程求1000以内的所有阿姆斯特朗数。

\*\*输出格式要求："%d "

\*\*输出信息：

"There are following Armstrong number smaller than 1000:\n"

程序运行示例如下：

There are following Armstrong number smaller than 1000:

153 370 371 407

## 自解&反思：

#include <stdio.h>

#include <stdlib.h>

int main()

{

printf("There are following Armstrong number smaller than 1000:\n");

int a,b,c,n,i;

for (n>1;n<1000;n++)

{

i=n;

a=i/100;

c=i%10;

b=(i/10)%10; //头一次用的是b=i-100\*a-c,但是这样不对的，脑抽了。

if (i==a\*a\*a+b\*b\*b+c\*c\*c)// pow函数的局限性，如果用了pow，它会失精，除非全部用double，所以简单的情况下完全可以不用Pow

printf("%d ",i);

}

return 0;

}

网上的启发性解法：

//有意思在我是从2枚举到999，它是用的每一位的组合。当然，我效率其实还高一些……

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

int main()

{

int x,i,j,k;

printf("There are following Armstrong number smaller than 1000:\n") ;

for(i=1;i<10;i++)

{

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

{

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

{

x=100\*i+10\*j+k;

if(x==pow(i,3)+pow(j,3)+pow(k,3))//他用的Pow ,会掉一个407.

{

printf("%d ",x);

}

}

}

}

return 0;

}

# 10.汉诺塔问题

编程求解汉诺塔问题。

汉诺塔（Hanoi）是必须用递归方法才能解决的经典问题。它来自于印度神话。上帝创造世界时作了三根金刚石柱子，在第一根柱子上从下往上按大小顺序摞着64片黄金圆盘，如图7-3所示。上帝命令婆罗门把圆盘从下面开始按大小顺序重新摆放到第二根柱子上，并且规定，每次只能移动一个圆盘，在小圆盘上不能放大圆盘。有人预言说，这件事完成时宇宙会在一瞬间闪电式毁灭，也有人相信婆罗门至今仍在一刻不停地搬动着圆盘。

\*\*输入格式要求："%d" 提示信息："Input the number of disks:"

\*\*输出格式要求："Steps of moving %d disks from A to B by means of C:\n" "Move %d: from %c to %c\n"

程序运行示例如下：

Input the number of disks:3

Steps of moving 3 disks from A to B by means of C:

Move 1: from A to B

Move 2: from A to C

Move 1: from B to C

Move 3: from A to B

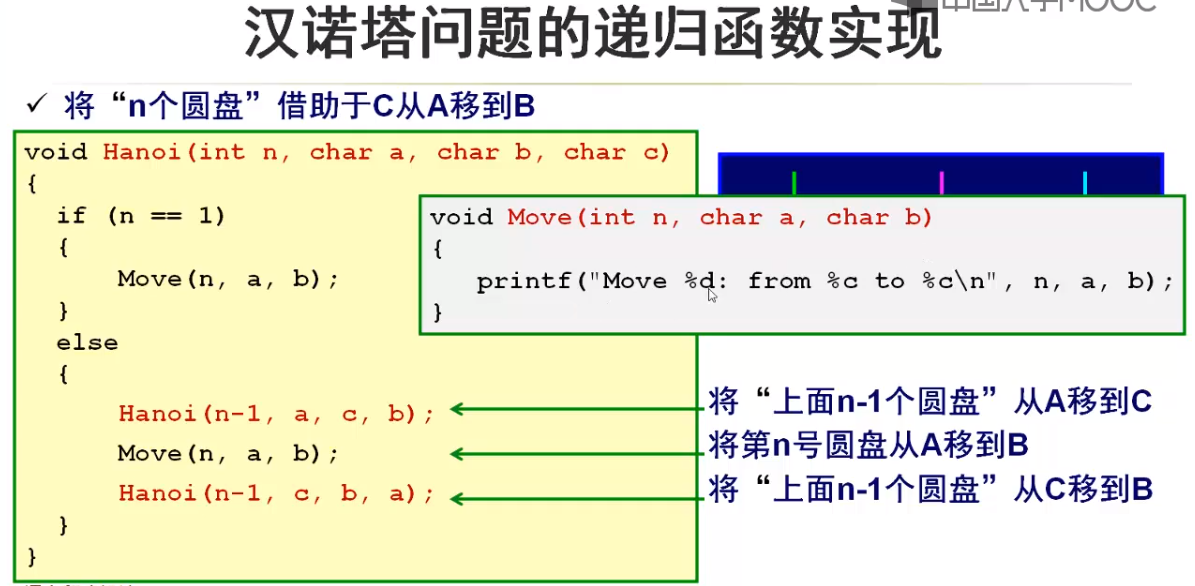
Move 1: from C to A

Move 2: from C to B

Move 1: from A to B

## 课堂笔记：

思想：将n-1看成一个整体，把n个盘的问题简化成两个盘的问题。



## 参考答案：

|  |
| --- |
| #include <stdio.h>  **void** Hanoi(**int** n, **char** a, **char** b, **char** c);  **void** Move(**int** n, **char** a, **char** b);  **int** main()  {  **int** n;  **printf**("Input the number of disks:");  **scanf**("%d", &n);  **printf**("Steps of moving %d disks from A to B by means of C:\n", n);      Hanoi(n, 'A', 'B', 'C'); /\*调用递归函数Hanoi()将n个圆盘借助于C由A移动到B\*/  **return** 0;  }  /\* 函数功能：用递归方法将n个圆盘借助于柱子c从源柱子a移动到目标柱子b上 \*/  **void** Hanoi(**int** n, **char** a, **char** b, **char** c)  {  **if** (n == 1)      {          Move(n, a, b);       /\* 将第n个圆盘由a移到b \*/      }  **else**      {          Hanoi(n - 1, a, c, b); /\* 递归调用Hanoi()，将第n-1个圆盘借助于b由a移动到c\*/          Move(n, a, b);       /\* 第n个圆盘由a移到b \*/          Hanoi(n - 1, c, b, a); /\*递归调用Hanoi()，将第n-1个圆盘借助于a由c移动到b\*/      }  }  /\* 函数功能：  将第n个圆盘从源柱子a移到目标柱子b上 \*/  **void** Move(**int** n, **char** a, **char** b)  {  **printf**("Move %d: from %c to %c\n", n, a, b);  } |

11. 题目：一个正整数，它加上100后是一个完全平方数，再加上168又是一个完全平方数，请按从小到大的顺序，连续输出3个满足这样条件的数？

程序分析：使用穷举法，如果找到三个这样的数据，就停止。

提示：判断一个数是否为完全平方数，可以先将该数开方，在平方，如果结果与原数相等，即该数为完全平方数。

要求输入，每行一个满足条件的数，例如：

结果1

结果2

结果3

## 错误解法1：

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

#define eps 1e-10

int n=1;

int main()

{

int i,b,a=1;

for(i=1;i<10000000;i++)

{

a=i+100;

b=a+168;

if (n<4)

{if ((sqrt(a\*a)-a)<eps&&(sqrt(b\*b)-b)<eps)

{

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

n++;}

else

break;

}

}

}

//死于精度不够？？！

//必须先开方后平方。

## 标答

#include <stdio.h>

#include <math.h>

int main()

{

    int i, j;

    int a, b;

    for (i = 0;i<100000; i++)

    {

        a = i + 100;

        b = i + 168;

        if ((sqrt(a) == (int)sqrt(a)) && ((sqrt(b) == (int)sqrt(b))))

            printf("%d\n", i);

    }

    return 0;

}

12.（不使用函数计算阶层）

编程计算: 1!+3!+5!+...+(2n-1)!

