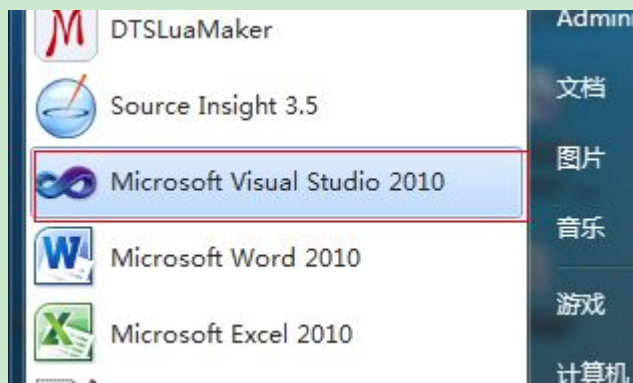


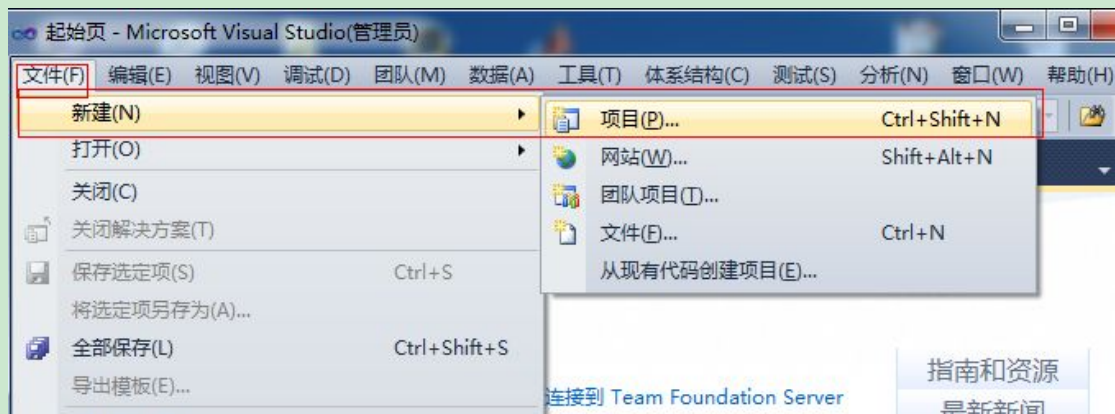
VS2010、VS2012 中 C++编写、调用 dll

