VS2017 调试工具的使用

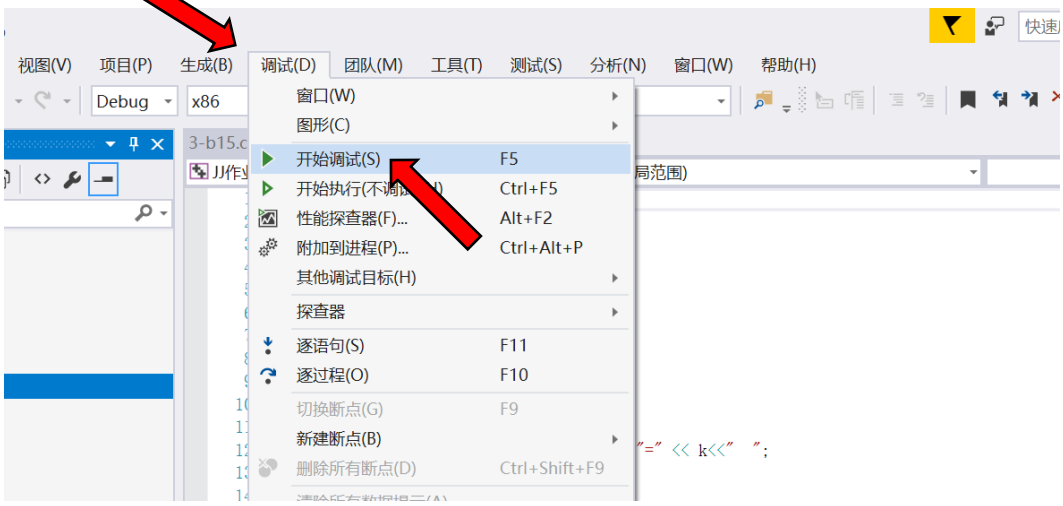
计 1 郭思远 1751151

完成日期：12 月 28 日

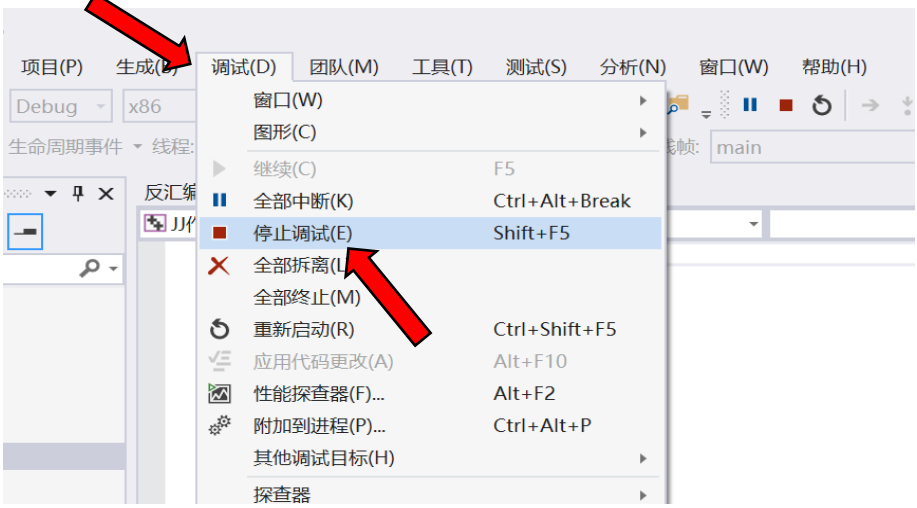
1. VS2017 下调试工具的基本使用方法

1.1 如何开始调试？如何结束调试？

选择菜单，选择“调试”中的“开始调试”，或按 F5。