\*\*\*\*输入提示信息格式: "Input n:\n"

\*\*\*\*输入数据格式要求: "%d"

\*\*\*\*输出数据格式要求: "sum=%ld\n"

注：请勿使用函数编写此题

## 标答：

#include <stdio.h>

void main()

{

int i,j,k,n;

long int sum=0;

printf("Input n:\n");

scanf("%d",&n);

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

{

k=1;

for(j=1;j<=(2\*i-1);j++)//计算（2\*n-1的阶乘）

{

k=k\*j;

}

sum+=k;

}

printf("sum=%ld\n",sum);

}

# 13. 身高预测

每个做父母的都关心自己孩子成人后的身高，据有关生理卫生知识与数理统计分析表明，影响小孩成人后的身高的因素包括遗传、饮食习惯与体育锻炼等。小孩成人后的身高与其父母的身高和自身的性别密切相关。

设faHeight为其父身高，moHeight为其母身高，为float类型，身高预测公式为

男性成人时身高 = (faHeight + moHeight) × 0.54 cm

女性成人时身高 = (faHeight × 0.923 + moHeight) / 2 cm

此外，如果喜爱体育锻炼，那么可增加身高2%；如果有良好的卫生饮食习惯，那么可增加身高1.5%。

请编程从键盘输入用户的性别（用字符型变量sex存储，输入字符F表示女性，输入字符M表示男性）、父母身高（用实型变量存储，faHeight为其父身高，moHeight为其母身高）、是否喜爱体育锻炼（用字符型变量sports存储，输入字符Y表示喜爱，输入字符N表示不喜爱）、是否有良好的饮食习惯等条件（用字符型变量diet存储，输入字符Y表示良好，输入字符N表示不好），利用给定公式和身高预测方法对身高进行预测。

运行示例：

Are you a boy(M) or a girl(F)?F↙

Please input your father's height(cm):182↙

Please input your mother's height(cm):162↙

Do you like sports(Y/N)?N↙

Do you have a good habit of diet(Y/N)?Y↙

Your future height will be 167(cm)

## 解析：

#include <stdio.h>

#include <stdlib.h>

int main()

{

char sex; /\*孩子的性别\*/

char sports; /\*是否喜欢体育运动\*/

char diet; /\*是否有良好的饮食习惯\*/

float myHeight; /\*孩子身高\*/

float faHeight; /\*父亲身高\*/

float moHeight; /\*母亲身高\*/

printf("你是男孩(M)还是女孩(F):");

scanf(" %c", &sex); /\*在前加一个空格，将存于缓冲区的回车符读入\*/

printf("你爸多高(cm):");

scanf("%f", &faHeight);

printf("你妈多高(cm):");

scanf("%f",&moHeight);

printf("你喜欢运动吗(Y/N)?");

scanf(" %c", &sports); /\*在前加一个空格，将存于缓冲区的回车符读入\*/

printf("你有一个良好的生活习惯吗(Y/N)?");

scanf(" %c", &diet); /\*在前加一个空格，将存于缓冲区的回车符读入\*/

if (sex=='M'||sex=='m')//输入‘Y’和’N’完全可以不定义，直接当字符输入的。

myHeight = (faHeight + moHeight) \* 0.54;

else

myHeight = (faHeight \* 0.923 + moHeight)/2.0;

if (sports =='Y'||sports=='y')

myHeight = myHeight \* ( 1 + 0.02);

if (diet =='Y'||diet=='y')

myHeight = myHeight \* (1 + 0.015);

printf("你将来的身高是 %.0f(cm)\n",myHeight); //自悟，在不强行整形化的前提下保留整数。

}

# 14.【自例】利用goto达成循环

用switch语句编程设计一个简单的计算器程序，要求根据用户从键盘输入的表达式：

操作数1 运算符op 操作数2

计算表达式的值，指定的算术运算符为加（+）、减（-）、乘（\*）、除（/）。

要求如下：

（1）要求程序能进行浮点数的算术运算，操作数1和2用float类型定义，运算符op用字符型定义。

（2）要求输入的算术表达式中的操作数和运算符之间可以加入任意多个空白符

\*\*\*输入提示信息\*\*\*："Please enter the expression:\n"

\*\*\*输入数据格式\*\*\*："%f %c%f"

\*\*\*输出数据格式\*\*\*：以加运算为例："%f + %f = %f\n"

（3）要求连续做多次算术运算，每次运算结束后，程序都给出提示：

Do you want to continue(Y/N or y/n)?

用户输入Y或y时，程序继续进行其他算术运算；否则程序退出运行状态。

\*\*\*输入提示信息\*\*\*："Do you want to continue(Y/N or y/n)?"

\*\*\*输入数据格式\*\*\*：" %c"

（4）如果输入错误的运算符，则输出"Unknown operator!\n"，不结束程序运行，继续询问是否做下一次运算。

(5)如果除数为0，则输出"Division by zero!\n"，不结束程序运行，继续询问是否做下一次运算。

(6)允许浮点数的误差在1e-7范围内。

提示：请直接拷贝粘贴题目中给定的格式字符串。

## 自解：

#include <stdio.h>

#include <stdlib.h>

int main()

{

char op,answer;

float c1,c2,a;

START:printf("Please enter the expression:\n");

scanf("%f %c%f",&c1,&op,&c2);

switch (op)

{

case '+':

a=c1+c2;

printf("%f + %f = %f\n",c1,c2,a);

break;

case '-':

a= c1-c2;

printf("%f - %f = %f\n",c1,c2,a);

break;

case '\*':

a=c1\*c2;

printf("%f \* %f = %f\n",c1,c2,a);

break;

case '/':

if (c2==0)

{

printf("Division by zero!\n");

break;

}

else

{

a=c1/c2;

printf("%f / %f = %f\n",c1,c2,a);

break;

}

default:

printf("Unknown operator!\n");