一、编写生成 DLL 文件

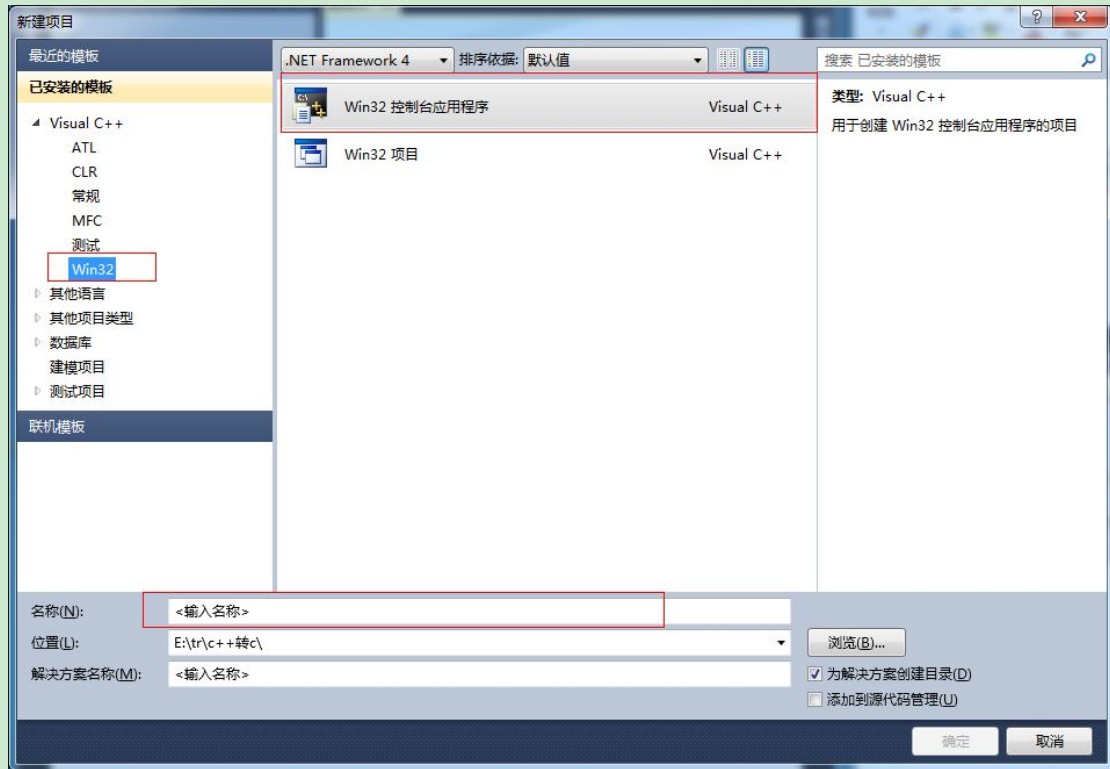
步骤一：打开 vs2010



步骤二：文件->新建->项目

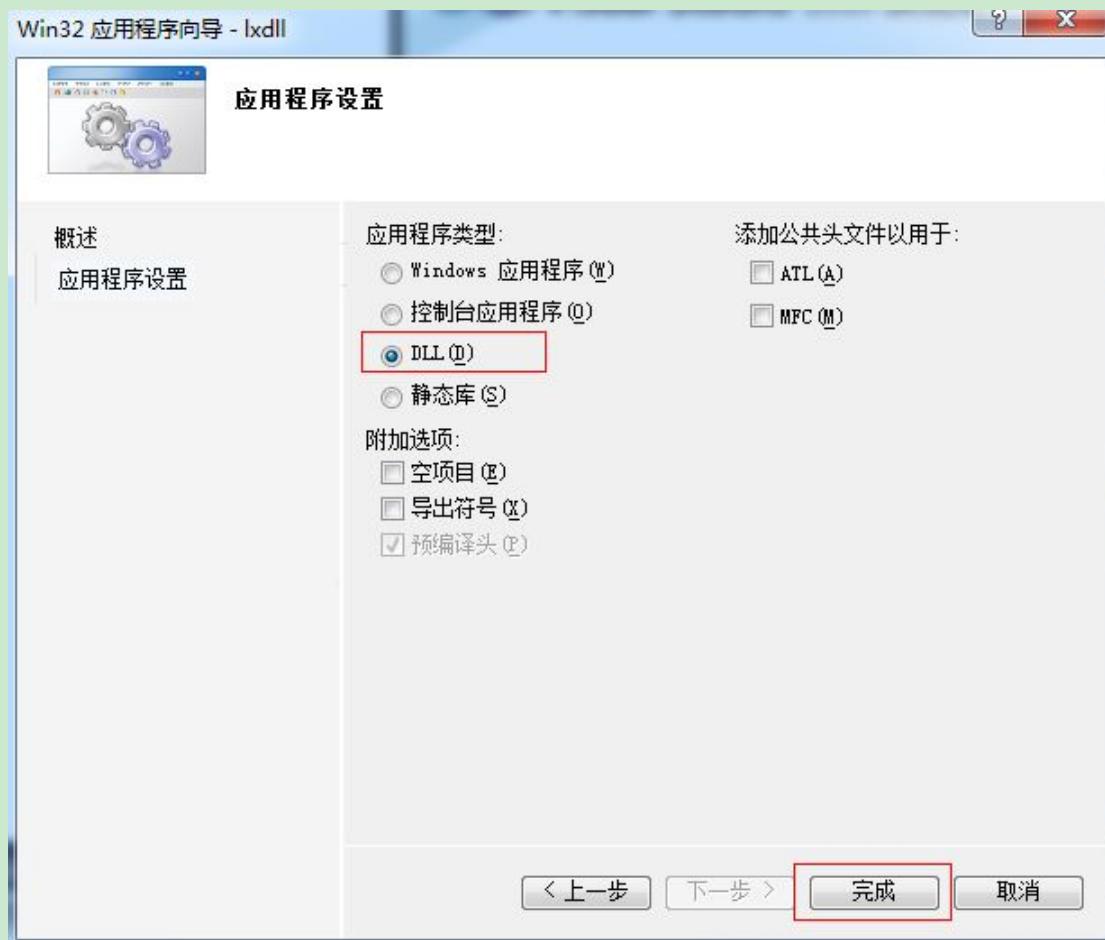


