注意：本次学习需配合大量实践

#define \_CRT\_SECURE\_NO\_WARNINGS 1

//调试技巧

/\*

1.什么是bug？

2.调试是什么？

3.debug和release的介绍。

(1).debug通常称为调试版本，它包含调试信息，

并且不做任何优化，便于程序员调试程序。

(2).release称为发布版本，它往往是进行了各种优化，

使得程序在大妈大小和运行速度上都是最优的，以

便用户很好地使用。

4.windows环境调试介绍。

（1）.deubg

注：linux开发环境的调试工具gdb。

5.一些调试的实例。

6.如何写出易于调试的代码。

7.编程常见的错误。

\*/

/\*调试的基本步骤：

（1）.发现程序错误的存在

（2）.以隔离、消除等方式对错误进行定位

（3）.确定错误产生的原因

（4）.提出纠正错误的解决办法

（5）.对程序错误予以改正,重新测试

\*/

//#include <stdio.h>

//void test()

//{

// int a = 10;

// int b = 20;

// int c = a + b;

//}

//int main()

//{

// int arr[10] = { 0 };

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

// int i = 0;

// for (i = 0; i < sz; i++)

// {

// arr[i] = i + 1;

// }

// for (i = 0; i < sz; i++)

// {

// test( );

// printf("%d\n", arr[i]);

// }

// return 0;

//}

//学会快捷键

/\*

最常用的几个快捷键:

F5：

启动调试，经常用来直接调到下一个断点处

F9：

创建断点和取消断点

F10：

逐过程，通常用来处理一个过程，一个过程可以是一次函数调用，

也可以是一条语句。

F11：

逐语句，就是每次都执行一条语句，但是这个快捷键可以使我们的执行逻辑

进入函数的内部（这是最常用的）。

CTRL+F5

开始执行不调试，如果你想让程序直接运行起来而不调试就可以直接使用

\*/

/\*

3.调试程序的时候如何查看程序当前的信息

（1）.查看临时变量的值

\*/

//调用堆栈:

//（1）.返回的是函数的调用逻辑

//（2）.堆栈=栈

//void test2()

//{

//

//}

//void test1( )

//{

// test2( );

//}

//void test()

//{

// test1( );

//}

//int main( )

//{

// test( );

// return 0;

//}