getchar() 输入字符函数

putchar() 输出字符函数

#include <stdio.h>

int main()

{

    int ch = getchar();

    putchar(ch);

    printf("%c\n",ch);

    return 0;

}

//EOF - end of file 文件结束标志

缓冲区问题

#include <stdio.h>

int main()

{

    int ret = 0;

    char password[20] = {0};

    int ch = 0;

    printf("请输入密码:\n");

    scanf("%s",password);       //输入密码123456 后enter 缓冲区存在123456\n

    printf("请确认(Y/N)\n");   //123456被password带走 缓冲区剩余 \n

    //getchar();                  //让getchar() 将缓冲区的 \n 带走，但只能带走一个

    while((ch=getchar()) != '\n')

    {

        ;

    }

    ret = getchar();            //此时第一个getchar() 自动拾取 \n  缓冲区清空

    if (ret == 'Y')

    {

        printf("确认密码\n");

    }

    else

    {

        printf("取消输入\n");

    }

    return 0;

}

#include <stdio.h>

int main()

{

    int ch = 0;

    //EOF - end of file 文件结束标志

    while ((ch = getchar()) != EOF) //EOF 常量=-1

    {

        if(ch<'0' || ch>'9')

            continue;

        putchar(ch);

    }

    return 0;

}

for 循环

for(表达式1;表达式2;表达式3)

循环语句；

表达式1为初始化内容，用于初始化循环变量。表达式2为条件判断部分，用于判断循环啥时候终止。表达式3为调整部分，用于循环条件的调整

#include <stdio.h>

int main()

{

    int i = 0;

    for(i=1;i<=10;i++)

    {

        printf("%d\n",i);

    }

    return 0;

}

for循环的注意事项

1、不可在for循环体内修改循环变量，防止for循环失去控制

2、建议for语句的循环控制变量的取值采用“前闭后开区间”写法

#include <stdio.h>

int main()

{

    int i = 0;

    //for(i=1;i<=10;i++)

for(i=1;i<=11;i++) //前闭后开区间

    {

        if (i = 5) //此时对变量i进行了修改，导致死循环

continue;

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

    }

    return 0;

}

for循环的变种

for循环的初始化、调整、判断都可以省略

判断部分省略时，判断条件恒为真

#include <stdio.h>

int main()

{

    //for循环的初始化、调整、判断都可以省略

    //for(;;)

    //    printf("hehe\n");

    int i = 0;

    int j = 0;

    for (;i<10;i++)

    {

        for(;j<10;j++)

        {

            printf("hehe\n");

        }

        //此时只会打印10个hehe，因为省略了j的初始化变量，j变为了全局变量

    }

    return 0;

}

do ... while() 循环

语法：

do

循环语句;

while(表达式);

类似倒装句

#include <stdio.h>

int main()

{

    //1-10

    int i = 1;

    do

    {

        printf("%d ", i);

        i++;

    }

    while (i<=10);

    return 0;

}

continue 在do...while中的使用

#include <stdio.h>

int main()

{

    //1-10

    int i = 1;

    do

    {

        if (i == 5)

            continue;

        printf("%d ", i);

        i++;

    }

    while (i<=10);

    return 0;

}

**练习**

阶乘的运算

//for

#include <stdio.h>

int main()

{

    int m = 0;

    int n = 0;

    int k = 1;

    printf("请输入要求的阶乘数：\n");

    scanf("%d",&n);

    for (m=1;m<=n;m++)

    {

        k = k\*m;

    }

    printf("%d\n",k);

    return 0;

}

//while

#include <stdio.h>

int main()

{

    int m = 1;

    int n = 0;

    int k = 1;

    printf("请输入要求的阶乘数：\n");

    scanf("%d",&n);

    while (m<=n)

    {

        k = k \* m;

        m++;

    }

    printf("%d\n",k);

    return 0;

}

//do ... while

#include <stdio.h>

int main()

{

    int m = 1;

    int n = 0;

    int k = 1;

    printf("请输入要求的阶乘数：\n");

    scanf("%d",&n);

    do

    {

        if(n == 0)

        {

            printf("0");

            break;

        }

        k = k \* m;

        m++;

    } while (m<=n);

    printf("%d\n",k);

    return 0;

}

求1！+2！+3！

#include <stdio.h>

int main()

{

    int m = 0;

    int n = 0;

    int k = 1;

    int sum = 0;

    for (m=1;m<=3;m++)

    {

        k = 1;

        for(n=1;n<=m;n++)

        {

            k = k \* n;

        }

        sum = sum + k;

