# Text2 两数相加求和

1. **设计程序的逻辑思路**

本实验采用C++的程序风格，通过定义声明函数sum，传递形参到主函数内实现功能。其次是本次上机实验中，通过一个简单的程序，练习了.h文件在C++中的应用。在C++的应用中，如果存在很多声明函数，如果把它放在主函数里面，显得太多太繁琐，所以需要事先在.h文件里面声明所定义的函数，这便是.h文件的应用。

1. **程序主体代码**

#include<iostream>

#include<conio.h>

using namespace std;

int add(int a,int b);

int main()

{

int x,y,sum;

cout<<"enter two numbers:"<<endl;

cin>>x>>y;

sum=add(x,y);

cout<<x<<"+"<<y<<"="<<sum<<endl;

getch();

return 0;

}

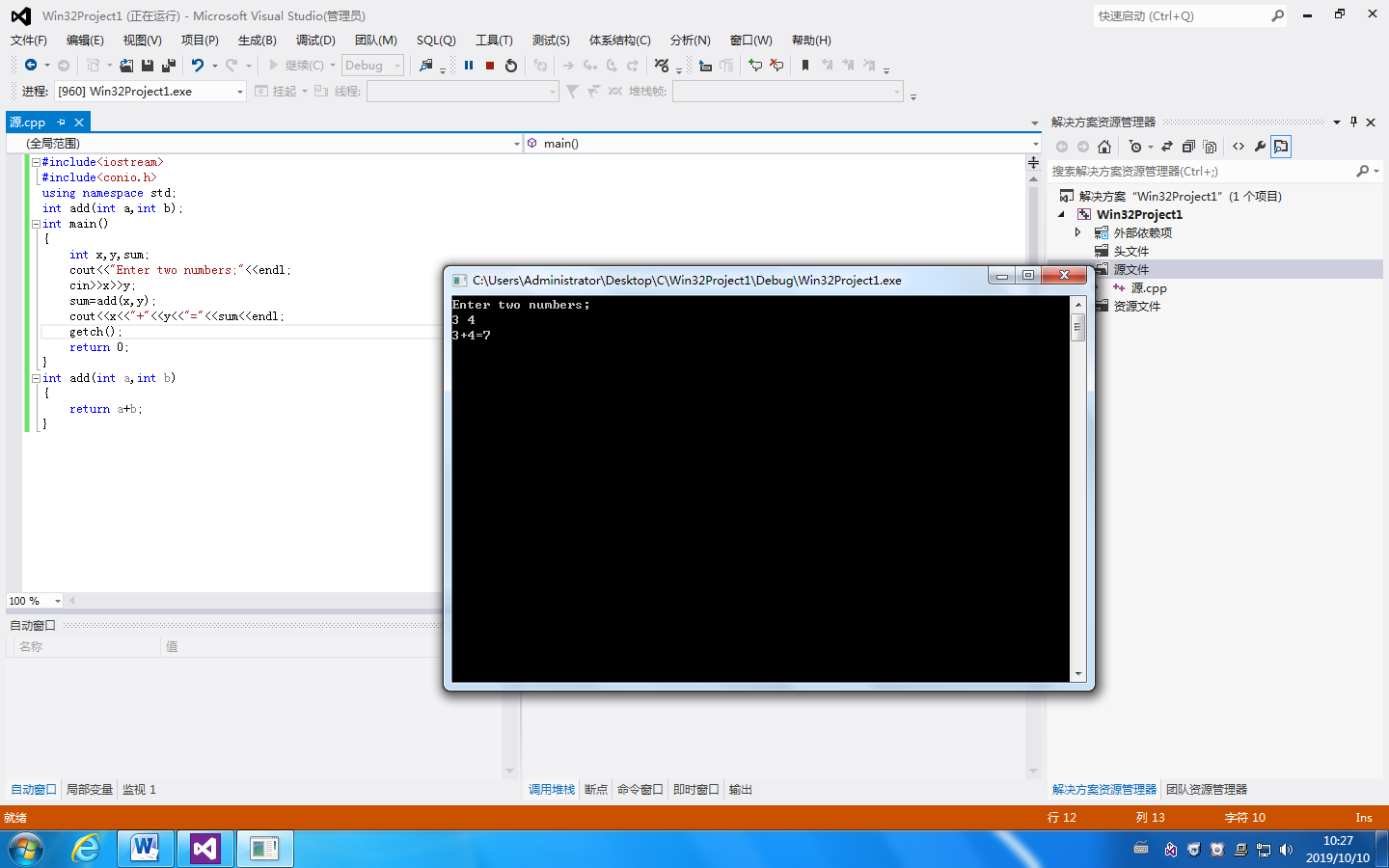
int add(int a,int b)

{

return a+b;

}

其输出显示如图所示：



1. **在设计程序时产生的错误及解决方法**

上述程序设计是只在源文件里面写程序，下面换一种程序设计方法，通过多文件函数

实现程序的求和。程序代码如下：

main.cpp里面写主函数程序：

#include<iostream>

#include"add.h"

#include<conio.h>

using namespace std;

int main()

{

int x,y,sum;

cout<<"Enter two numbers"<<endl;

cin>>x>>y;

sum=add(x,y);

cout<<x<<"+"<<y<<"="<<sum<<endl;

getch();

return 0;

}

add.cpp里面写定义的函数：

#include"add.h"

int add(int a,int b)

{

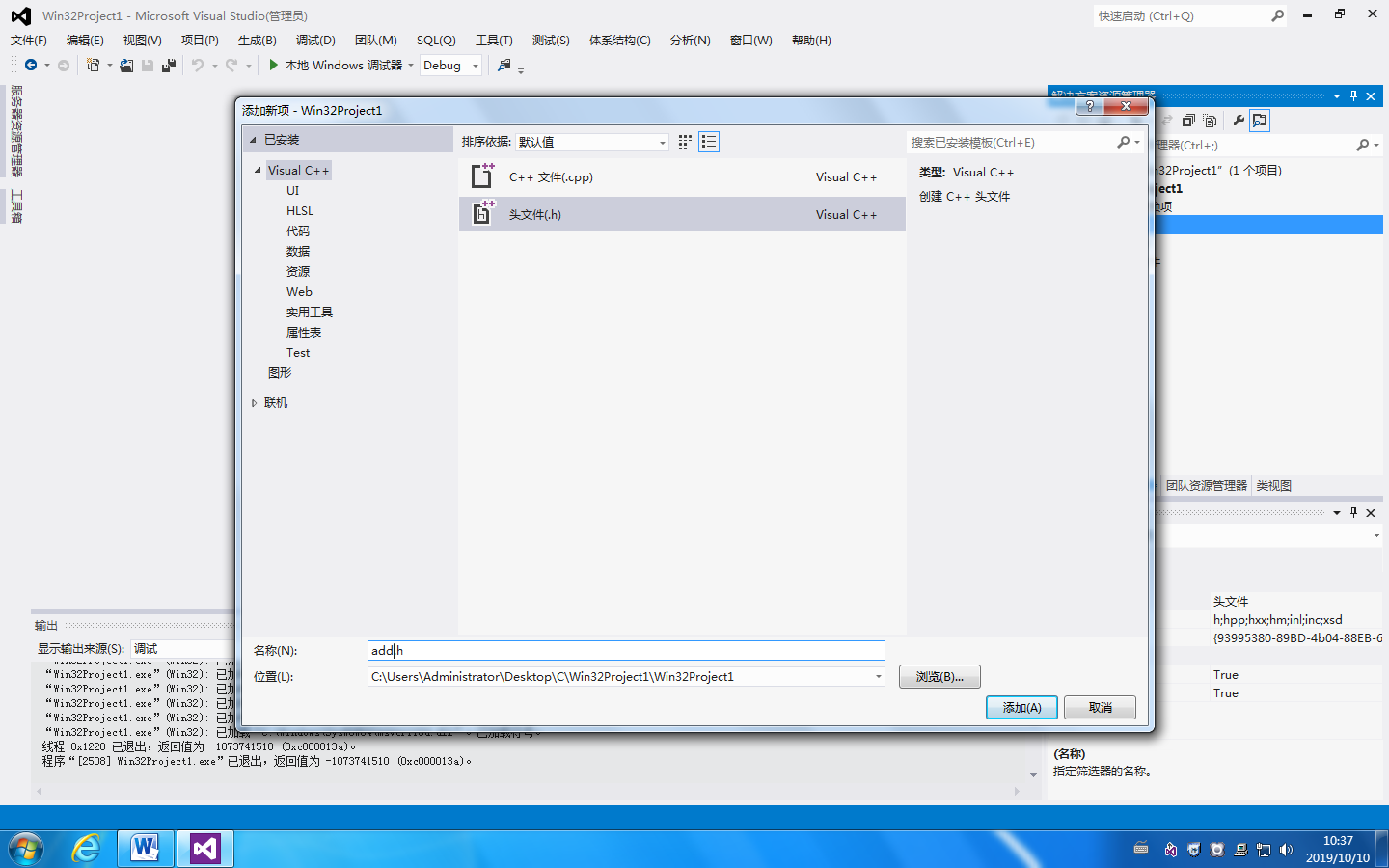
return a+b;