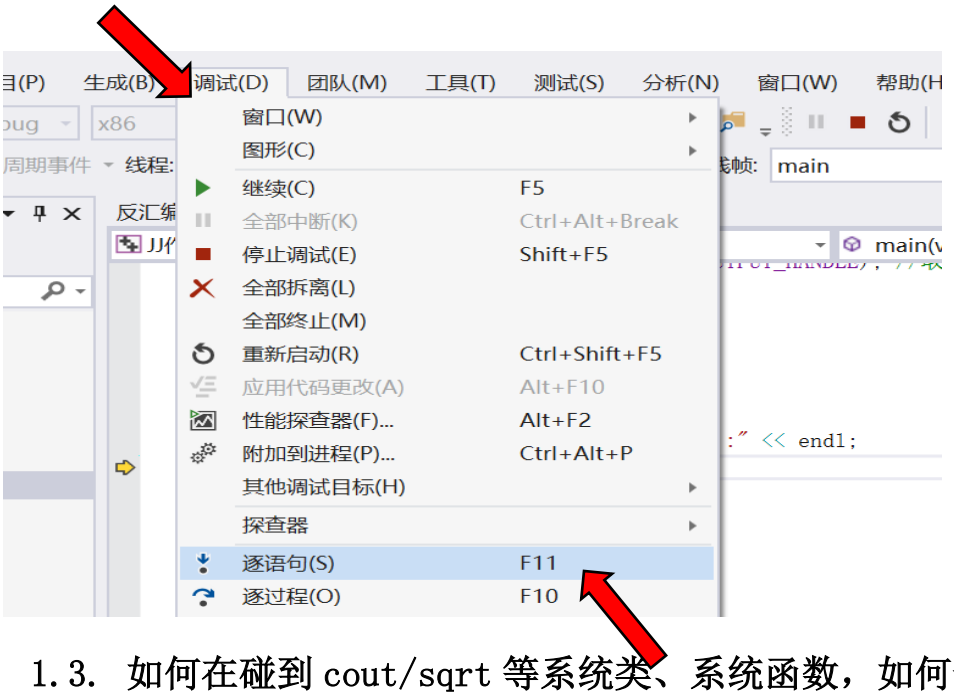


开始调试后，选择菜单，选择“调试”中的“停止调试”，或按 Shift+F5。



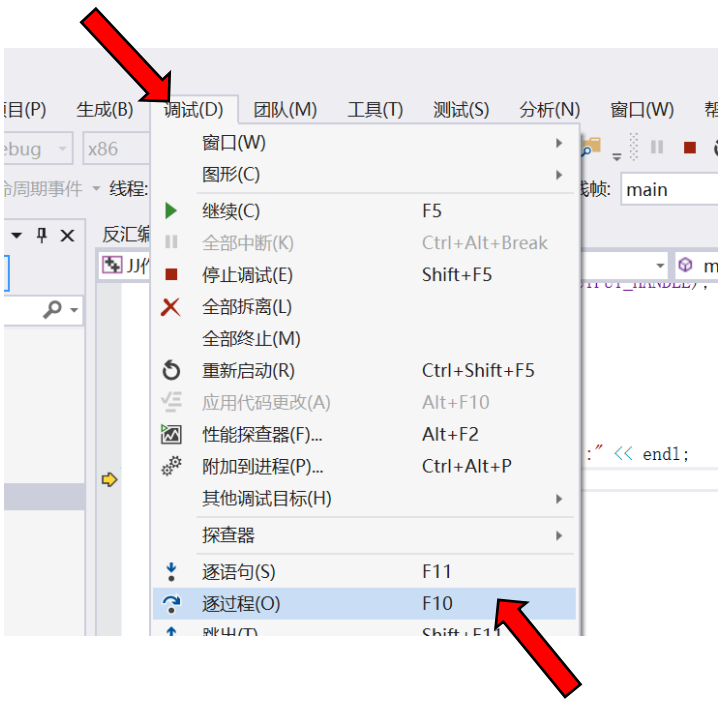
1.2. 如何在一个函数中每个语句单步执行

选择菜单，选择“调试”中的逐语句，或按 F11.