步骤三：点击 win32(win32 控制应用程序(名称输入你的工程名如我的
是 lxdll)——>确定

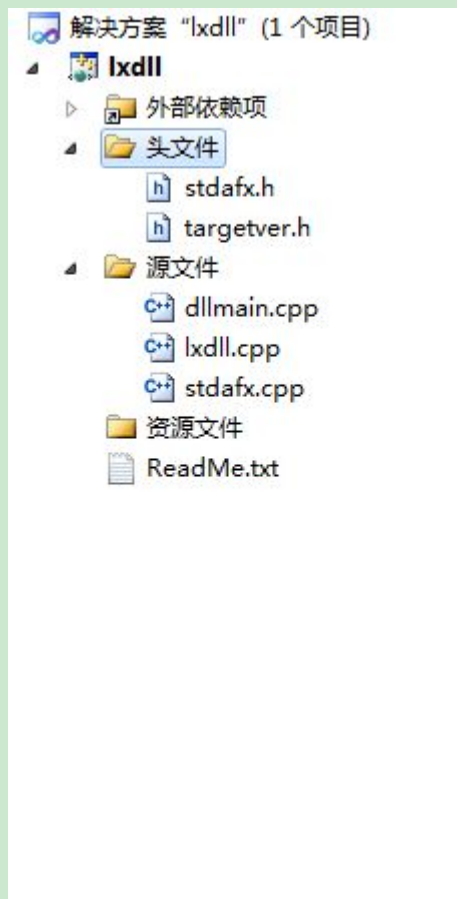


点击下一步