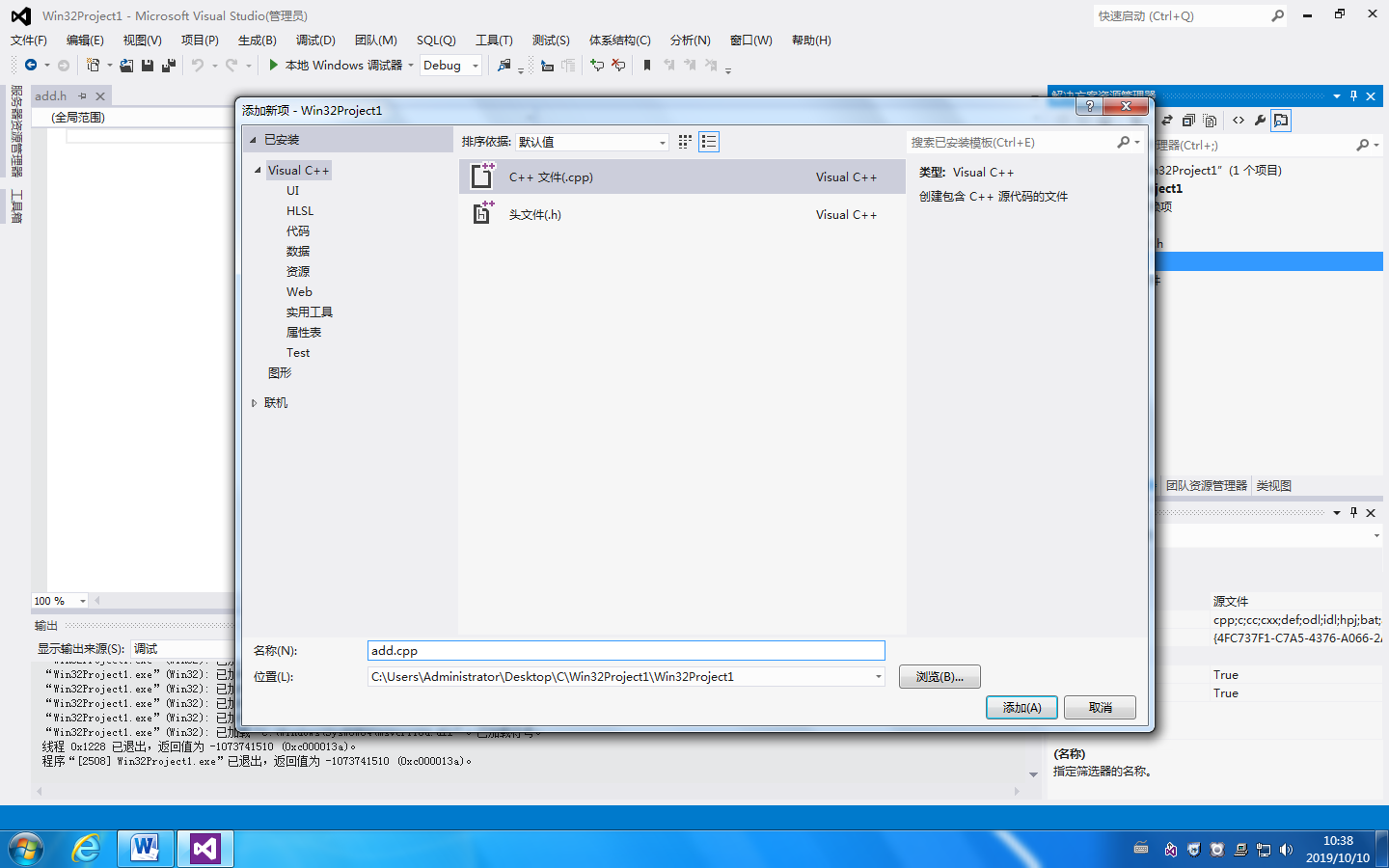
}

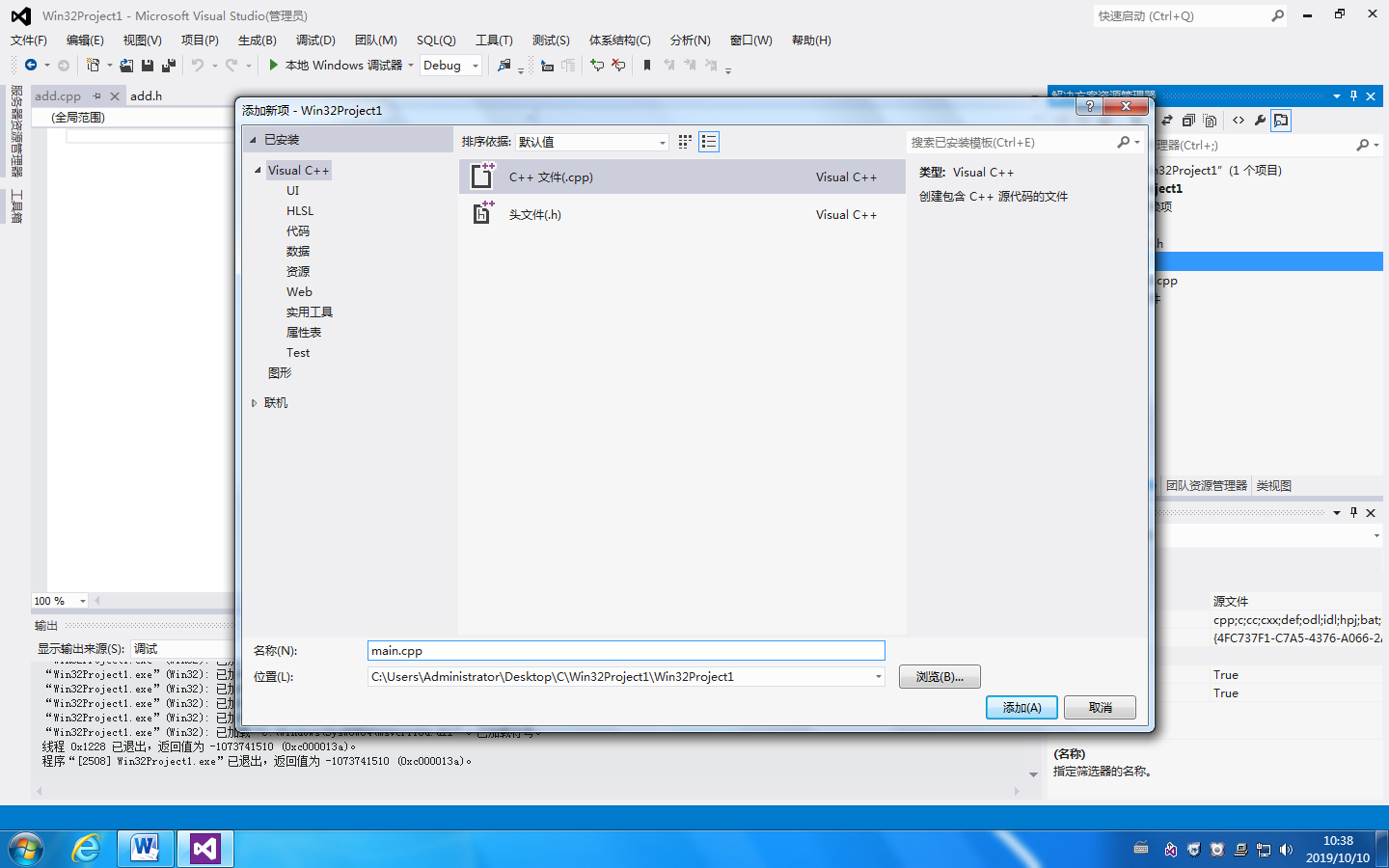
add.h里面写声明函数add：

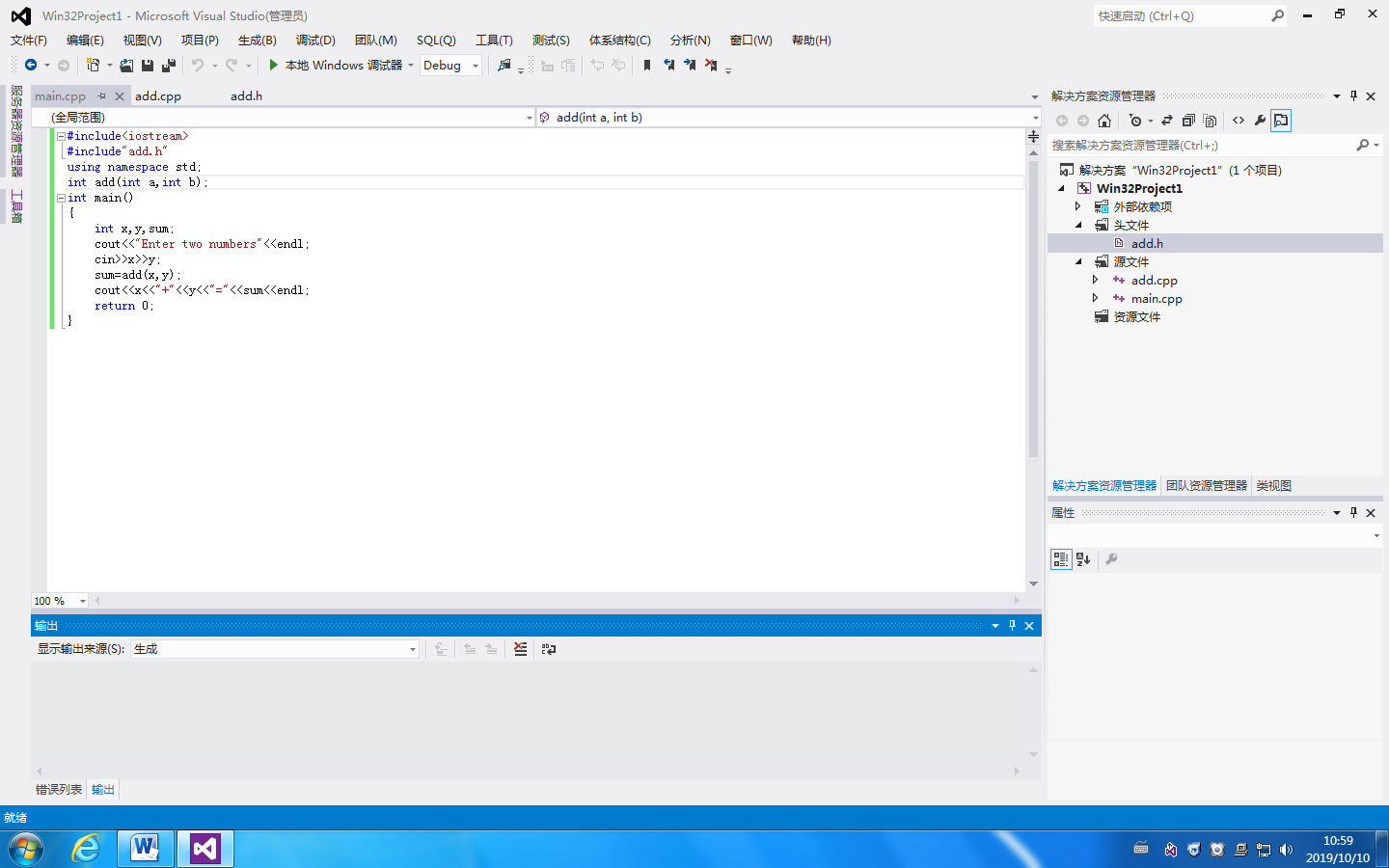
int add(int a,int b);