1.3. 如何在碰到 cout/sqrt 等系统类、系统函数，如何一步完成？

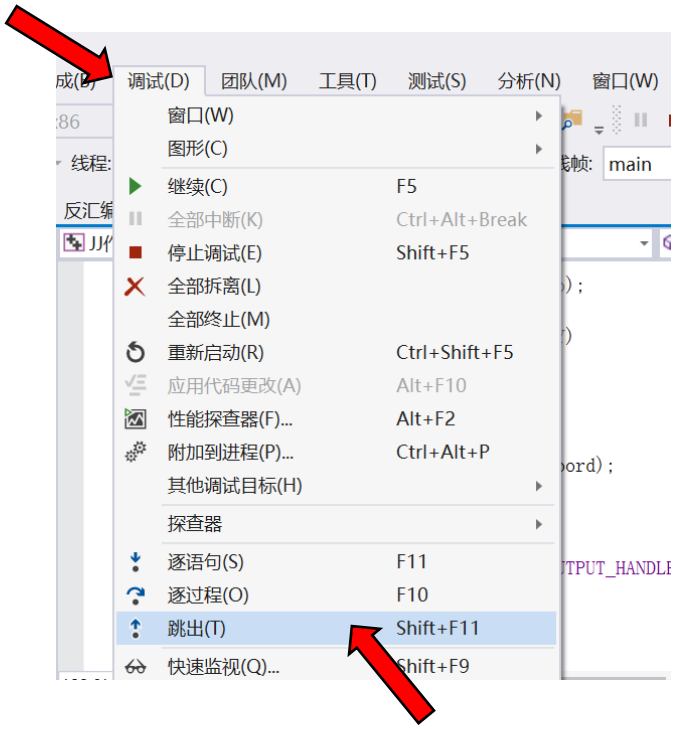
选择菜单，选择“调试”中的逐过程，或按 F10



1.4. 如果已经进入到 cout/sqrt 等系统类、系统函数的内部，

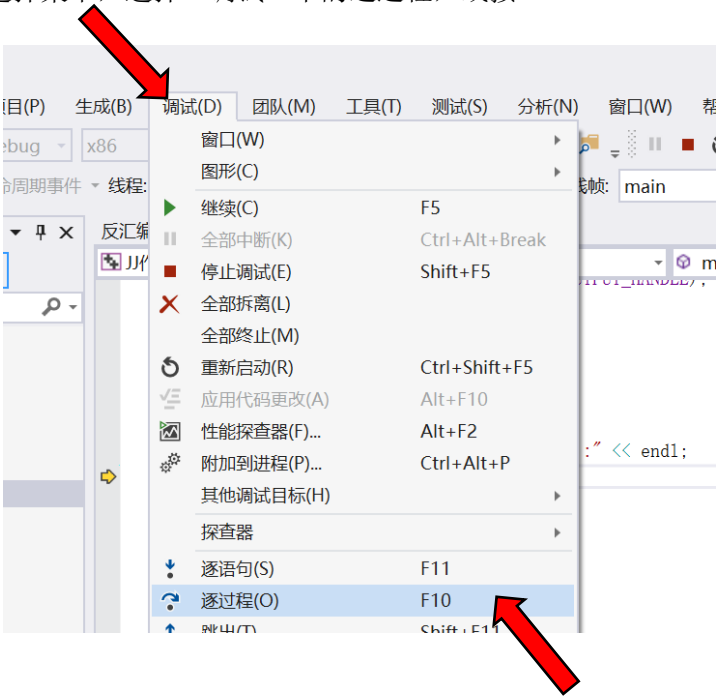
如何跳出并返回自己的函数？

选择菜单，选择“调试”中的“跳出”，或按 Shift+F11。



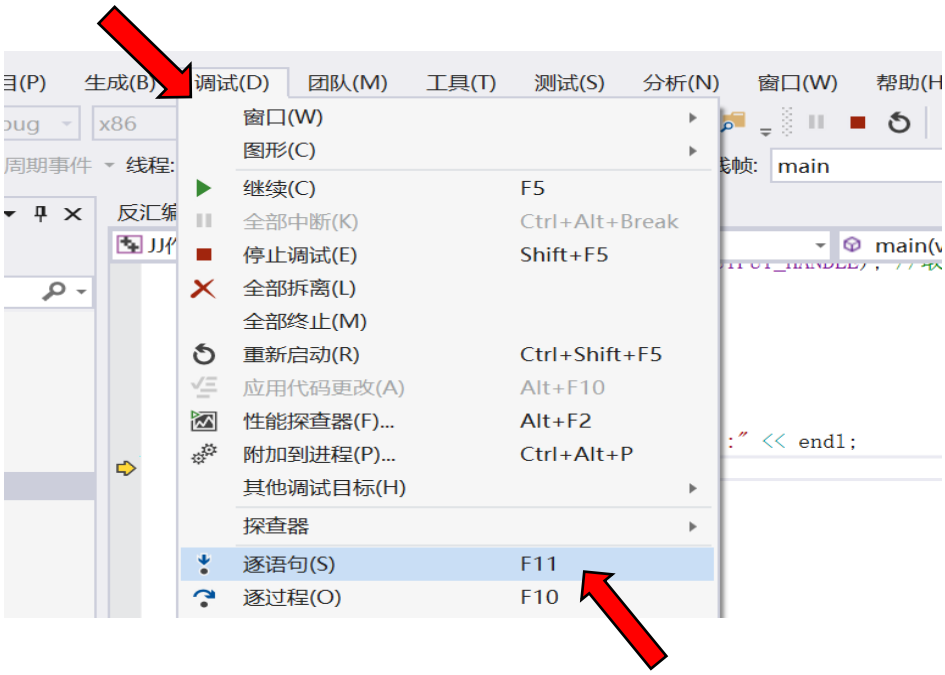
1.5. 在碰到自定义函数的调用语句，如何一步完成？

选择菜单，选择“调试”中的逐过程，或按 F10。



1.6. 在碰到自定义函数的调用语句，转到被调用函数单步执行？

选择菜单，选择“调试”中的逐语句，或按 F11.