}

printf("Do you want to continue(Y/N or y/n)?");

scanf(" %c",&answer);

if (answer=='Y'||answer=='y')

goto START;

return 0;

}

# 15.利用简单的递归来反向输出字符串（非指针）

编程将字符串s倒序输出，要求利用函数递归实现。

\*\*输入格式要求："%s" 提示信息："input your string:\n"

\*\*输出格式要求："%c"

程序运行的输入输出样例：

屏幕先输出提示信息：

input your string:

然后用户键盘输入：

abcdefg

最后屏幕输出：

gfedcba

## 解答1（对scanf的深一步理解）//来源网络：

#include<stdio.h>

void abc();

**int** main()

{

printf("input your string:\n");

abc();

printf("\n");

**return** 0;

}

**void** abc() *//就是它定义的这个函数的有点看不懂*

{

**char** c; *//定义字符形变量C*

scanf("%c",&c); *//读入C*

**if**(c!='\n') *//如果C不是换行*

{

abc();*//不是换行再调用abc，……然后这儿怎么达成的递归？//继续吃*

//理解有误！！scanf("%c",&c)在功能上和getchar()是一样的，都能吞掉一个字符。之前笔记中那个要“空格%c”是因为，前面输入了一个数据，缓冲区有个换行会导致读不到新输入的数据，当我们缓冲区里有一串字符的时候它是可以胜任的。

于是理解起来就很容易了：第一次输入了一大串字符，但是读取只能吃一个。如果吃的这个东西不是换行（输入需要敲一次回车，产生换行），我们就继续吃（绿注释），直到吃到换行为止；吃到换行后就能输出了，这样就完成了倒序输出。

printf("%c",c);

}

## 解法2&做完18后的体悟！！

#include<stdio.h>

void abc();

int main()

{

printf("input your string:\n");

abc();

printf("\n");

return 0;

}

void abc()

{

char b;

b=getchar();

if(b!='\n')

{

abc(b);

printf("%c",b);

}

}

//其实这个和解法1没区别？

那么观察错解才有收获！

### 错解：

#include<stdio.h>

void abc(char b);

int main()

{

char c;

printf("input your string:\n");

scanf("%c",&c);

abc(c);

printf("\n");

return 0;

}

void abc(char b)

{

b=getchar();

if(b!='\n')

{

abc(b);

printf("%c",b);

}

}

输入测试：

输入:abc

输出:cb

### 思路：

做完18后，我认为所有的递归，都能化成有形参的函数。所以我想把网上的解法改成读入字符，然后把字符在递归内消化的……（事实证明我学的知识不够）

我这个错解的错误原因在于，我主函数的scanf其实没有把我输入的所有的字符给吞掉，它和getchar相同，只吞了一个……所以其实并没输入所有的字符……getchar其实还是在读缓冲区……所以其实我儿在主函数里定义的c没有任何意义！！！！把scanf去掉就是对的！

### 错解修正：

#include<stdio.h>

void abc(char b);

int main()

{

char c;

printf("input your string:\n");

//删掉了scanf

abc(c);

printf("\n");

return 0;

}

void abc(char b)

{

b=getchar();

if(b!='\n')

{

abc(b);

printf("%c",b);

}

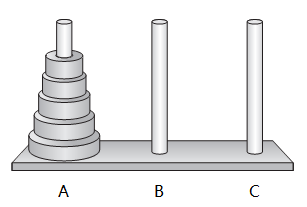
}

## 结论&待解：

在这种限制条件下我没法输入一整串字符给我的函数，以后或许可以吧！

# 16：复杂的汉诺塔问题

汉诺塔问题是一个著名的问题，初始模型如图所示。其来源据说是在约19世纪末欧洲的商店中出售一种智力玩具，在一块铜板上有三根杆，最左边的杆自上而下、由小到大顺序串着64个圆盘构成的塔，游戏的目的是将最左边A杆上的圆盘，借助最右边的C杆，全部移动到中间的B杆上，条件是一次仅能移动一个盘，且不允许大盘放在小盘的上面。



\*\*输入格式要求："%d" 提示信息："Please enter the number of discs:"

\*\*输出格式要求："\tTotal:%d\n"    "%2d-(%2d):%c==>%c\n"

程序运行示例如下：

Please enter the number of discs:

Please enter the number of discs: 5

 1-( 1):a==>b

 2-( 2):a==>c

 3-( 1):b==>c

 4-( 3):a==>b

 5-( 1):c==>a

 6-( 2):c==>b

 7-( 1):a==>b

 8-( 4):a==>c

 9-( 1):b==>c

10-( 2):b==>a

11-( 1):c==>a

12-( 3):b==>c

13-( 1):a==>b

14-( 2):a==>c

15-( 1):b==>c

16-( 5):a==>b

17-( 1):c==>a

18-( 2):c==>b

19-( 1):a==>b

20-( 3):c==>a

21-( 1):b==>c

22-( 2):b==>a

23-( 1):c==>a

24-( 4):c==>b

25-( 1):a==>b

26-( 2):a==>c

27-( 1):b==>c

28-( 3):a==>b

29-( 1):c==>a

30-( 2):c==>b

31-( 1):a==>b

    Total:31

# 17. 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。

\*\*输入提示："\n输入整数："

\*\*输入格式："%d"

\*\*输出提示："\n输出是："

\*\*输出转换的字符串

程序的运行示例如下：

输入整数：345

输出是：345

## 解答：

#include <stdio.h>

void int\_to\_char(int num)

{

if (num == 0) //字符型‘0’的asii码是48

printf("%c",48);//输出‘0’

if ((num /= 10) != 0)//如果num砍掉尾巴后还不等于0，并给num赋值它砍掉尾巴的数字

{

int\_to\_char(num);//继续调用num砍尾巴

printf("%c", num % 10 + 48);//输出尾巴

//输出放在函数后面，可正序输出

//printf一层一层累计, 按顺序执行，来达成正序输出.