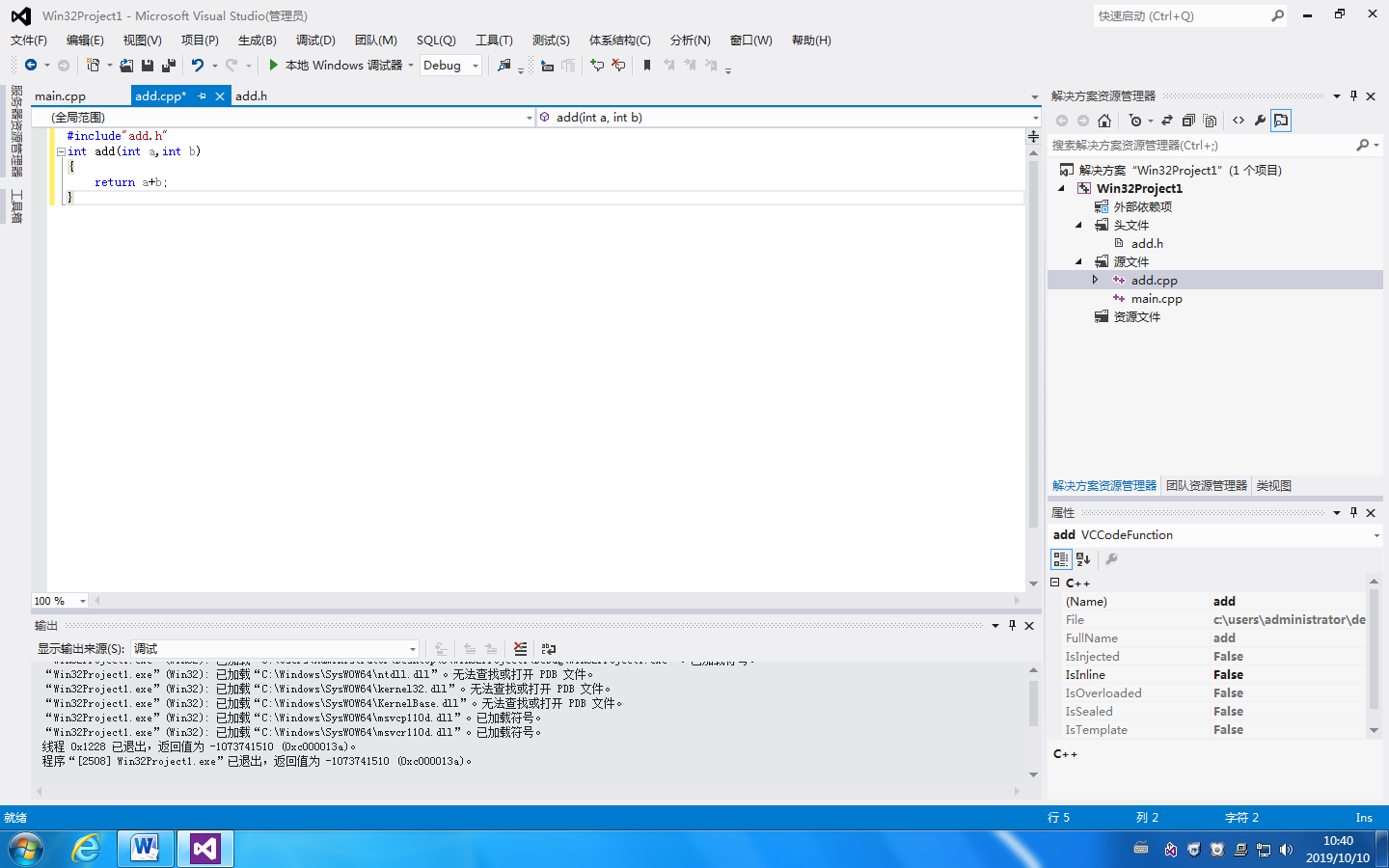
程序运行结果如下图所示：

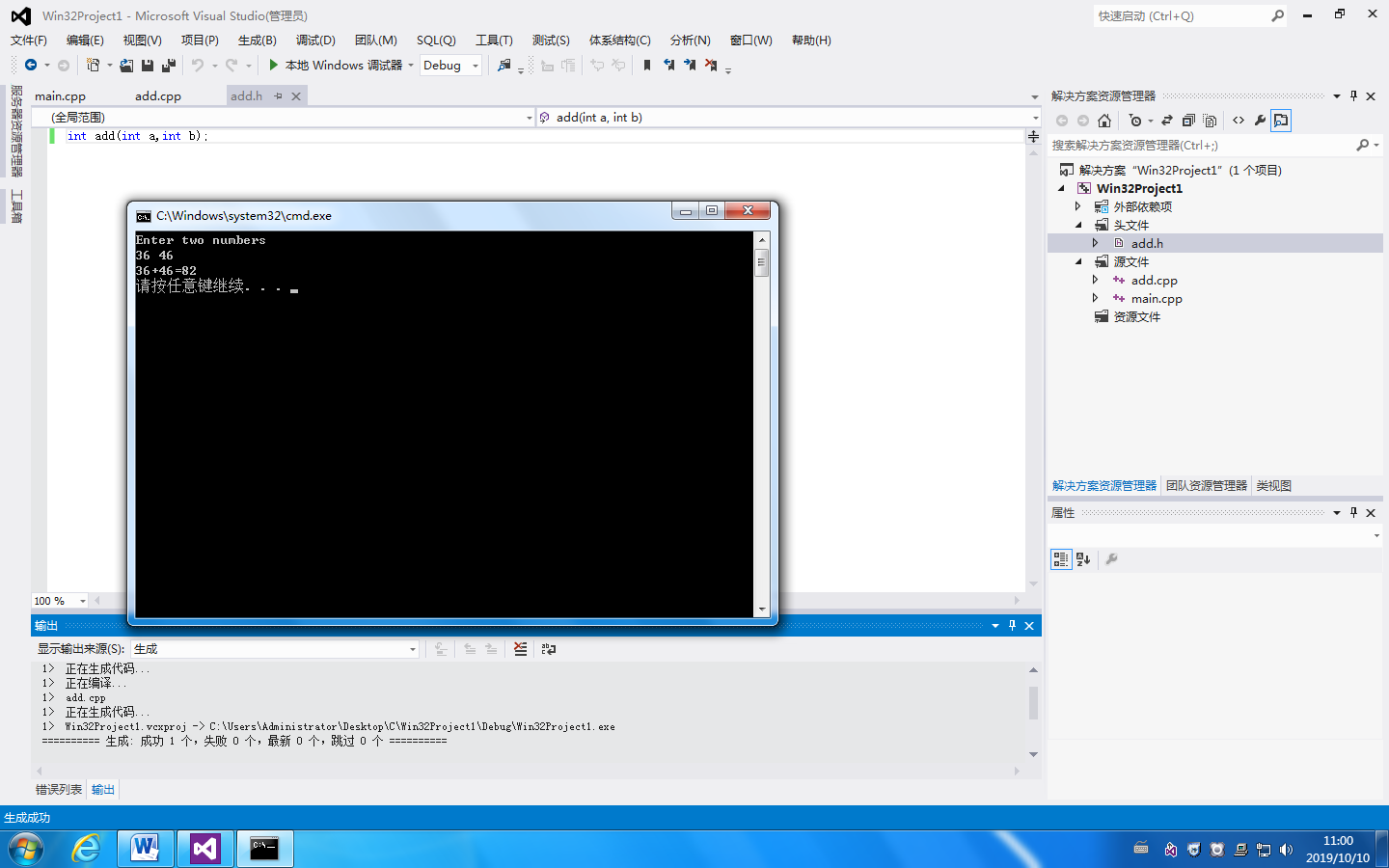




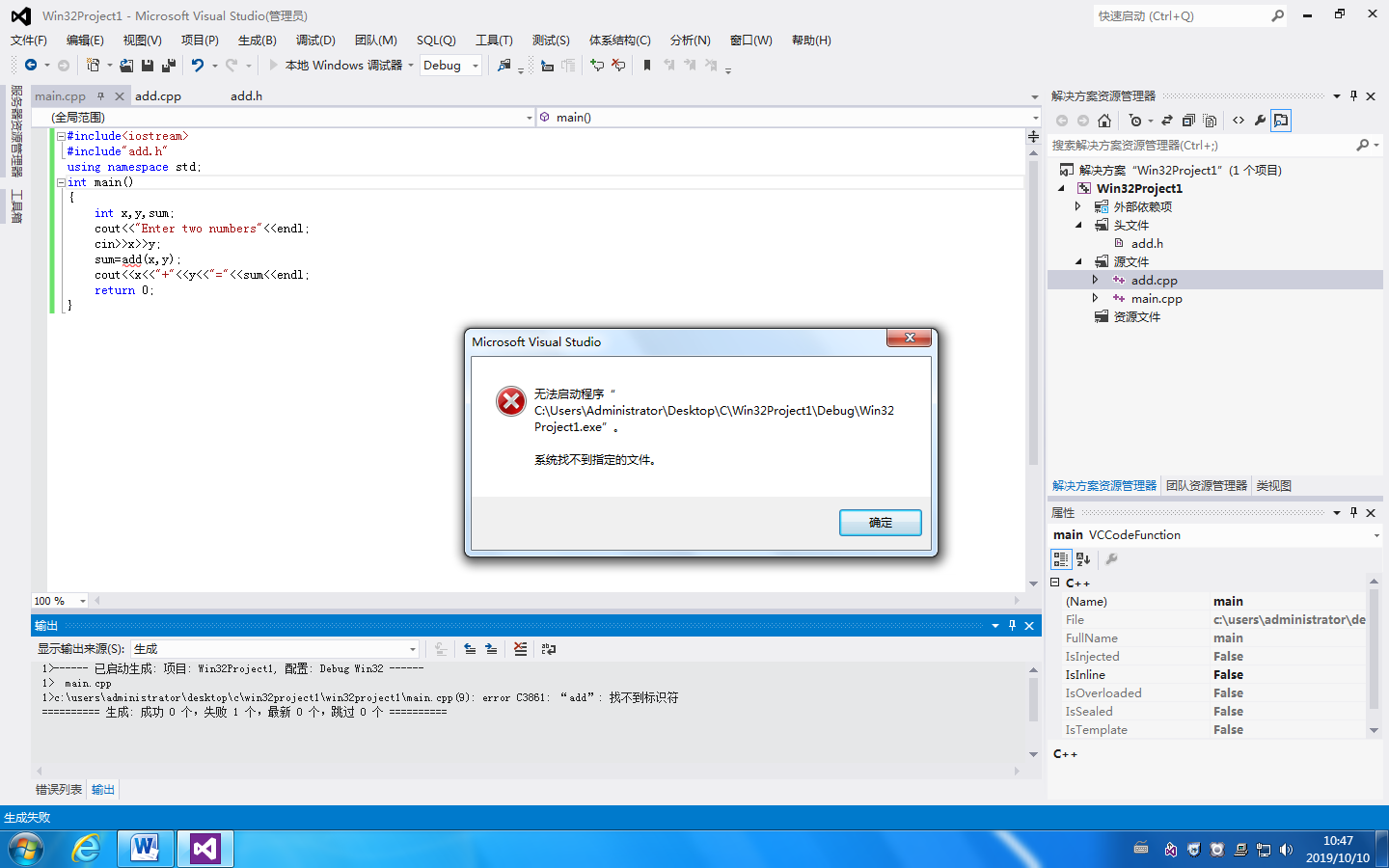








1. 这只是在开始执行不调试的情况下显示的，在调试的时候出现闪退，原因是没
2. 有添加头文件#include<conio.h>和主函数里的getch();
3. 没有在add.h里面加入声明，结果如下图所示：