步骤四：在应用程序类型选择 DLL—>完成



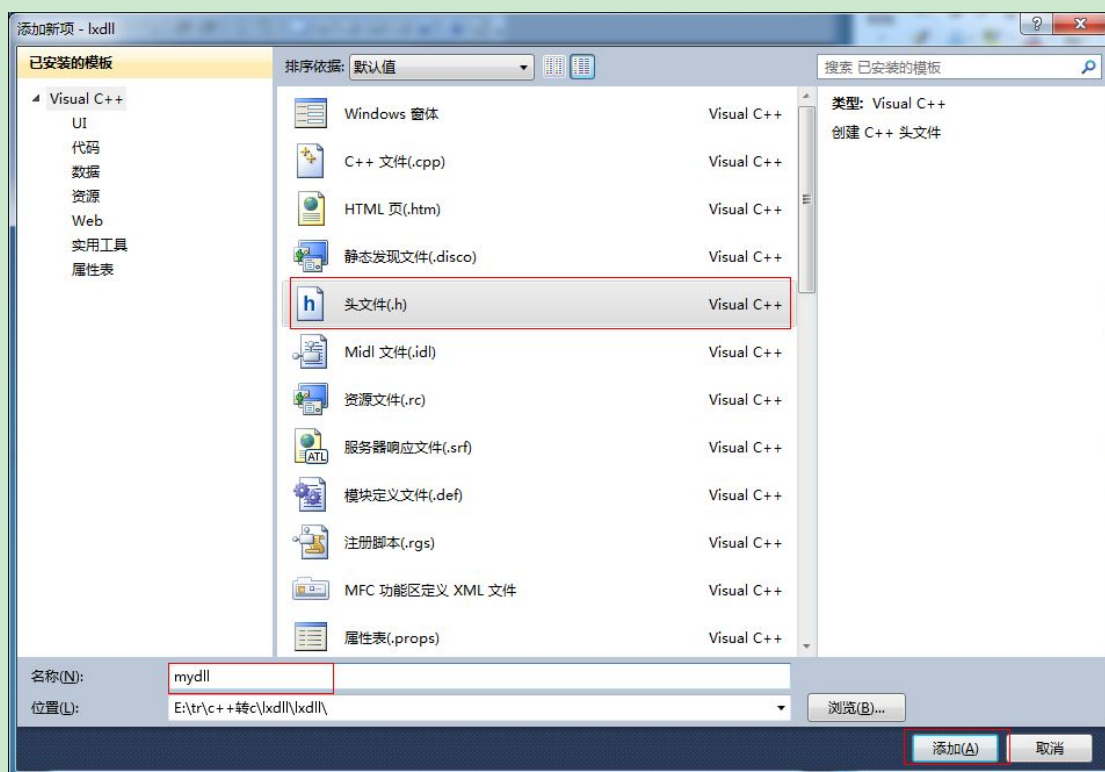
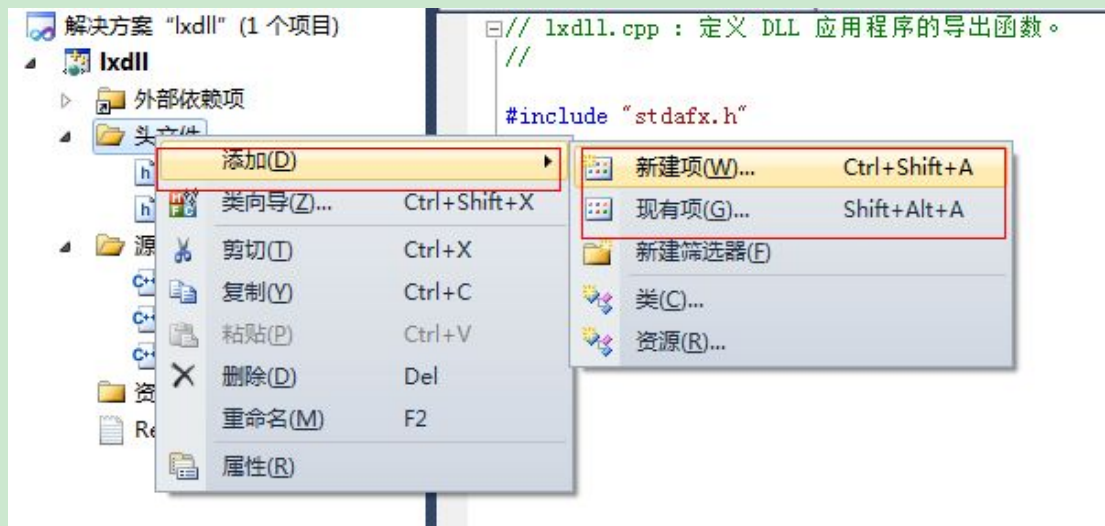
接下来你的项目会有如下目录和文件