}

}

int main()

{

int num = 0;

printf("\n输入整数：");

scanf("%d",&num);

printf("\n输出是：");

int\_to\_char(num\*10);//如果不乘10，会多输出一个0

return 0;

}

# 18. 简单的递归反序输出正整数

输入一个正整数，要求以相反的顺序输出该数。用递归方法实现。

\*\*输入格式要求："%d" 提示信息："Enter number:"

\*\*输出格式要求："%d"

程序运行示例如下：

Enter number:35567899

99876553

## 自解：

#include <stdio.h>

#include <stdlib.h>

void backput(int n);

int main()

{

int n;

printf("Enter number:");

scanf("%d",&n);

backput(n);

return 0;

}

void backput(int n)

{

if (n !=0)

{

printf("%d",n%10);

backput(n/10);

}

}

## 体悟：

我做这个的时候是想参照上次那个反序输出字符串玩儿的。

首先，递归好像一般用不上返回值？；然后，这次递归那个形参不能不要！T15那个位置他没要形参大概是因为输入的是字符，会一个一个读取，留在缓冲区，但是我们这个是数字， 会一次读走。要是不要形参，就会这样：

### 错解1：

#include <stdio.h>

#include <stdlib.h>

void backput();

int main()

{

printf("Enter number:");

backput();

return 0;

}

void backput()

{

int n; //每次都要Int一次n，但是没新的输入！第一次输入的n已经被读走了，所以输出结果就仅仅是个3

scanf("%d",&n);

if (n !=0)

{

printf("%d",n%10);

backput(n/10);

}

}

输出结果是只有一个3；

然后的体悟是，递归不用while。

虽然逻辑上是：直到……停止递归，但是其实使用If来层层嵌套逼近的，这题如果用了while，会直接死循环：

### 错解 2：

#include <stdio.h>

#include <stdlib.h>

void backput(int n);

int main()

{

int n;

printf("Enter number:");

scanf("%d",&n);

backput(n);

return 0;

}

void backput(int n)

{

while (n!=0) //死循环啦！！

{

printf("%d",n%10);

backput(n/10);

}

}

这道题的逻辑应该很是清晰：只要n不是0，我们就把它的尾数分离出来，然后输出，再调用。

# 19.数组查找实例（浙大MOOC）



这个ret=-1做初值很有意思，我觉得可以学一下。

## 注意：知识点

如果在函数中用到数组的大小，那么一般是要另起一个参数来定义数组的长度的。

# 笔记页：浙大指针

1. 对于指针变量\*p,p保存的是地址，如果printf(“%p,p”)，会输出地址。只有加了\*才会访问、输出原来的数据.
2. 修改1可知，如果修改\*p,就相当于修改了地址指向的值，可以达成修改。
3. 指针应用场景二b

函数返回运算的状态，结果通过指针返回

常用的套路是让函数返回特殊的不属于有效范围内的值来表示出错：

-1或0（在文件操作会看到大量的例子）

但是当任何数值都是有效的可能结果时，就得分开返回了

后续的语言（C++,Java）采用了异常机制来解决这个问题

4. \*P必须初始化，否则迟早会出错。

因为\*P=x 的机制是像\*P指向的内存中写入一个值，但是\*P要是没有初始化的话，，鬼晓得它指向的内存是什么地方，如果指向了不可写入的地方，那么就segmetation fault了……

5. 数组变量是const的指针，所以不能被赋值

* int a[] <==> int \* const a=….

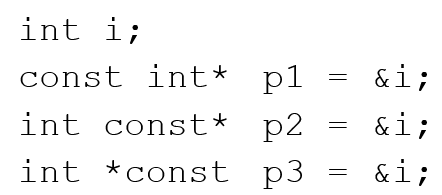
6. 数组就是特殊的指针。详见浙大9.1-4；传入函数时，数组传入的是首地址；

7. **所指是const：**

表示不能通过这个指针去修改那个变量（并不能使得那个变量成为const）

* + const int \*p = &i;
  + \*p = 26; // ERROR! (\*p) 是 const
  + i = 26; //OK
  + p = &j; //OK

8.



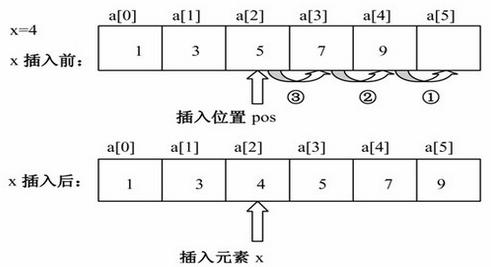
判断哪个被const了的标志是const在\*的前面还是后面

\*前表示 所指的东西不能被修改

\*后表示 指针不能被修改

# 20.在有序数列中插入数据

在升序排序的数组中插入一个元素 用函数编程实现在一个按升序排序的数组中查找x应插入的位置，将x插入数组中，使数组元素仍按升序排列。 提示：插入（Insertion）是数组的基本操作之一。插入法排序算法的关键在于要找到正确的插入位置，然后依次移动插入位置及其后的所有元素，腾出这个位置放入待插入的元素。插入排序的原理如图所示：



程序运行结果示例：

Input array size:

5↙

Input array:

1 3 5 7 9↙

Input x:

4↙

After insert 4:

   1   3   4   5   7   9

**输入格式:**

插入前数组元素个数、数组元素、待插入的元素x的输入格式都是："%d"

**输出格式：**

输入插入前数组元素个数提示信息："Input array size:\n"

输入插入前已按升序排序的数组元素提示信息："Input array:\n"

输入待插入的元素x提示信息："Input x:\n"

输出插入x后的数组元素提示信息："After insert %d:\n"

数组元素输出格式："%4d"

## CSDN解答：

#include<stdio.h>

int main()

{

int n,x,i;

printf("Input array size:\n");

scanf("%d",&n);

int a[n+1];//插入一个数据则把数组设定为n+1大小，我觉得很好~

printf("Input array:\n");

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

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

printf("Input x:\n");

scanf("%d",&x);

int pos;

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

{

if(x<a[i])

{

pos=i;

break; //这个Break相当重要！一找到应该待的位置就不继续往下比较了 否则会插到最后面（倒数第二位置）去！

}

}

for(i=n; i>=pos; i--)

{

a[i]=a[i-1];

}

a[pos]=x;

printf("After insert %d:\n",x);

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

printf("%4d",a[i]);

return 0;

}

# 21. 分组排序

将数据按照奇偶排序

输入10个数，将10个整数按升序排列输出，并且奇数在前，偶数在后。如果输入的10个数是:10 9 8 7 6 5 4 3 2 1 ，则输出：1,3,5,7,9,2,4,6,8,10。（编程提示：可利用2个数组变量，一个用来存放输入的整数，输入后，对这个数组进行排序，然后将数据复制到另一个数组中，先复制奇数再复制偶数）。

程序运行结果如下：

Input 10 numbers:

3 12 7 65 8 4 17 25 36 50↙

Output: 3,7,17,25,65,4,8,12,36,50

输入格式: "%d"

输入数据提示信息："Input 10 numbers:\n"

输出格式：

输出数据提示信息："Output: " （注：冒号“：”的后面有一个空格）

前9个数据的输出格式："%d,"

最后一个数据的输出格式："%d\n"

## 参考答案

#include "stdio.h"

**int** main()

{

**int** arr[10], brr[10];

**int** i, j, tmp;

**printf**("Input 10 numbers:\n");

**for** (i = 0; i < 10; i++)

    {

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

}

**//从小到大排序一次**

**for** (i = 0; i < 9; i++)

    {

**for** (j = i + 1; j < 10; j++)

        {

**if** (arr[j] < arr[i])

            {

                tmp = arr[i];

                arr[i] = arr[j];

                arr[j] = tmp;

            }

        }

    }

**for** (i = 0, j = 0; i < 10; i++)//第一个循环

    {                 //对于数组a[]遍历

**if** (arr[i] % 2 == 1)

        {

            brr[j++] = arr[i];//用新数组来储存奇数列

//brr[j]=arr[i],再让j=j+1.