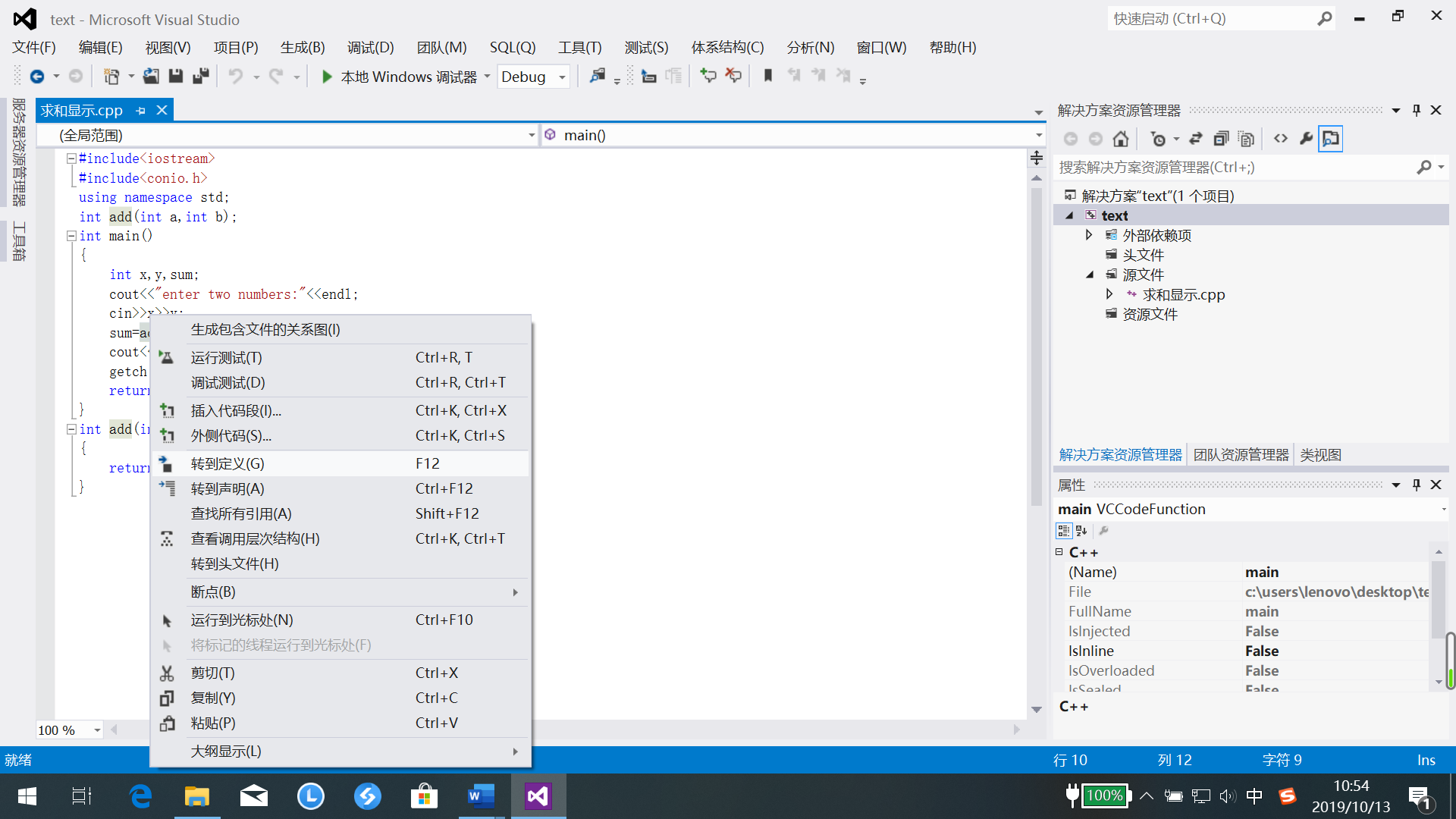


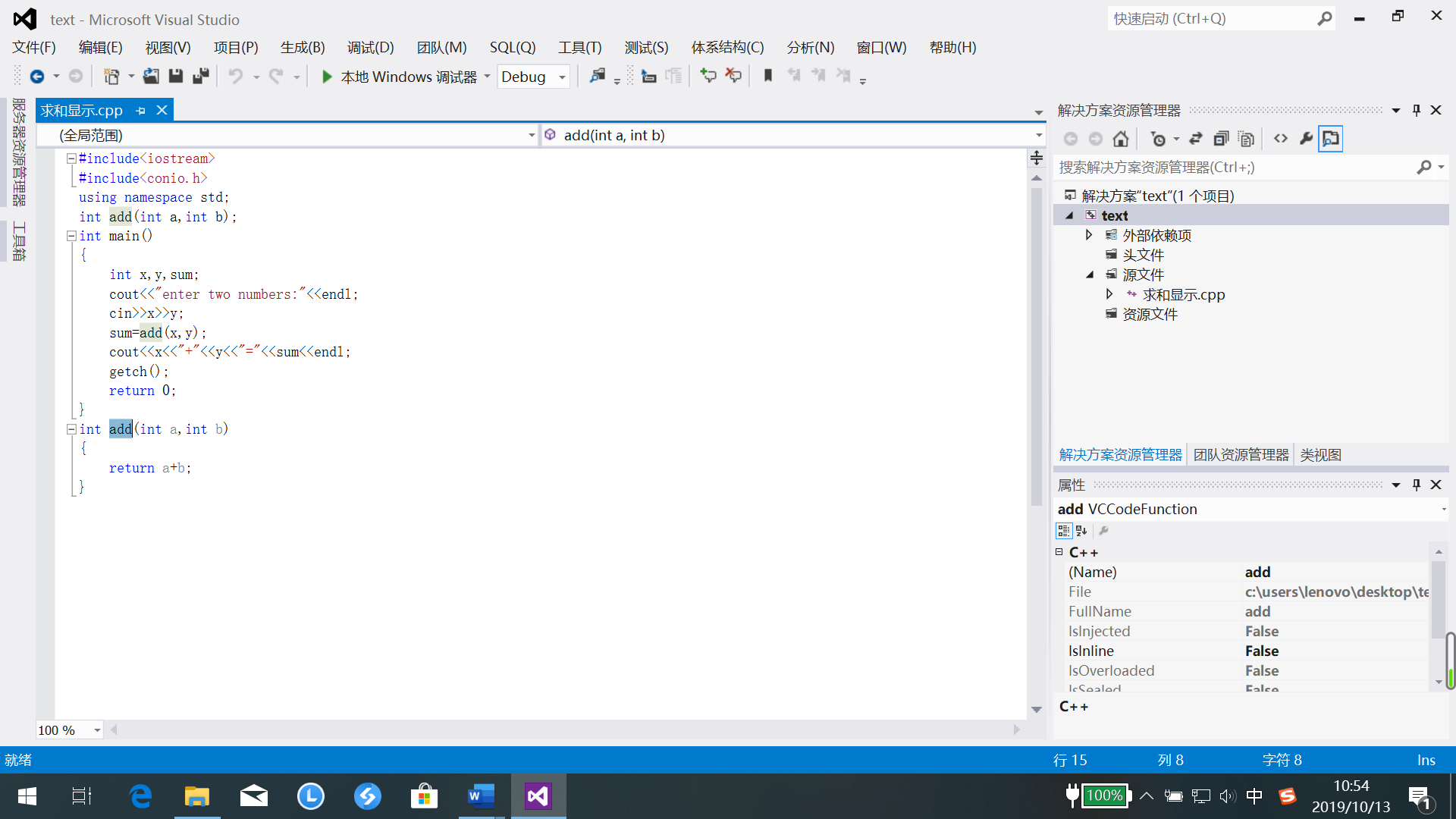
**解决方法：**

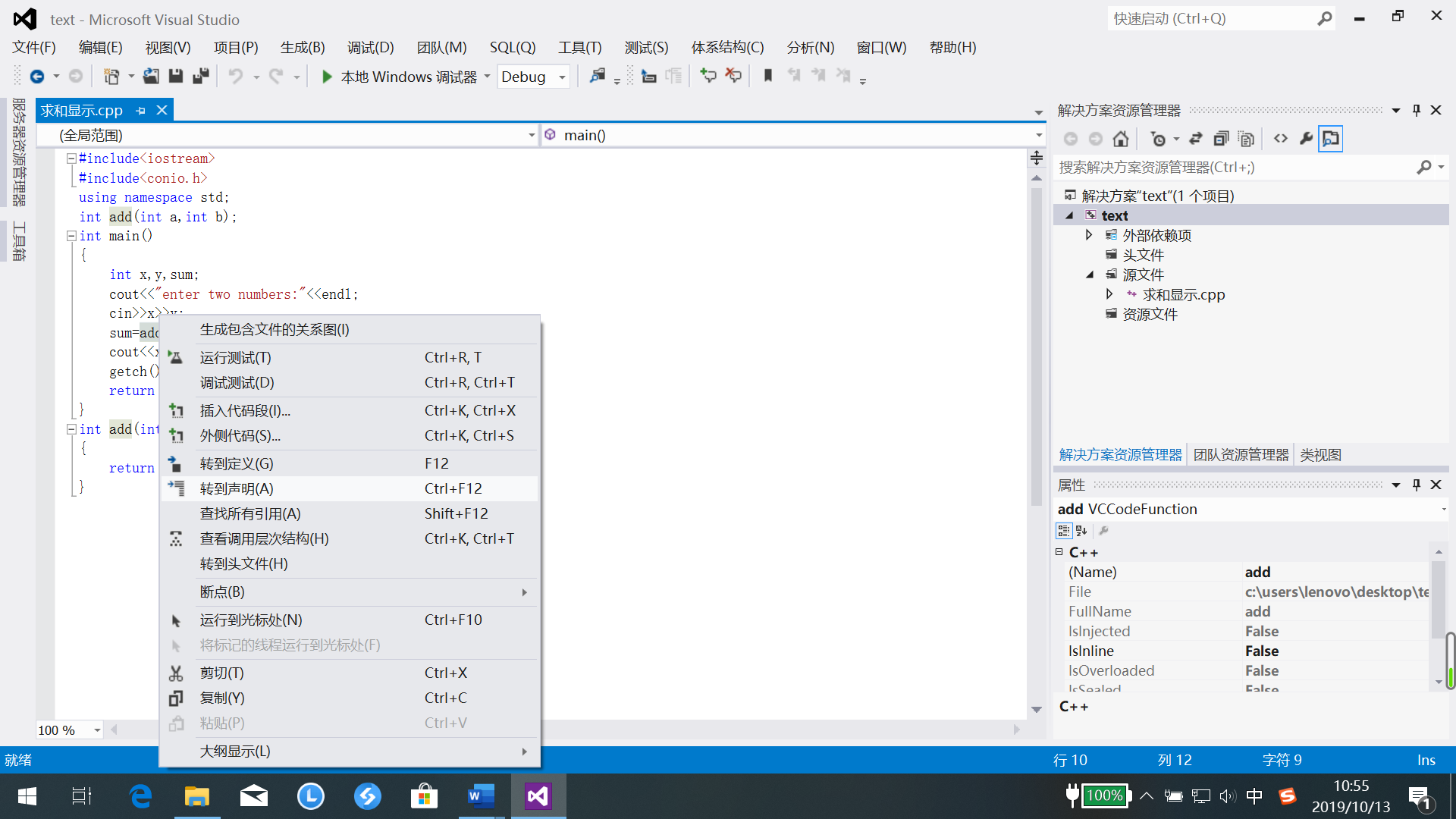
头文件一般含全局量，[函数原型](http://www.so.com/s?q=%E5%87%BD%E6%95%B0%E5%8E%9F%E5%9E%8B&ie=utf-8&src=internal_wenda_recommend_textn" \t "_blank)，宏等。[函数](http://www.so.com/s?q=%E5%87%BD%E6%95%B0&ie=utf-8&src=internal_wenda_recommend_textn)的具体定义放在cpp里。不含main的 cpp们，可以先单独编译成 .obj 们。含main的 cpp 编译 时可以链接 编好的 .obj 。   
把 无main的cpp 作为 .h 当然可以。用 #include 的作用 是 把程序作为 文本 嵌入 #include 所在那行。作为整个[源程序](http://www.so.com/s?q=%E6%BA%90%E7%A8%8B%E5%BA%8F&ie=utf-8&src=internal_wenda_recommend_textn" \t "_blank)的一部分，一起编译。   
写一些单独的cpp 是为了方便自己阅读，查错。可以一块一块地 单独编译。add.cpp为定义函数add里面的具体内容，add.h为add函数的头文件，里面需要放add函数的声明。（函数必须声明之后才能定义）。