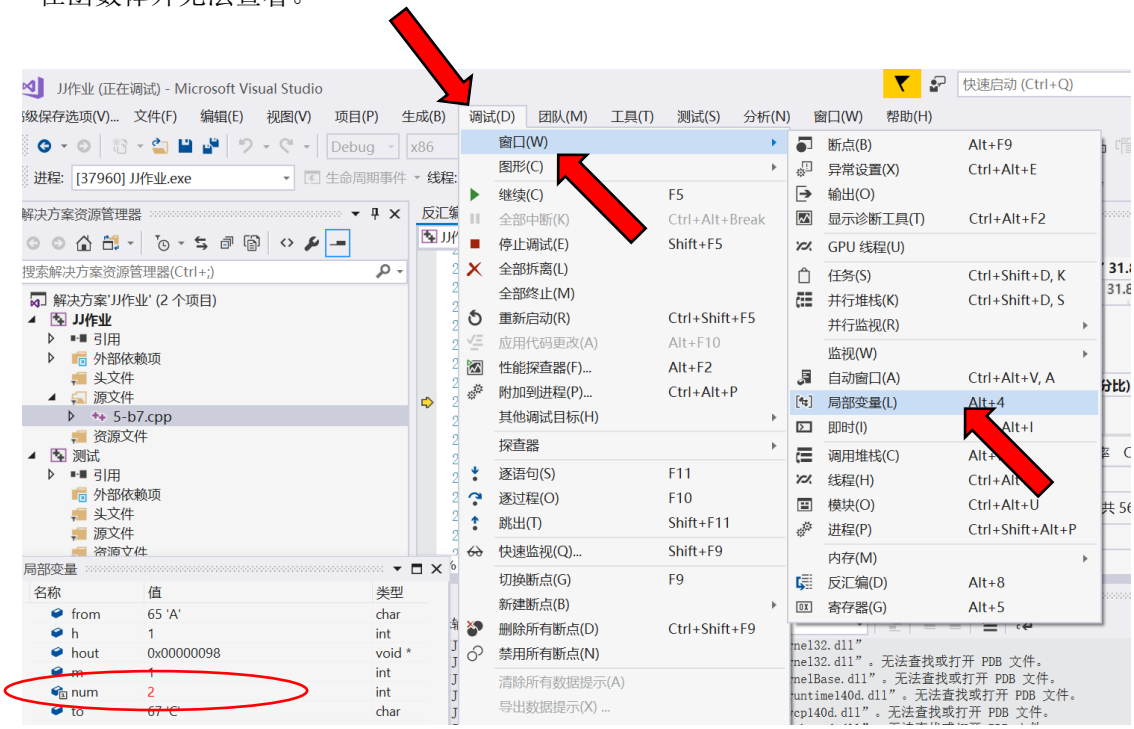
2. 掌握用 VS2017 的调试工具查看各种生存期、作用域变量的方法

2.1. 查看形参/自动变量的变化情况

选择菜单，选择“调试”中的“窗口”中的“局部变量”，或按 Alt+4

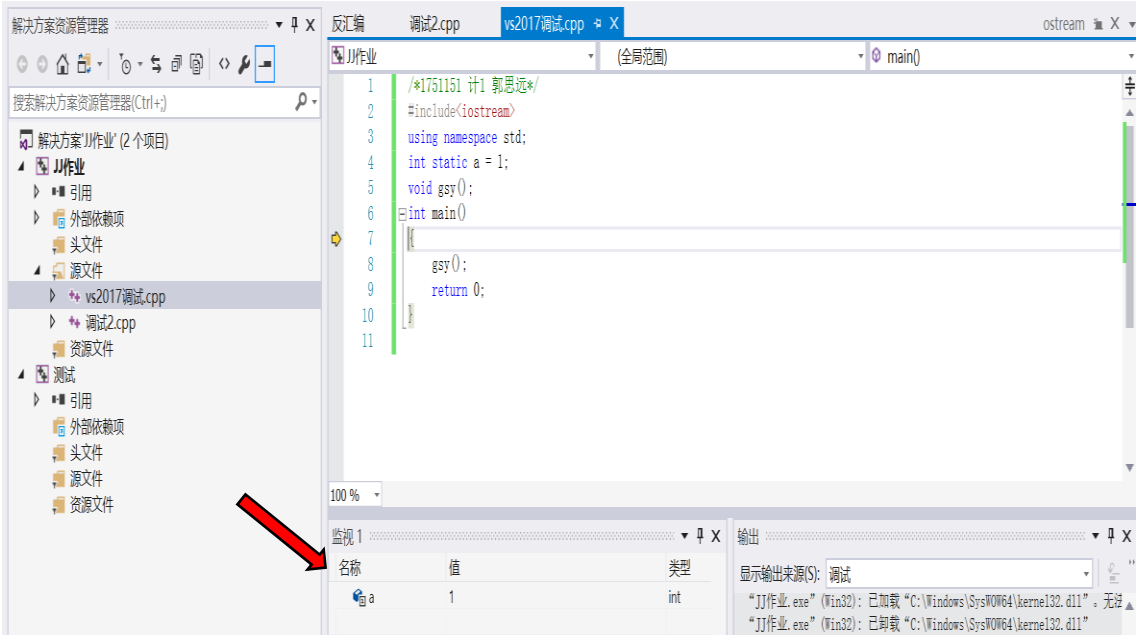
2.2. 查看静态局部变量的变化情况