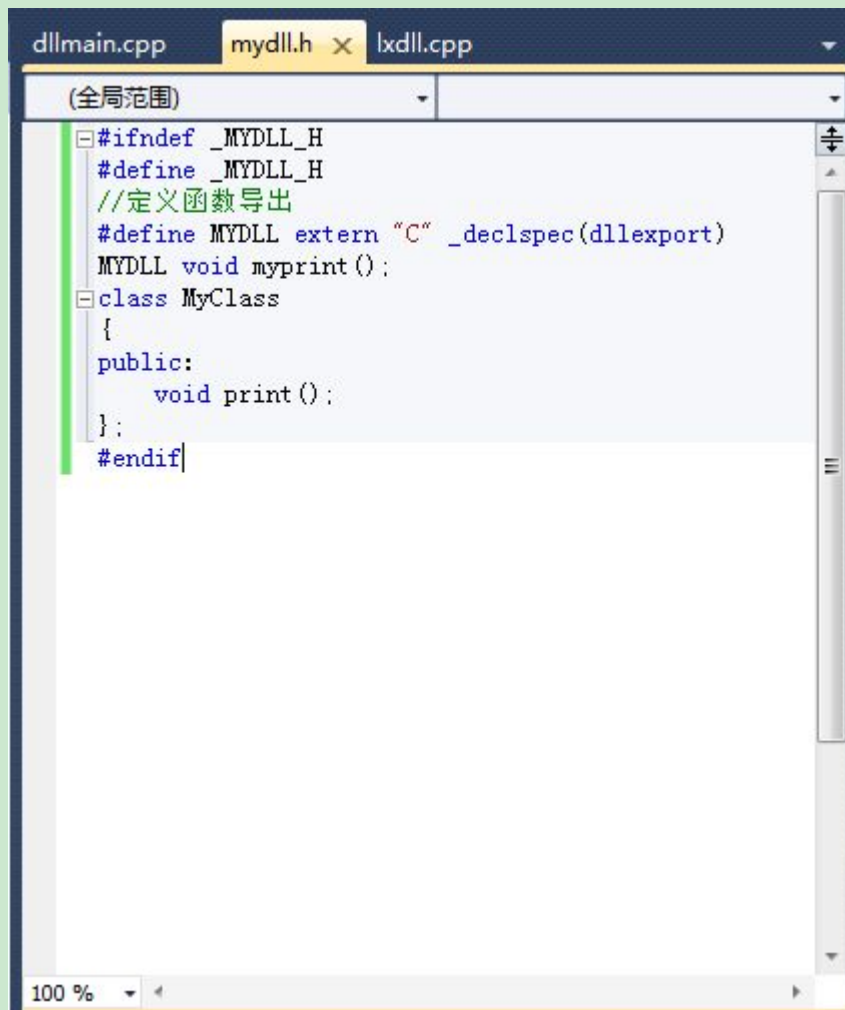
二、添加头文件

在头文件中添加你要编写的函数声明操作如下（注：若你已编写完成就点击现有项，没有就点击新建项 我的是新建项目 如下操作

点中头文件右击添加→新建项→选中头文件(.h)→名称中输入你要起的名字如我的 mydll→添加

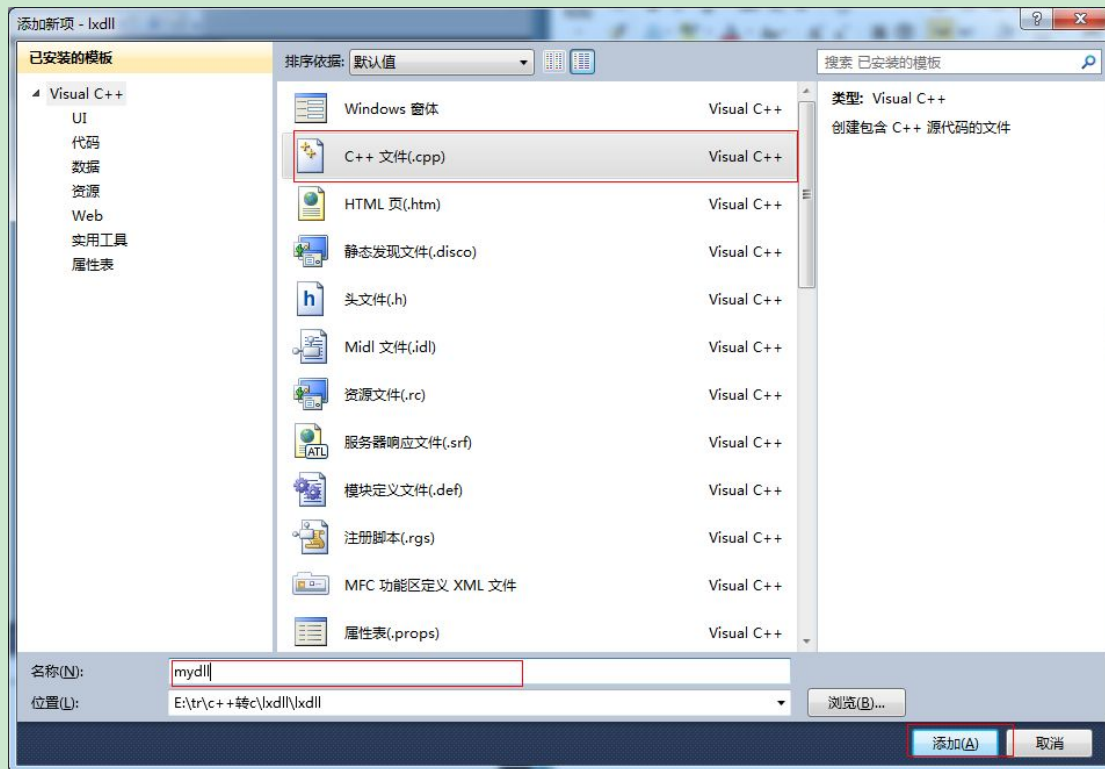


添加完成后编辑你的代码（注：是在你新建的头文件中 如我在我的 mydll.h 编辑如下）

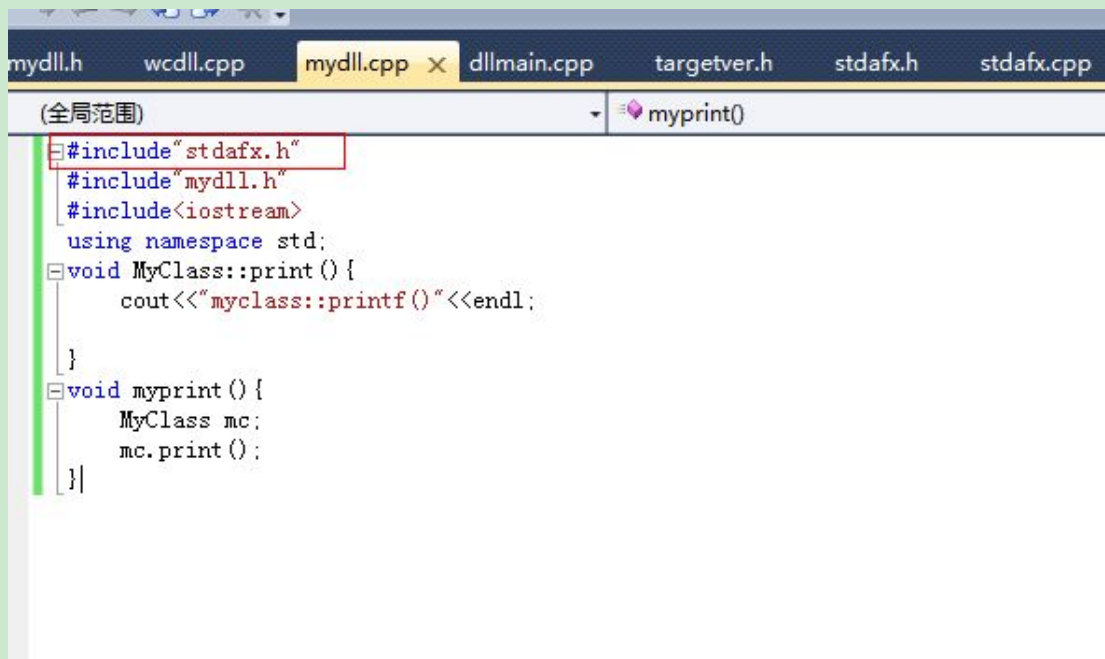


```
dllmain.cpp  mydll.h x  lxdll.cpp
(全局范围)
#ifndef _MYDLL_H
#define _MYDLL_H
//定义函数导出
#define MYDLL extern "C" __declspec(dllexport)
MYDLL void myprint();
class MyClass
{
public:
    void print();
};
#endif
```

好接下来我们对函数进行实现 函数的实现在源文件中添加步骤和添加头文件一样 只是在选择文件类型时选择 C++ 文件(.cpp) 名称中填写你自定义的文件（我的如下）