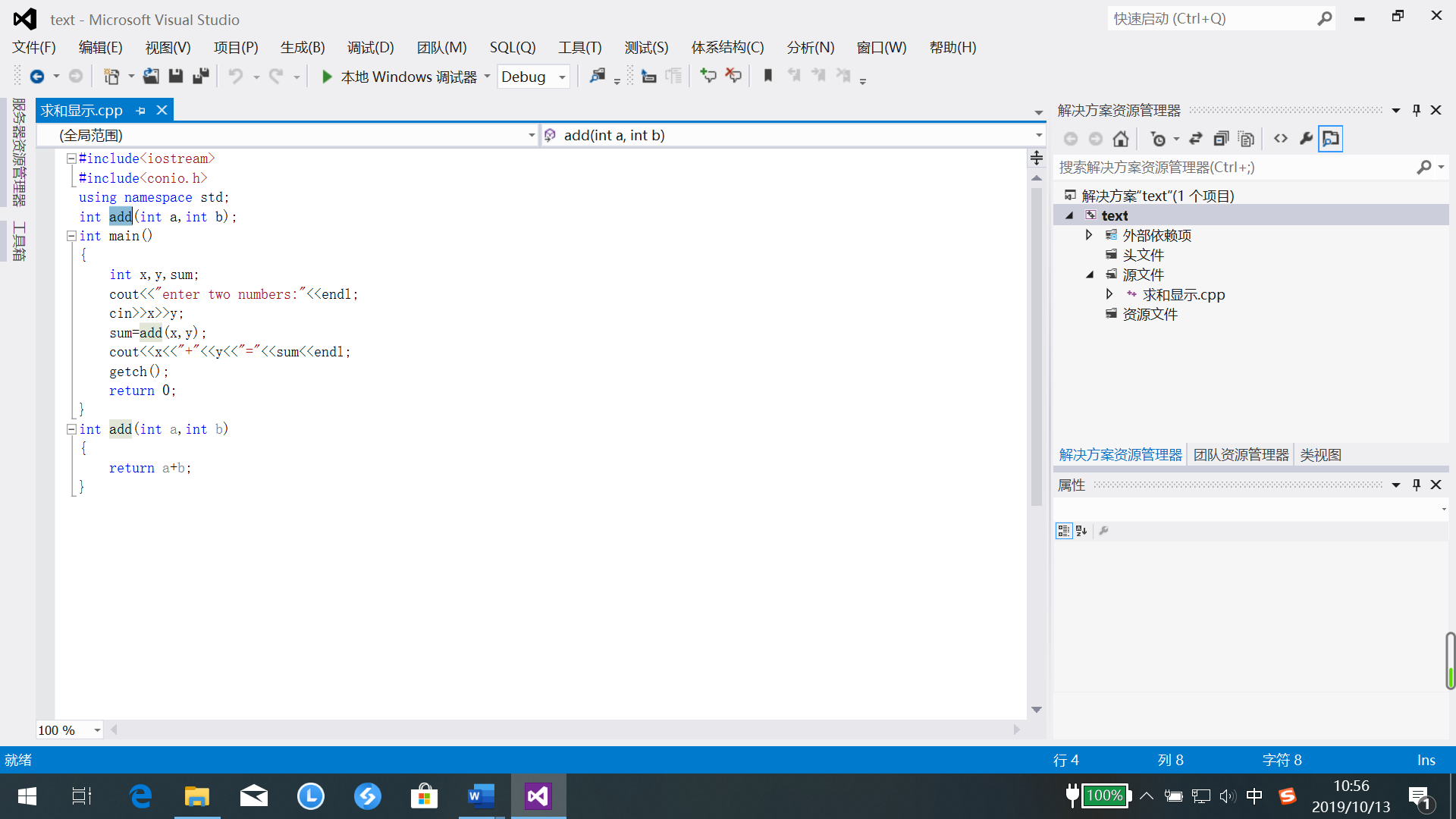
1. **思想感悟**

为了方便迅速查找定义的函数还是声明的函数，在任何一个定义函数add上右键，下拉菜单里面有个转到定义和转到声明，点击转到定义，光标就会立即显示在定义上，点击转到声明，光标就会立即显示到声明上。这是方便在复杂的程序中，能迅速找到具体操作如下图所示：