        }

    }

**for** (i = 0; i < 10; i++)//第二个循环

    {

**if** (arr[i] % 2 == 0)

        {

            brr[j++] = arr[i];

        }

    }

 标答字体 Consolas

### //思维误区！

我刚才在疑惑第二个循环中的j怎么继承第一个循环中的j的值，然而这不是函数。两个循环都在主函数中，所以j总是一个j。i反复使用是因为在循环开始的时候我们总是把i初始化为0了。

**printf**("Output: ");

**for** (i = 0; i < 10; i++)

    {

**if** (i < 9)

        {

**printf**("%d,", brr[i]);

        }

**else**

        {

**printf**("%d\n", brr[i]);

        }

    }

}

# 22.四舍五入

编写一个C函数，实现保留小数点后第N-1位，从第N位四舍五入。用该函数对数组a[5]中的各元素从小数点后第2位开始四舍五入。

a[5]={2.33,2.56,2.65,2.66,2.30}

\*\*输出格式要求："Old array:\n" "a[%d]=%.3f\t" "\nnew array:\n" "a[%d]=%.2f\t"

程序运行示例如下：

Old array:

a[0]=2.330 a[1]=2.560 a[2]=2.650 a[3]=2.660 a[4]=2.300

new array:

a[0]=2.30 a[1]=2.60 a[2]=2.70 a[3]=2.70 a[4]=2.30

## 自解答案：

#include <stdio.h>

#include <stdlib.h>

int main()

{

float a[5];

int i;

float b[5];

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

{

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

b[i]=(int)(a[i]\*10+0.5)/10.0;//用强制转换来完成了四射五入，必须是放大到保留的位数之前，然后加上一个0.5，然后强制转换，然后用浮点除。

}

printf("Old array:\n");

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

{

printf("a[%d]=%.3f\t",i,a[i]);

}

printf("\nnew array:\n");

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

{

printf("b[%d]=%.2f\t",i,b[i]);

}

return 0;

}

# 23.数据的顺时针排序

数字排列。 在下图中的九个点上，空出中间的点，其余的点上任意填入数字1至8；1的位置固定不动，然后移动其余数字，使1到8顺时针从小到大排列。移动的规则是：只能将数字沿线移向空白的点。请编程显示数字移动过程。

○—○—○

︱＼︱／︱

○—◇—○

︱／︱＼︱

○—○—○

\*\*输入格式要求："%d" 提示信息："Please enter original order of digits 1~8:"

\*\*输出格式要求："%2d\n" "The sorting process is as felow:\n" "---%2d---\n"

程序运行示例如下：

Please enter original order of digits 1~8:1 2 3 4 5 6 7 8

The sorting process is as felow:

1 2 3

8 0 4

7 6 5

--- 0---

# 24.常用小代码：找出一组数据中最大的数，并输出其下标

int max=a[0],cou=0;

for (i=1;i<10;i++)

{

if (a[i]>max)

{

max=a[i];

cou=i;

}

}

printf("值最大的元素值为：%d 下标为%d\n",max,cou);

# 25.进阶：同时输出最大和最小值

int main()

{

int a[10],i;

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

{

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

}

int max=a[0],cou1=0;

for (i=1;i<10;i++)

{

if (a[i]>max)

{

max=a[i];

cou1=i;

}

}

int min=a[0],cou2=0;

for (i=1;i<10;i++)

{

if (a[i]<min)

{

min=a[i];

cou2=i;

}

}

printf("max=%d, pos=%d\n",max,cou1);

printf("min=%d, pos=%d\n",min,cou2);

return 0;

}

## 代码优化（一次循环完成）

int min=a[0],max=a[0],cou1=0,cou2=0;

for (i=1; i<10; i++)

{

if (a[i]>max)

{

max=a[i];

cou1=i;

}

if(a[i]<min)

{

min=a[i];

cou2=i;

}

}

printf("max=%d, pos=%d\n",max,cou1);

printf("min=%d, pos=%d\n",min,cou2);

# 23.连续读入数据的小段代码举例

## 一：万能型，‘\n’为停止条件

char a[n];

int i=-1;

printf("please in put the number: ");

do

{

i++;

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

}

while (a[i]!='\n');

## 二：能用特征值作为停止条件

#include <stdio.h>

#include <stdlib.h>

#define n 10

//想要以-1为特征值结束

int main()

{

int a[n]={0};

int i=-1,j;

printf("please in put the number: ");

do

{

i++;

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

}

while (a[i]!=-1);

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

{

printf("%d ",a[j]);

}

return 0;

}

# 24. 用数组完成解密工作

有一电文，已按下面规律译成密码。

A->Z a->z

B->Y b->y

C->X c->x

… …

即第一个字母变成第26个字母，第i个字母变成第（26-i+1）个字母，非字母字符不变。要求编程将密码译回原文，并打印出密码和原文。

程序的运行示例如下：

请输入字符：hello,world!

密码是：hello,world!

原文是：svool,dliow!

## CSDN1：完成hello,world!没毛病

#include <stdio.h>

int main()

{

char a[26], b[26], c[15], m, n;

int i, j;

for (i=1, m='A'; i<=26; i++)

a[i]=m++;

for (i=1, n='a'; i<=26; i++)

b[i]=n++;

printf("请输入字符：");

for (i=0; i<12; i++) //输入密码

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

printf("\n密码是：");

for (i=0; i<12; i++) //输出原密码

printf("%c", c[i]);

for (i=0; i<12; i++){

for (j=1; j<=26; j++){

if (c[i]==a[j]){

c[i]=a[26-i+1];

break;

}

else if (c[i]==b[j]){

c[i]=b[26-j+1];

break;

}

}

}

printf("\n原文是：");

for (i=0; i<12; i++) //输出译出的原文

printf("%c", c[i]);

printf("\n");

return 0;

}

## 参考答案：

#include <stdio.h>

**int** main()

{

**int** i, n;

**char** ch[80], tran[80];

**printf**("请输入字符：");

**gets**(ch);

**printf**("\n密码是：%s", ch);

    i = 0;

**while** (ch[i] != '\0')

    {

**if** ((ch[i] >= 'A') && (ch[i] <= 'Z'))

        {

            tran[i] = 26 + 64 - ch[i] + 1 + 64;