在函数体内：选择菜单，选择“调试”中的“窗口”中的“局部变量”，或按 Alt+4
在函数体外无法查看。

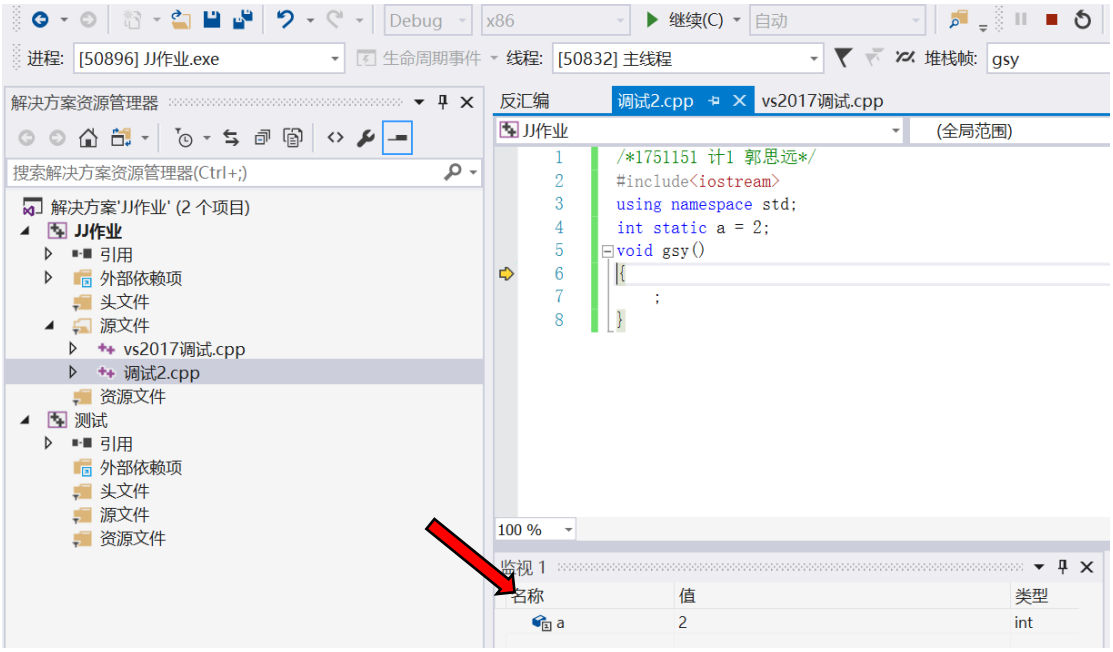


2.3. 查看静态全局变量的变化情况

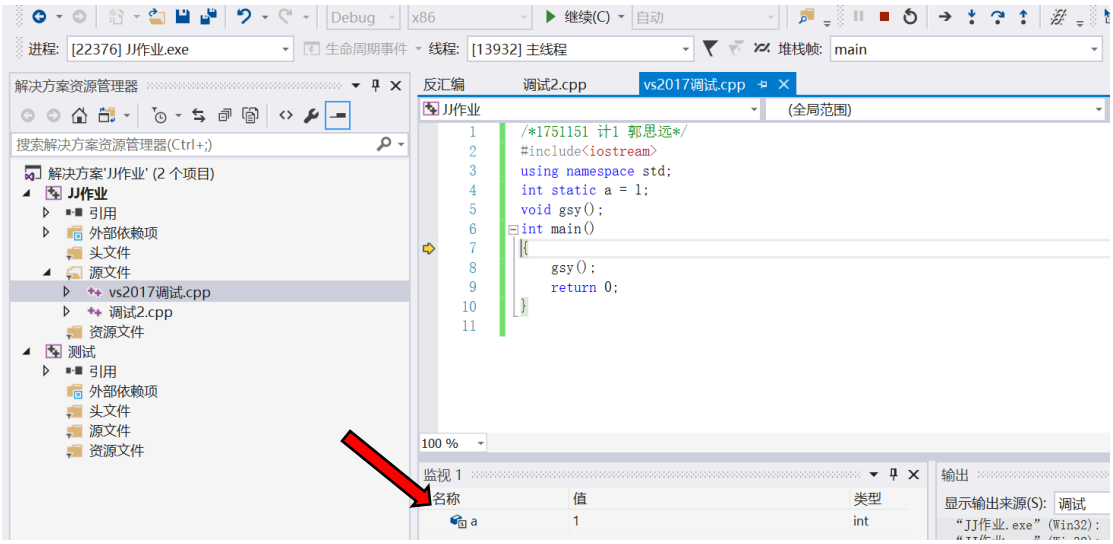
初始，在监视 1 中输入变量名来查看
在一个 cpp 里静态全局变量 a 的值为 1