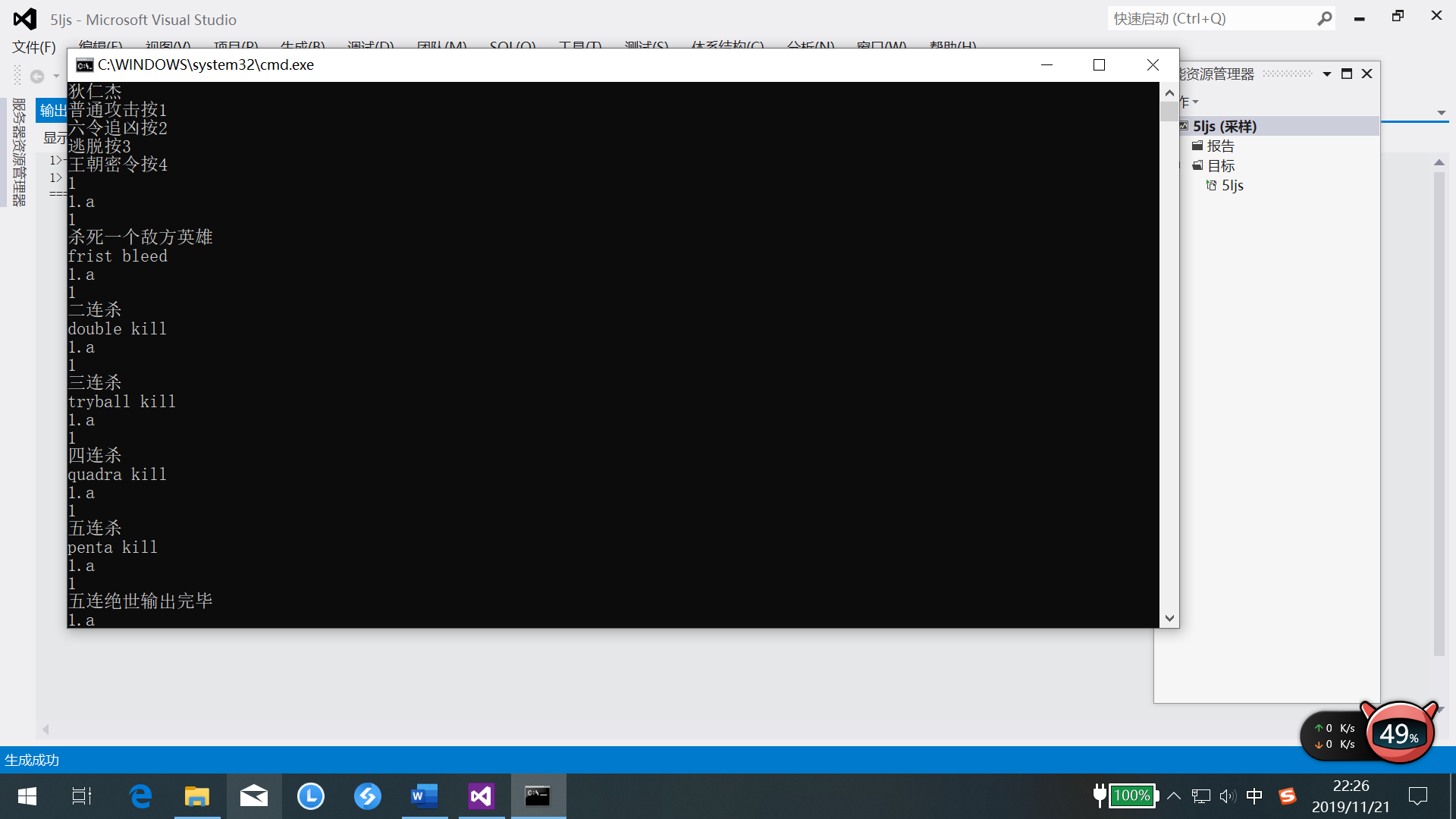


1. **上机作业：五连绝世的输出**

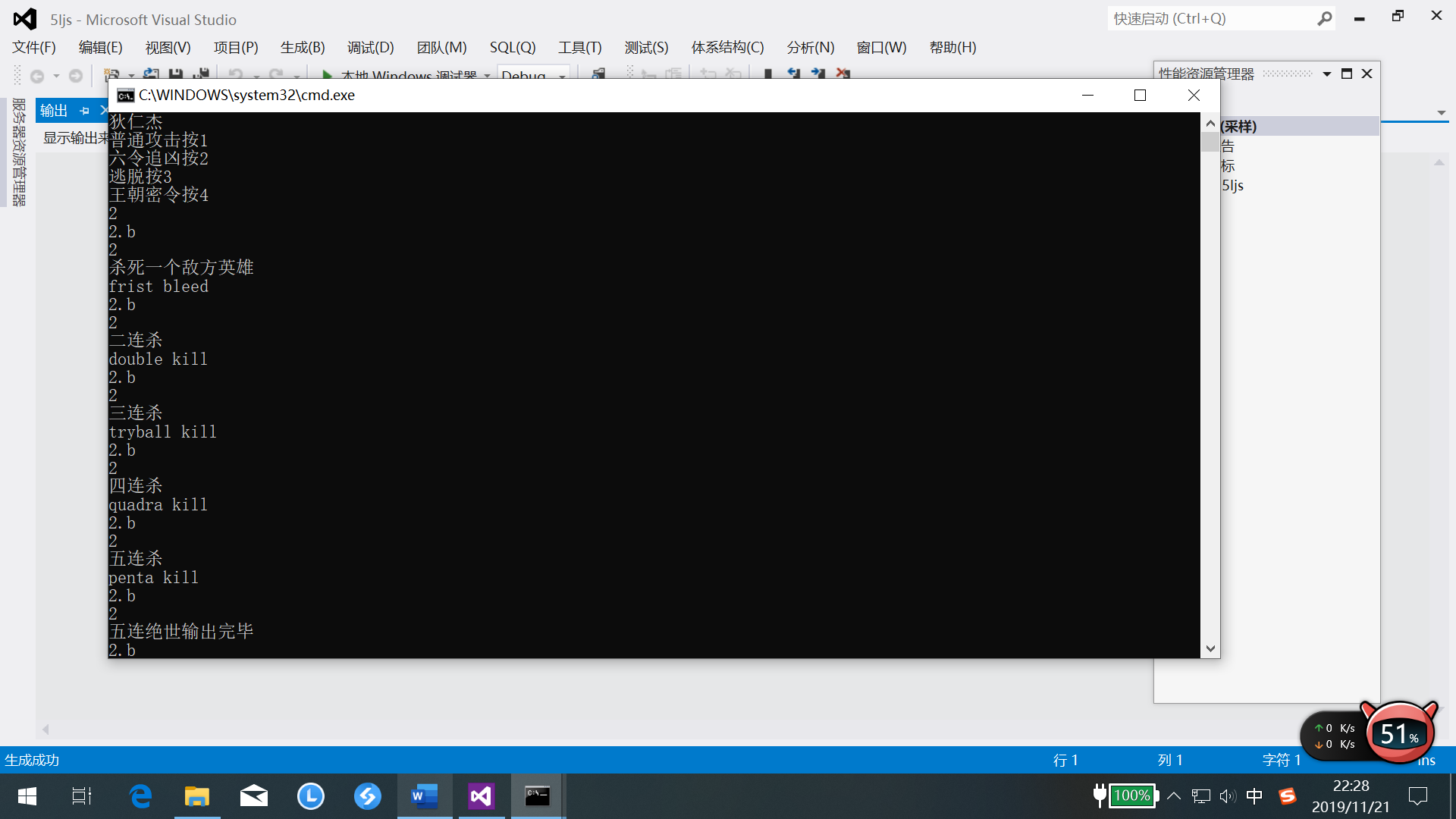
**程序附源代码**

程序的运行结果输出如下图所示：