        }

**else** **if** ((ch[i] >= 'a') && (ch[i] <= 'z'))

        {

            tran[i] = 26 + 96 - ch[i] + 1 + 96;

        }

**else**

        {

            tran[i] = ch[i];

        }

        i++;

    }

    n = i;

**printf**("\n原文是：");

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

    {

**putchar**(tran[i]);

    }

**return** 0;

}

错解：

#include <stdio.h>

#include <stdlib.h>

int main()

{

// const int n=12;

char a[12]={0},b[12]={0}; //只输入“hello,world!”，12个正好拿来试试……

int i=-1,j;

printf("please put:");

do

{

i++;

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

}

while (a[i]!='\n');//读入输入的数据

printf("\nthe mi ma is :");

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

printf ("%c",a[j]);

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

{

if (a[j]!=','||a[j]!='!')

b[j]=27-a[j];

else

b[j]=a[j];

}

printf ("\n the yuan is :");

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

printf ("%c",b[j]);

return 0;

}

# 25.判断元音出现的次数

输入一行字符，分别统计其中每个元音字母（a、e、i、o、u不分大小写）的个数。

\*\*输入格式要求：信息提示："Input a line of characters:\n"

\*\*输出格式要求："%4d"

程序的运行示例如下：

Input a line of characters:

How old are you?↙

1 1 0 3 1

## 答案：

#include <stdio.h>

#include <stdlib.h>

#define N 10000

int main()

{

int i=-1, m,count1=0,count2=0,count3=0,count4=0,count5=0;

char str[N];

printf("Input a line of characters:\n");

**do**

**{**

**i++;**

**str[i]=getchar();**

**}while(str[i]!='\n');**

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

{

switch(str[m])

{

case 'a' :

case 'A' :

count1++;

break;

case 'i' :

case 'I' :

count3++;

break;

case 'e' :

case 'E' :

count2++;

break;

case 'o':

case 'O' :

count4++;

break;

case 'u' :

case 'U' :

count5++;

break;

}

}printf("%4d%4d%4d%4d%4d",count1,count2,count3,count4,count5);

}

# C:\Users\75798\Documents\Tencent Files\757986279\Image\C2C\AF8E1231B8DE8548C455F037F759D28E.png26.cy问的一个题

## 自解：

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

double find(double a,double b);

int main()

{

double a,n,sum=0;

printf("input a,n ：");

scanf ("%lf%lf",&a,&n);

double b[100];//数组和常数不能同名

int i;

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

{

b[i]=find(a,i);

}

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

{

sum=sum+b[i];

}

printf("sn=%d",(int)sum);

return 0;

}

double find(double a,double b)//函数想用来计算并返回a[i]

{

double i,sum=0;

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

{

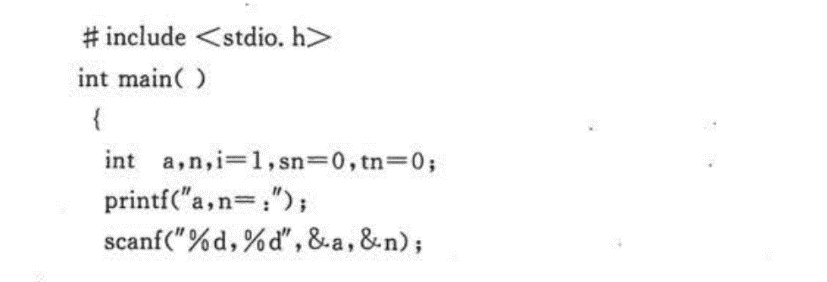
sum=sum+a\*pow(10,i);

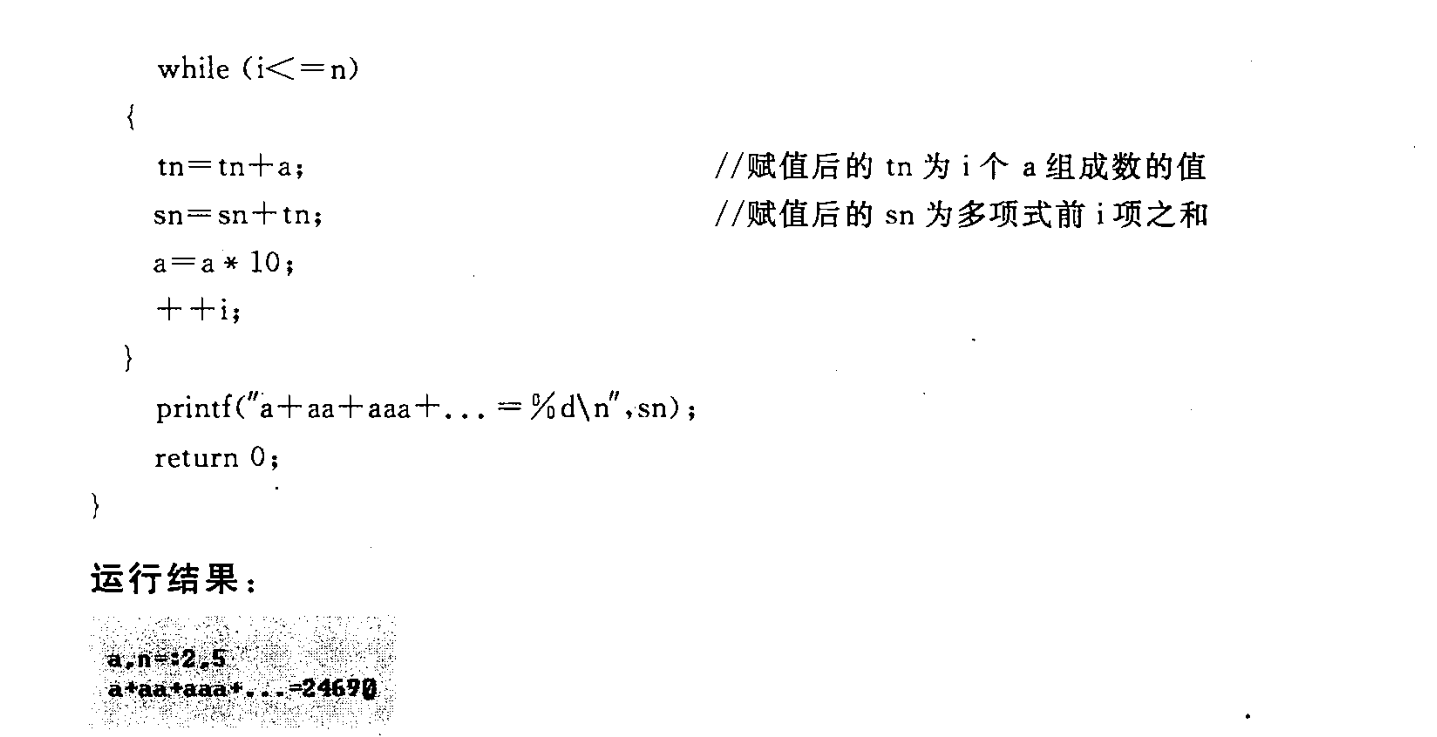
}

return sum;