    }

    printf("%d\n",sum);

    return 0;

}

遍历数组找到该数对应的下标

#include <stdio.h>

int main()

{

    int arr[] = {1,2,3,4,5,6,7,8,9};

    int k = 7;

    int i = 0;

    //计算数组的长度

    int sz = sizeof(arr)/sizeof(arr[0]);

    for (i=0;i<sz;i++)

    {

        if (k == arr[i])

        {

            printf("找到了，下标是：%d\n",i);

            break;

        }

    }

    if (i == sz)

    {

        printf("找不到\n");

    }

    return 0;

}

二分法找到数组中要找数据的下标

#include <stdio.h>

int main()

{

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

    int sz = sizeof(arr)/sizeof(arr[0]);

    int k = 12;

    int left = 0;   //左下标

    int right = sz - 1; //右下标

    while (left <= right)

    {

        int mid = (left+right)/2;

        if (arr[mid] > k)

        {

            right = mid-1;

        }

        else if(arr[mid] < k)

        {

            left = mid+1;

        }

        else

        {

            printf("找到了，下标是：%d\n",mid);

            break;

        }

    }

    if (left>right)

    {

        printf("找不到\n");

    }

    return 0;

}

多个字符从两端向中间汇集

#include <stdio.h>

#include <string.h>

#include <windows.h>

int main()

{

    char arr1[] = "Hello World!";

    char arr2[] = "############";

    int left =0;

    //int right = sizeof(arr1)/sizeof(arr1[0])-2;

    int right = strlen(arr1)-1;

    while (left<=right)

    {

        arr2[left] = arr1[left];

        arr2[right] = arr1[right];

        printf("%s\n",arr2);

        Sleep(1000);    //休息一秒

        left++;

        right--;

    }

    return 0;

}

//登录系统

“==”不能用来比较两个字符串是否相等，应该使用一个库函数 - strcmp

语法： strcmp(password,'123456') == 0//等于0即为相等

#include <stdio.h>

#include <string.h>

int main()

{

    int i = 1;

    char password[20] = {0};

    for (i=0;i<3;i++)

    {

        printf("请输入密码:>");

        scanf("%s",password);

        if (strcmp(password,"123456") == 0)

        {

            printf("登录成功！\n");

            break;

        }

        printf("密码错误，请重试！\n");

    }

    if(i == 3)

    {

        printf("登录失败！\n");

    }

    return 0;

}

将三个数按照从大到小的顺序排列

#include <stdio.h>

int main()

{

    int a = 0;

    int b = 0;

    int c = 0;

    scanf("%d%d%d", &a, &b, &c);

    if(a<b)

    {

        int tmp = a;

        a = b;

        b = tmp;

    }

    if (a<c)

    {

        int tmp = a;

        a = c;

        c = tmp;

    }

    if (b<c)

    {

        int tmp = b;

        b = c;

        c = tmp;

    }

    printf("%d %d %d", a, b, c);

    return 0;

}

判断素数

1.试除法

#include <stdio.h>

#include <math.h>

int main()

{

    int i = 0;

    int count = 0;

    for(i=100;i<200;i++)

    {

        //判断是否为素数

        //素数判断规则（素数是只能被1和他本身整数的数）

        //1.试除法

        int j = 0;

        //for (j=2;j<i;j++)

        //sqrt - 数学库函数中的开平方

        for (j=2;j<=sqrt(i);i++)

        {

            if (i%j == 0)

            {

                break;

            }

        }

        //if (j == i)

        if(j>sqrt(i))

        {

            count++;

            printf("%d ", i);

        }

    }

    printf("\ncount=%d\n",count);

    return 0;

}

//当涉及到小数、分数时，定义的浮点数需要带小数点，计算时也应该带小数点

#include <stdio.h>

int main()

{

    //当涉及到小数、分数时，定义的浮点数需要带小数点，计算时也应该带小数点

    int i = 0;

    float sum = 0.0;

    for (i=1;i<100;i++)

    {

        sum += 1.0/i;

    }

    printf("%f\n",sum);

    return 0;

}

//输出数组中最大的数

//输出数组中最大的数

#include <stdio.h>

int main()

{

    int arr[] = {-1,-2,-3,-4,-5,-6,-7,-8,-9};

    int max = arr[0];

    int i = 0;

    int sz = sizeof(arr)/sizeof(arr[0]);

    for (i=1;i<sz;i++)

    {

        if (arr[i]>max)

        {

            max = arr[i];

        }

    }

    printf("%d\n",max);

    return 0;

}

九九乘法表

#include <stdio.h>

int main()