在另一个 cpp 里 a 的值为 2。

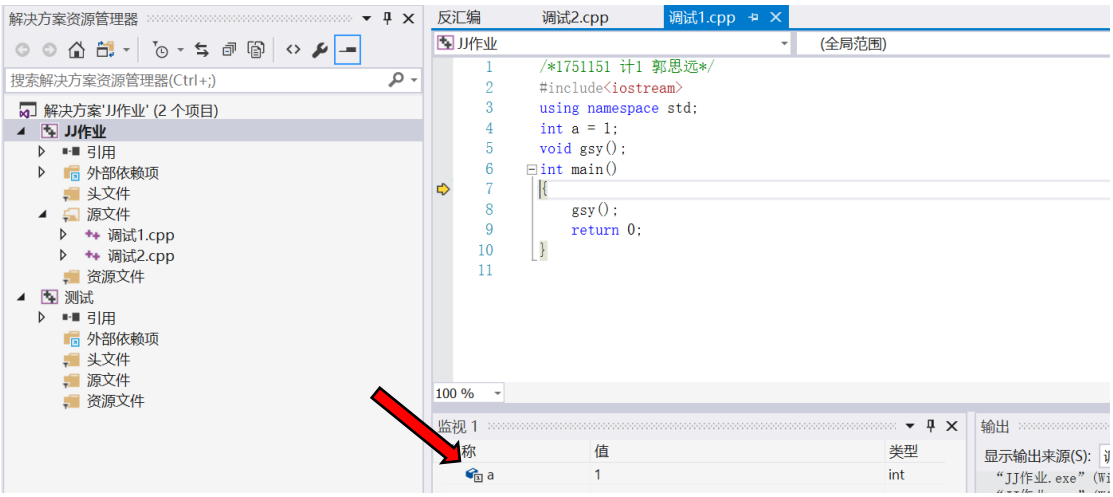


回到原来的 cpp 中，a 值变为 1。

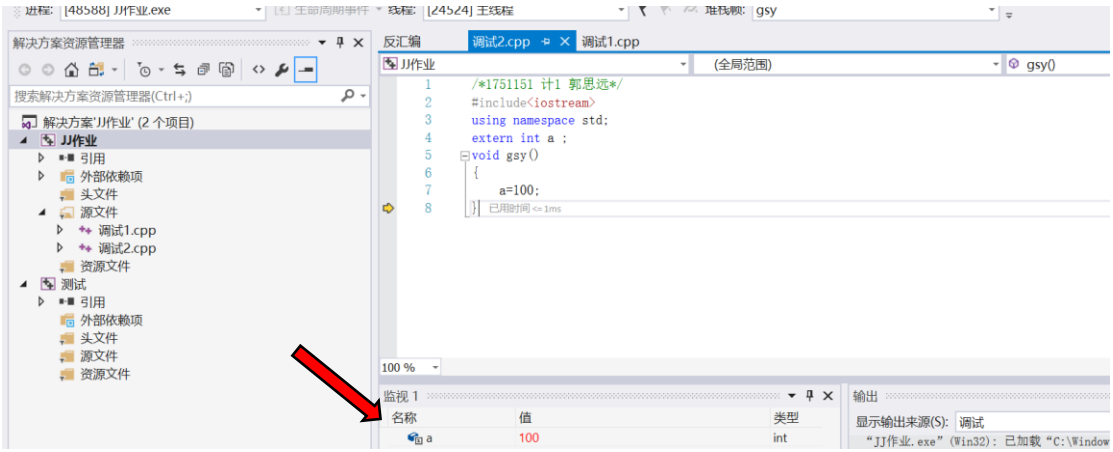


2.3. 查看外部全局变量的变化情况

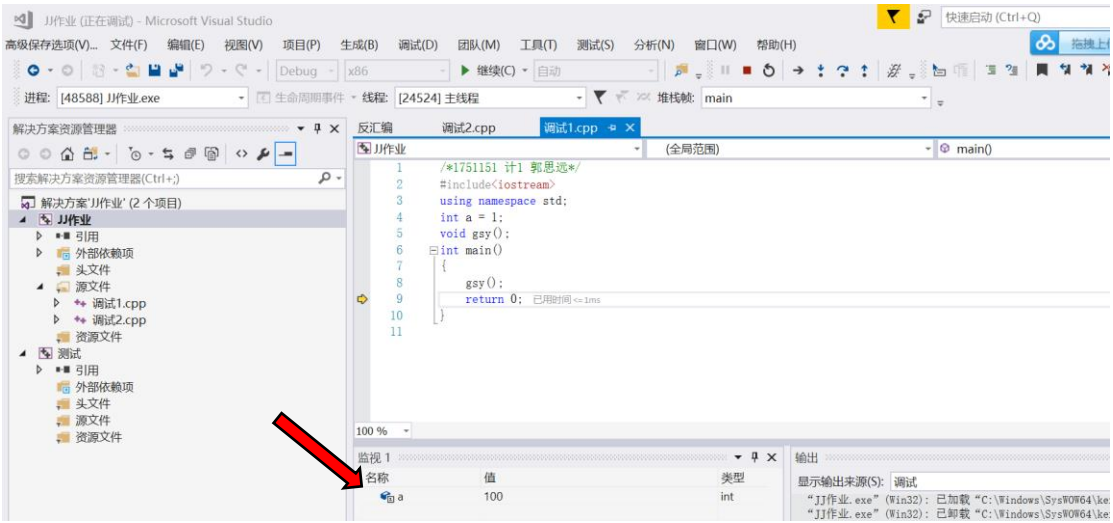
初始，在监视 1 中输入变量名来查看。a 为 1。



在另一个 cpp 中 a 的值被改为 100。



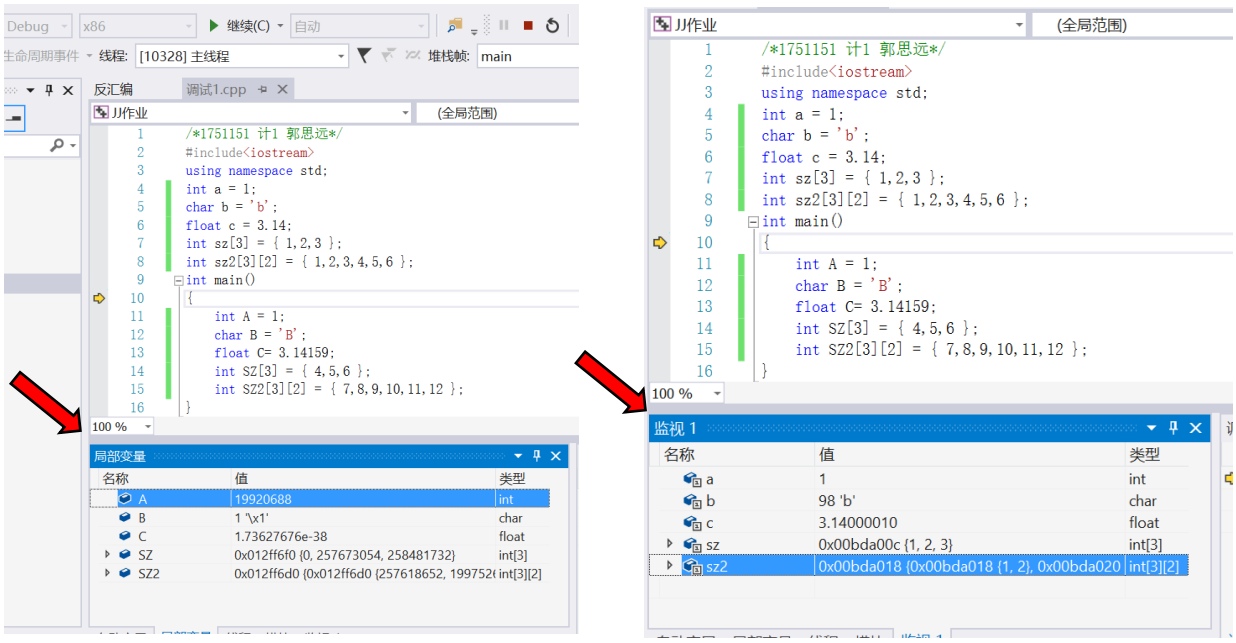
返回原来 cpp 后 a 的值仍为 100。



3. 掌握用 vs2017 的调试工具查看各种不同类型变量的方法。

3.1. char/int /float 等简单变量，一维数组，二维数组

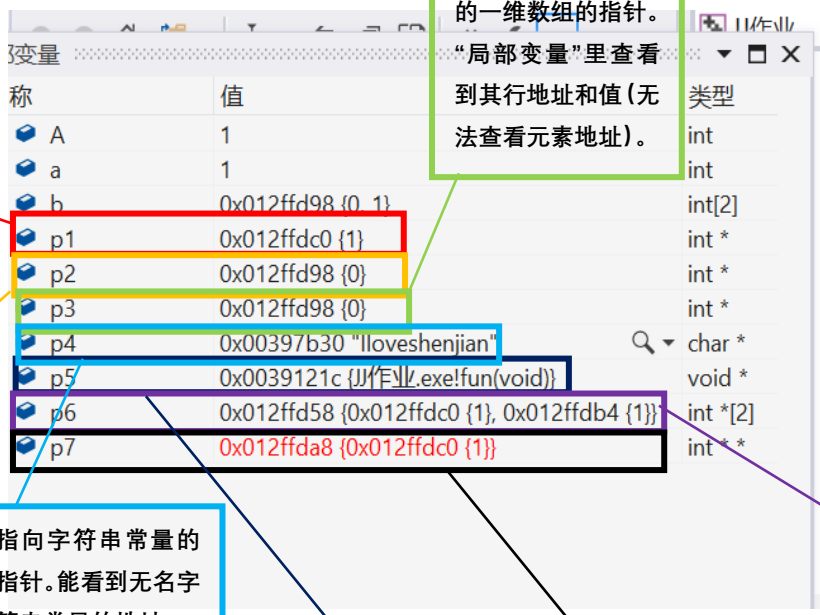
局部变量在局部变量窗口查看，全局变量在监视 1 中输入查看。



3.2. 指针

指向简单变量的指针。“局部变量”里查看到其地址和值。

指向 m 个元素组成的一维数组的指针。“局部变量”里查看到其行地址和值(无法查看元素地址)。