}

## 非常精简的标答：





# 27. 检查数中重复出现的数字

程序：检查数中重复出现的数字。

用户输入数后，程序显示信息Repeated digit或No Repeated digit：

Enter a number: 28212

Repeated digit

数28212有一个重复的数字（2），而数9357则没有。

\*\*输入格式要求："%ld" 提示信息："Enter a number :"

\*\*输出格式要求："Repeated digit\n\n" "No Repeated digit\n\n"

## 自解：

#include <stdio.h>

#include <stdlib.h>

int main()

{

long int a[100]= {0},num;

int i=0,m,n;

// int t;

printf("Enter a number :");

scanf("%ld",&num);

while (num!=0)

{

a[i]=num%10;

i++;

num=num/10;

}

for (m=0; m<i-1; m++)

{

for (n=m+1; n<i; n++)

{

if (a[m]-a[n]==0)

{

printf("Repeated digit\n\n");

// t=-1;

goto loop;

}

}

}

printf("No Repeated digit\n\n");

loop:

return 0;

}

### 体悟

跳出多重循环用goto方便的多；然后if 后面千万别傻不拉几的加;……

# 28. 求并输出3~6位自方幂数

一个n位正整数如果等于它的n个数字的n次方和，该数称为n位自方幂数。设计求3~6位自方幂数。

\*\*输出格式要求："%d位自幂数有:" "%ld\t" "\n" (每位完后换行)

程序运行示例如下：

3位自幂数有:153 370 371 407

4位自幂数有:1634 8208 9474

5位自幂数有:54748 92727 93084

6位自幂数有:548834

## 参考答案

#include<stdio.h>

#include<math.h>

**int** main()

{

**int** a1, a2, a3, a4, a5, a6, i, j;

**long** m3, m4, m5, m6;

**long** n3, n4, n5, n6;

**static** **int** t[9];

**static** **long** s[9][10];

**for** (a1 = 1; a1 <= 9; a1++)

**for** (a2 = 0; a2 <= 9; a2++)

**for** (a3 = 0; a3 <= 9; a3++)

            {

                m3 = a1 \* 100 + a2 \* 10 + a3;

                n3 = (**long**)(**pow**(a1, 3) + **pow**(a2, 3) + **pow**(a3, 3));

**if** (m3 == n3)

                {

                    s[3][++t[3]] = m3;          //三位水仙花数

                }

**for** (a4 = 0; a4 <= 9; a4++)

                {

                    m4 = m3 \* 10 + a4;

                    n4 = (**long**)(**pow**(a1, 4) + **pow**(a2, 4) + **pow**(a3, 4) + **pow**(a4, 4));

**if** (m4 == n4)

                    {

                        s[4][++t[4]] = m4;    //四位玫瑰花数

                    }

**for** (a5 = 0; a5 <= 9; a5++)

                    {

                        m5 = m4 \* 10 + a5;

                        n5 = (**long**)(**pow**(a1, 5) + **pow**(a2, 5) + **pow**(a3, 5) + **pow**(a4, 5) + **pow**(a5, 5));

**if** (m5 == n5)

                        {

                            s[5][++t[5]] = m5; //五位五角星数

                        }

**for** (a6 = 0; a6 <= 9; a6++)

                        {

                            m6 = m5 \* 10 + a6;

                            n6 = (**long**)(**pow**(a1, 6) + **pow**(a2, 6) + **pow**(a3, 6) + **pow**(a4, 6) + **pow**(a5, 6) + **pow**(a6, 6));

**if** (m6 == n6)

                            {

                                s[6][++t[6]] = m6; //六位六合数

                            }

                        }

                    }

                }

            }

**for** (i = 3; i <= 6; i++)

    {

**printf**("%d位自幂数有:", i);

**for** (j = 1; j <= t[i]; j++)

        {

**printf**("%ld\t", s[i][j]);

        }

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

    }

**return** 0;

}

# 29. 二维数组达成闰年问题

从键盘输入某年某月（包括闰年），编程输出该年的该月拥有的天数。

\*\*输入格式要求："%d,%d" 提示信息："Input year,month:" "The number of days is %d\n"

\*\*输出格式要求："%d"

程序运行示例1如下：

Input year,month:1984,2

The number of days is 29

程序运行示例2如下：

Input year,month:2000,2

The number of days is 29

程序运行示例3如下：

Input year,month:1985,2

The number of days is 28

程序运行示例4如下：

Input year,month:1983,13

Input year,month:1983,-1

Input year,month:1983,1

The number of days is 31

## 答案

#include <stdio.h>

#include <stdlib.h>

int main()

{

int a[2][12]= {31,28,31,30,31,30,31,31,30,31,30,31,31,29,31,30,31,30,31,31,30,31,30,31};

int year,mon;

begin:

printf("Input year,month:");

scanf("%d,%d",&year,&mon);

int y=0;

if (((year%4==0)&&(year%100!=0))||year%400==0)

y=1;

else

y=0;

if (mon<0||mon>12)

goto begin;

printf("The number of days is %d\n",a[y][mon-1]);

return 0;

}

# 30.思考：出现了意料之外的数字大概就是超出数组范围了

定义1个3 \* 2的二维数组a，数组元素的值由下式给出，按矩阵的形式输出a。

a[i][j] = i + j (0<=i<=2, 0<=j<=1)

\*\*输出格式要求："%4d"

## 解：

#include <stdio.h>

#include <stdlib.h>

//a[i][j] = i + j (0<=i<=2, 0<=j<=1)

int main()

{

int a[3][2];

int i,j;

for (i=0; i<=2; i++)

{

for (j=0; j<=1; j++)

{

a[i][j] = i + j ;

printf("%4d",a[i][j]);

}

printf("\n");

}

return 0;

}

# 31.指针入门

给定如下的数组：

float litres[] = { 11.5, 11.21, 12.7, 12.6, 12.4 } ;

float miles[] = { 471.5, 358.72, 495.3, 453.6, 421.6 } ;

int mpl[5] ; /\* Miles per litre. \*/

写一个程序计算并显示mpl中每个元素的值。使用指针而不是下标访问数组元素。

\*\*输出格式要求："%d\t"

