1. 一语句实现x是否为2的若干次幂的判断。

Printf(“%c”, (a)&(a-1)?’n’:’y’) //如果是打印y，不是打印n

x如果是2的若干次幂，那么它的二进制就只有一个1

1. 中断是嵌入式系统中重要的组成部分，这导致了很多编译开发商提供一种扩展—让标准C支持中断。具代表事实是，产生了一个新的关键字 \_\_interrupt。下面的代码就使用了\_\_interrupt关键字去定义了一个中断服务子程序(ISR)，请评论一下这段代码的。  
   \_\_interrupt double compute\_area (double radius)   
   {   
        double area = PI \* radius \* radius;   
        printf(" Area = %f", area);   
        return area;   
   }

ISR不能返回一个值

ISR不能传递参数

在许多的处理器/编译器中，浮点一般是不可重入的。有些处理器/编译器就是不允许在ISR中做浮点运算

ISR应该是短而有效率的，在ISR中做浮点运算是不明智的

Printf经常有重入和性能上的问题。

解决：

不能在中断服务程序中执行类似操作的原因是运算时间太长,不能在中段中作复杂的工作,你可以在中断服务程序中置一个标志位.然后在主程序中查询此位,判断是否执行计算子程序.当然这样做会在中断产生与实际的中断响应之间产生一定的延迟,如果你的系统对时间不太敏感还好,如果非常敏感可以考虑采用rtos。否则。。。多加几句对中断标志位的判断语句。。。

1. 下面的代码输出是什么，为什么？  
   void foo(void)   
   {   
        unsigned int a = 6;   
        int b = -20;   
        (a+b > 6)? puts("> 6") : puts("<= 6");   
   }

这个问题测试你是否懂得C语言中的整数自动转换原则，我发现有些开发者对这些东西懂得极少，不管如何，这无符号整型问题的答案是输出是 ">6"。原因是当表达式中存在有符号类型和无符号类型时所有的操作数都自动转换为无符号类型。

因此-20变成了一个非常大的正整数，所以该表达式计算出的结果大于6。这一点对于应当频繁用到无符号数据类型的嵌入式系统来说是丰常重要的。

1. 评价下面的代码片断：  
   unsigned int zero = 0;   
   unsigned int compzero = 0xFFFF;   
   /\*1‘s complement of zero \*/

对于一个int型不是16位的处理器为说，上面的代码是不正确的。应编写如下：

unsigned int compzero= ~0;

1. 下面的代码片段的输出是什么，为什么？  
   char \*ptr;   
   if ((ptr = (char \*)malloc(0)) == NULL)   
    puts("Got a null pointer");   
   else   
    puts("Got a valid pointer");

Got a valid pointer

1. 编写 strcpy 函数

已知 strcpy 函数的原型是 char \*strcpy(char \*strDest, const char \*strSrc);其中 strDest 是目的字符串，strSrc 是源字符串。

（1）不调用 C++/C 的字符串库函数，请编写函数 strcpy 。

（2）strcpy 能把 strSrc 的内容复制到 strDest，为什么还要 char \* 类型的返回值？

1. 写出二分查找的代码 。
2. 请编写一个 C 函数，该函数给出一个字节中被置 1 的位的个数。
3. 请编写一个 C 函数，该函数将给定的一个字符串转换成整数。
4. 请编写一个 C 函数，该函数将给定的一个整数转换成字符串。