{

    int i = 0;

    for (i=1;i<=9;i++)

    {

        int j = 1;

        for (j=1;j<=i;j++)

        {

            printf("%d\*%d=%2d ", i, j, i\*j);

            //%2d - 打印的结果存在两位，当只有一位数是，自动补0

        }

        printf("\n");

    }

    return 0;

}

猜数字游戏 - 随机数

//rand() 生成随机数，使用前需要先使用srand()进行设置，srand() - 括号中需要一个随时变换的数才能进行随机
数的生成，此时变引用时间戳的概念

时间戳 - 当前计算机的时间 减去 计算机的起始时间（1970.1.1.0：0：0）-转换成的数字

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

void menu()

{

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

    printf("\*\*\*  1.Play  0.Exit  \*\*\*\n");

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

}

void game()

{

    //1.生成随机数

    //rand()  生成随机数，使用前需要先使用srand()进行设置，srand() - 括号中需要一个随时变换的数才能进行随机

    //数的生成，此时变引用时间戳的概念

    //时间戳 - 当前计算机的时间 减去 计算机的起始时间（1970.1.1 0：0：0）-转换成的数字

    int ret = 0;

    int guess = 0;

    //拿时间戳来设置随机数的生成起始点

    //time\_t time(time\_t\*timer)

    //time\_t

    //srand((unsigned int)time(NULL)); - 此处是为了给随机数设置起始点，起始点的设置只要在主函数中设置一次，

    //使用rand()进行调用就可以

    //RAND\_MAX - 32767

    ret = rand()%100+1;   //生成1-100之间随机数，可以使用取模的方法！

    //printf("猜数字:%d\n",ret);

    //2.猜数字

    while (1)

    {

        printf("请猜数字:");

        scanf("%d",&guess);

        if (guess>ret)

        {

            printf("猜大了\n");

        }

        else if (guess<ret)

        {

            printf("猜小了\n");

        }

        else

        {

            printf("猜对了\n");

            break;

        }

    }

}

int main()

{

    srand((unsigned int)time(NULL));

    int input = 0;

    do

    {

        menu();

        printf("请选择:");

        scanf("%d",&input);

        switch (input)

        {

        case 1:

            game();

            break;

        case 0:

            printf("游戏结束\n");

            break;

        default:

            printf("选择错误\n");

            break;

        }

    } while (input);

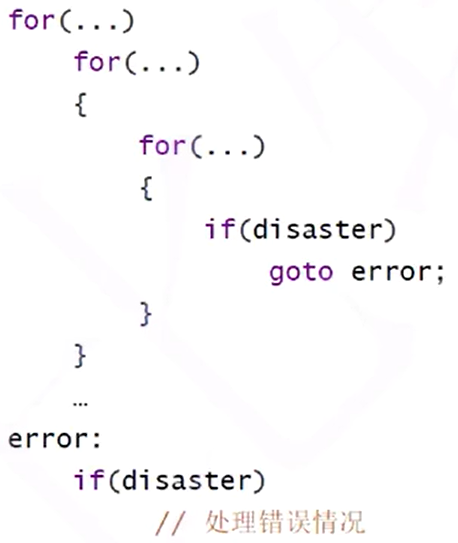
    return 0;

}

goto 语句

goto语句可以在代码中跳来跳去，一般代码不使用goto

使用场景为：为了一次性跳出多个嵌套循环，例如：



#include <stdio.h>

#include <string.h>

#include <stdlib.h>

int main()

{

    //1.

// again:

//     printf("Hell0 World\n");

//     goto again;

    //2.

//     printf("Hello World\n");

//     goto again;

//     printf("Python\n");

// again:

//     printf("HeHe\n");

//     //此时goto语句将python跳过了，直接执行HeHe

    //3.关机小程序

    //shutdown -s -t 60    计划60秒后关机

    //shutdowm -a          取消计划关机

    //system               执行系统命令     头文件：<stdlib.h>

    char input[20] = {0};

    system("shutdown -s -t 600");

again:

    printf("电脑将在60秒后关机，输入“取消”进行取消:\n");

    scanf("%s", input);

    if (strcmp(input, "qx") == 0)

    {

        system("shutdown -a");

    }

    else

    {

        goto again;

    }

    return 0;

    //不使用goto

    // while (1)

    // {

    //     printf("电脑将在60秒后关机，输入“取消”进行取消:\n");

    //     scanf("%s", input);

    //     if (strcmp(input, "qx") == 0)

    //     {

    //         system("shutdown -a");

    //         break;

    //     }

    // }

    // return 0;

}