## 参考答案

|  |
| --- |
| #include <stdio.h>    **int** main()  {  **float** litres[] = {11.5, 11.21, 12.7, 12.6, 12.4};  **float** miles[] = {471.5, 358.72, 495.3, 453.6, 421.6};  **int** mpl[5];  // Miles per litre.  **int** \*p=NULL;  **float** \*pl=NULL, \*pm=NULL;        pl = litres;**//指针指向首地址**      pm = miles; **//指针指向首地址**  **for** (p = mpl; p < mpl + (**sizeof**(mpl) / **sizeof**(**int**)); p++)  //用P对上mpl的首地址，然后计算它的尾地址，完成限制；  //这个循环限制条件局限性很大，不推荐!!!!!      {          \*p = (**int**)(\*pm / \*pl);  **printf**("%d\t", \*p);          pm++;**//指针地址顺移来完成遍历**          pl++;**//指针地址顺移来完成遍历**      }    **return** 0;  } |

# 32.指针法倒序输出输入的字符串

写一个程序逆反输入的字符串。

\*\*提示信息："请输入一个字符串："

\*\*输出格式要求："该字符串逆反后为%s！\n"

## C语言论坛答案：

#include <stdio.h>

#include <string.h>

int main()

{

int len;

char s[256],\*p;

printf("请输入一个字符串：");

gets(s);

len = strlen(s);

p = &s[len-1];

printf("该字符串逆反后为");

while(p!=s)//p的地址没到s的首地址之前都继续执行循环。

{

printf("%c", \*p--);

}

printf("%c！\n", \*p);

return 0;

}

# 33. 指针的循环控制条件

写一个程序读入你的姓名，然后每个字母间加一个空格后输出。例如，姓名John显示为J o h n。

\*\*输入格式要求："%s" 提示信息："请输入你的姓名："

## 法一：

#include <stdio.h>

#include <stdlib.h>

int main()

{

char name[100];

printf("请输入你的姓名：");

gets(name);

char \*p=NULL;

for (p=name;\*p!='\0';p++)

{printf("%c ",\*p);

}

return 0;

}

## 法二：

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

int main()

{

char name[100];

printf("请输入你的姓名：");

gets(name);

char \*p=NULL;

for (p=name;p<(name+strlen(name)+1);p++)

{printf("%c ",\*p);

}

return 0;

}

## 解析

！=’/0’才是最好的限制条件，另一个有点危险，似乎会丢分……

# 34. 字符串：数组名相当于个const指针

通过键盘输入10个单词，将10个单词排序后重新打印输出（单词最大长度20，且可以包含有空格）。下面程序中存在比较隐蔽的错误，请通过分析和调试程序，发现并改正程序中的错误。注意：

1、请将修改正确后的完整源程序拷贝粘贴到答题区内；

2、改错时不能改变程序原有的意图，不能删除原有的语句但可修改原有的语句；

3、对于没有错误的语句，请不要修改，修改原本正确的语句也要扣分；

4、当且仅当错误全部改正，且程序运行结果调试正确，才给加5分。

|  |
| --- |
| #include <stdio.h>  **int** main()  {  **char** \*t, \*p[10];  **int** i, j;    **for** (i = 0; i < 10; i++)  **gets**(p[i]);    **for** (i = 0; i < 10; i++)      {  **for** (j = i ; j < 10; j++)          {  **if** ((p[j] < p[i]))              {  **strcpy**(t, p[i]);  **strcpy**(p[i], p[j]);  **strcpy**(p[j], t);              }          }      }  **for** (i = 0; i < 10; i++)      {  **puts**(p[i]);      }    **return** 0;  } |

## 错解：

#include <stdio.h>

#include <string.h>

int main()

{

char t[10][20], p[10][10];

int i, j;

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

gets(p[i]);//输入10个字符串

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

{

for (j = i +1; j < 10; j++)

{

if (strcmp(p[j] ,p[i])<0)

{

strcpy(t, p[i]);

strcpy(p[i], p[j]);

strcpy(p[j], t);

}

}

}

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

{

puts(p[i]);

}

return 0;

}

## 正解：

#include <stdio.h>

#include <string.h>

int main()

{

char t[20], p[10][20];

int i, j;

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

gets(p[i]);//输入10个字符串

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

{

for (j = i +1; j < 10; j++)

{

if (strcmp(p[j] ,p[i])<0)

{

strcpy(t, p[i]);

strcpy(p[i], p[j]);

strcpy(p[j], t);

}

}

}

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

{

puts(p[i]);

}

return 0;

}

# 35. 编程判断输入的一个字符串是否是“回文”

## 代码寄存：

#include <stdio.h>

#include <string.h>

int fun(char a[]);

int main()

{

char a[80];

printf("Input a string:\n");

gets(a);

if (fun(a))

{

printf("This string is a plalindrome.\n");

}

else

{

printf("This string is not a plalindrome.\n");

}

}

int fun(char a[])

{

int i, j=strlen(a)-1;//-1非常重要！否则尾数是’\0’，这样永不回文……

for (i = 0; i < strlen(a)-1; i++,j--)

{

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

{

return 0;

}

}

}

# 36.自制的字符连接函数MyStrCat

## 代码寄存：

#include <stdio.h>

#include <string.h>

void MyStrCat(char \*dest, char \*source);

int main()

{

char first[80];

char second[80];

printf("Input the first string:");

gets(first);

printf("Input the second string:");

gets(second);

MyStrCat(first, second);

printf("The result is : %s\n", first);

}

void MyStrCat(char \*dest, char \*source)

{

int i = 0, j = 0;

while (\*(dest+i)!='\0')

{

i++;

}

j=i;//注意，这个位置的i是循环结束的i的大小，所以后面需要再初始化一次。

for (i=0; \*(source+i)!='\0'; i++,j++)

{

\*(dest+j) = \*(source+i);

}

\*(dest+j) = '\0';

}

# 37. 调用GetInput()函数从键盘任意输入一个字符串，然后分别在GetInput()和main()中输出该字符串。

调用GetInput()函数从键盘任意输入一个字符串，然后分别在GetInput()和main()中输出该字符串。

注意：

（1）请将修改正确后的完整源程序拷贝粘贴到答题区内。

（2）对于没有错误的语句，请不要修改，

修改原本正确的语句也要扣分。

（3）当且仅当错误全部改正，且程序运行结果调试正确，才给加5分。

|  |
| --- |
| #include  <stdio.h>  **void** GetInput(**char**\* s);    **int** main(**void**)  {  **char** \*ptr = NULL;     GetInput(ptr);  **puts**(ptr);  **return** 0;  }    **void** GetInput(**char**\* s)  {  **scanf**("%s", s);  **puts**(s);  } |

## 参考答案：

#include <stdio.h>

void GetInput(char\* s);

int main(void)

{

char str[80],\*ptr = str;**//定义指针字符串时，必须先定义字符数组再指过去，否则报错！**

GetInput(ptr);

puts(ptr);

return 0;

}

void GetInput(char\* s)

{

scanf("%s", s);

puts(s);

}