```
/*1751151 计1 郭思远*/
#include<iostream>
using namespace std;
void fun();
int main()
{
    int a = 1;
    int A = 1;
    int *p1 = &a;
    int b[2] = { 0,1 };
    int *p2 = &b[0];
    int *p3 = b;
    char *p4 = { "Iloveshenjian" };
    void *p5 = fun;
    int *p6[2] = { &a, &A };
    int **p7 = &p1;
```

指向一维数组的指针。“局部变量”里查看到其地址和值。

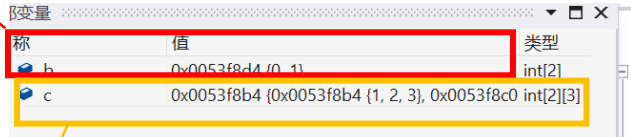
指向字符串常量的指针。能看到无名字符串常量的地址

指向函数的指针。能看到函数的地址

指向指针的指针。能看到函数的地址。“{}”前为二级指针，“{}”内分别为一级指针和值。

指针数组。在“{}”前看到数组自身的地址，在“{}”内看到数组中元素的值。

实参是一维数组名，形参是指针。在“局部变量”中查看实参数组的地址和值




```
void fun1(int *p1);
void fun2(int(*a)[3]);
int main()
{

    int b[2] = { 0,1 }, c[2][3] = {1,2,3,4,5,6};
    fun1(b);
    fun2(c);

    return 0;
}
void fun1(int *p1)
{
    ;
}
void fun2(int (*a)[3])
```

实参是二维数组名，形参是行指针。在“{}”外的是 2 元素一维数组首地址，“{}”内是行地址和值。

3.3. 引用与指针的区别

有区别。

区别：

1. 引用是指针常量，它的指向不能改变，只能指向一个指定的变量。
2. 声明一个引用必须同时使之初始化，即声明它代表哪一个变量。指针声明时可以不初始化。
3. 引用不能为空，指针可以为空。
4. 不能建立引用的引用可以建立指针的指针。
5. 不能建立引用数组，可以建立指针数组。
6. 对引用，只有声明，没有定义。指针有定义。

3.4. 函数的形参为数组的指针/引用时的区别

有区别。

区别：指针方式传递的是实参的值，引用方式传递的是实参变量的地址。

3.5. 使用指针时出现了越界访问。

指针越界访问了，运行弹窗。