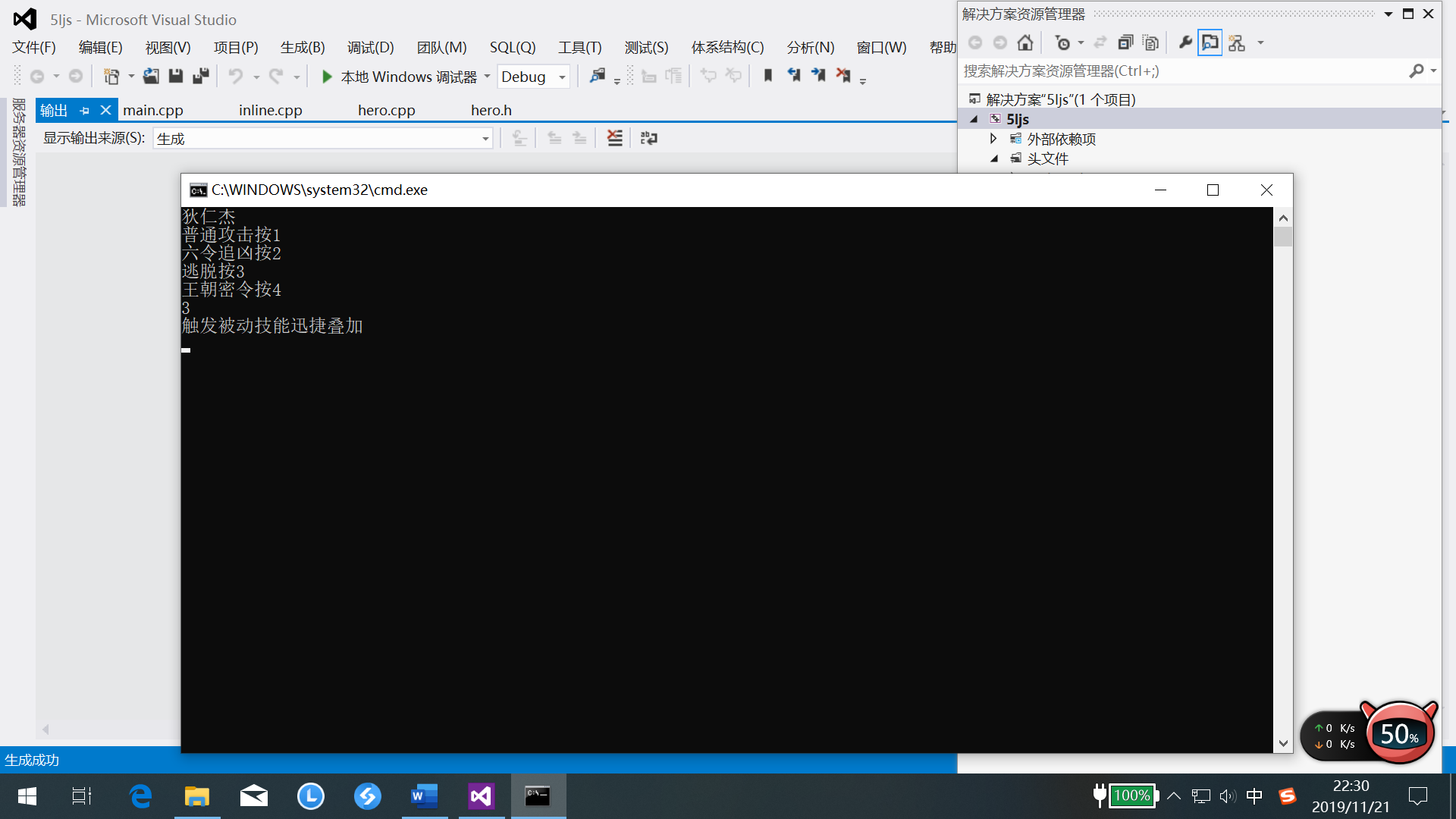
按普通攻击的时输出的效果：



按技能六令追凶时输出的效果：



按技能逃脱时输出的效果：



按技能王朝密令时输出的效果：