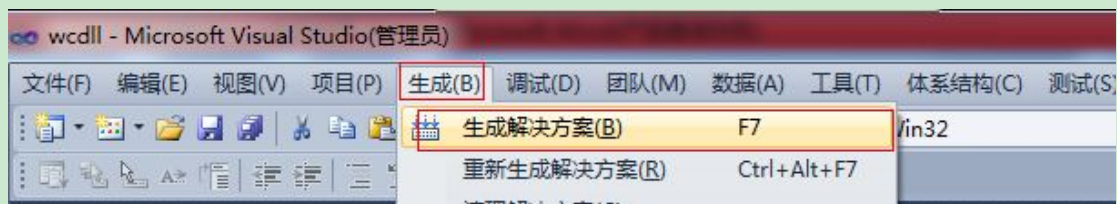
进入编辑面板对我们新添加的文件进行编写（我的代码如下）



注：stdafx.h 头文件不添加太会提示如下错误

```
输出
显示输出来源(S): 生成
1> 将指令添加到“StdAfx.h”或重新生成预编译器
1>e:\tr\c++\转c\wcdll\wcdll\mydll.cpp(2): warning C4627: “#include <iostream>”: 在查找预编译器使用时跳过
1> 将指令添加到“StdAfx.h”或重新生成预编译器
1>e:\tr\c++\转c\wcdll\wcdll\mydll.cpp(12): fatal_error C1010: 在查找预编译器时遇到意外的文件结尾。是否忘记了向源中添加“#include “StdAfx.h””?
1> 正在生成代码...
1>
1>生成失败。
1>
```

好编辑完成 我们就来编译 操作如下点击菜单栏中 生成→生成解决方案（B）



生成成功如下

```
生成成功如下
1> mydll.cpp
1> 正在生成代码...
1>LinkEmbedManifest:
1> wcdll.vcxproj -> E:\tr\c++\转c\wcdll\Debug\wcdll.dll
1>FinalizeBuildStatus:
1> 正在删除文件“Debug\wcdll.unsuccessfulbuild”。
1> 正在对“Debug\wcdll.lastbuildstate”执行 Touch 任务。
1>
1>生成成功。|
1>
1>已用时间 00:00:03.14
===== 生成: 成功 1 个, 失败 0 个, 最新 0 个, 跳过 0 个 =====
```

好大功告成接下来我们测试下面的是显示调用

三、显示调用 DLL

第一先建立一个工程

步骤如上我建立的工程名为 Dlltest 但注意如下几点 应用程序类型

选择 控制台应用程序

附加选项 选择 空项目 →完成

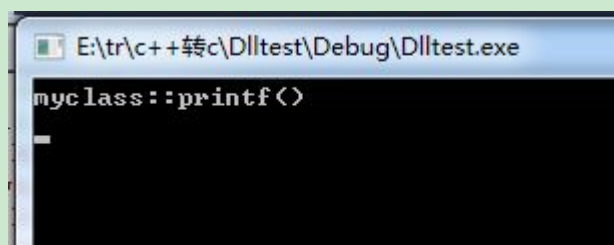


然后右击源文件添加你要编写的.c 文件（注：显示调用 c++和 c 调用动态库写法一样）我添加的是 dlltest.c 文件

接下来我面编写代码 我的代码如下

```
1 // #include "testclass.h"
2 // #include "mydll.h"
3 #include <stdio.h>
4 #include <Windows.h>
5 #include <conio.h>
6 typedef void (*print)();
7 int main(void) {
8     HINSTANCE hDll;
9     hDll = LoadLibraryA("../Dlltest/Debug/lxdll.dll"); // 加载动态库
10    if (hDll != NULL)
11    {
12
13        print printfun;
14
15        printf = (print)GetProcAddress(hDll, "myprint"); // 获取动态库中函数名为myprint的函数指针
16        if (printf != NULL)
17        {
18            printf(); // 执行
19        }
20        else {
21
22            printf("未找到该函数, 可能函数名错误!");
23        }
24        FreeLibrary(hDll);
25    }
26    else {
27
28        printf("未能打开dll可能给定路径错误!");
29    }
30    // myprint();
31    getch();
32    return 0;
33 }
```

接下来我们运行 运行结果如下 根据自己的 dll 路径位置 设置路径



接下来我们进行隐式调用

四、隐式调用 DLL

第一步我们建立工程 和上面一样 不再叙述 我建立的是 testlxdll

接下来我们添加头文件 我们先添加用 c 程序测试我们编写的 dll

我添加的是名为 head.h 内容如下

```
(全局范围)
#include<stdio.h>
#include<conio.h>
extern __declspec(dllimport)void myprint();
```

其中 `ldll.lib` 是在生成 `dll` 时产生的 `lib` 是什么？ 如下解释：

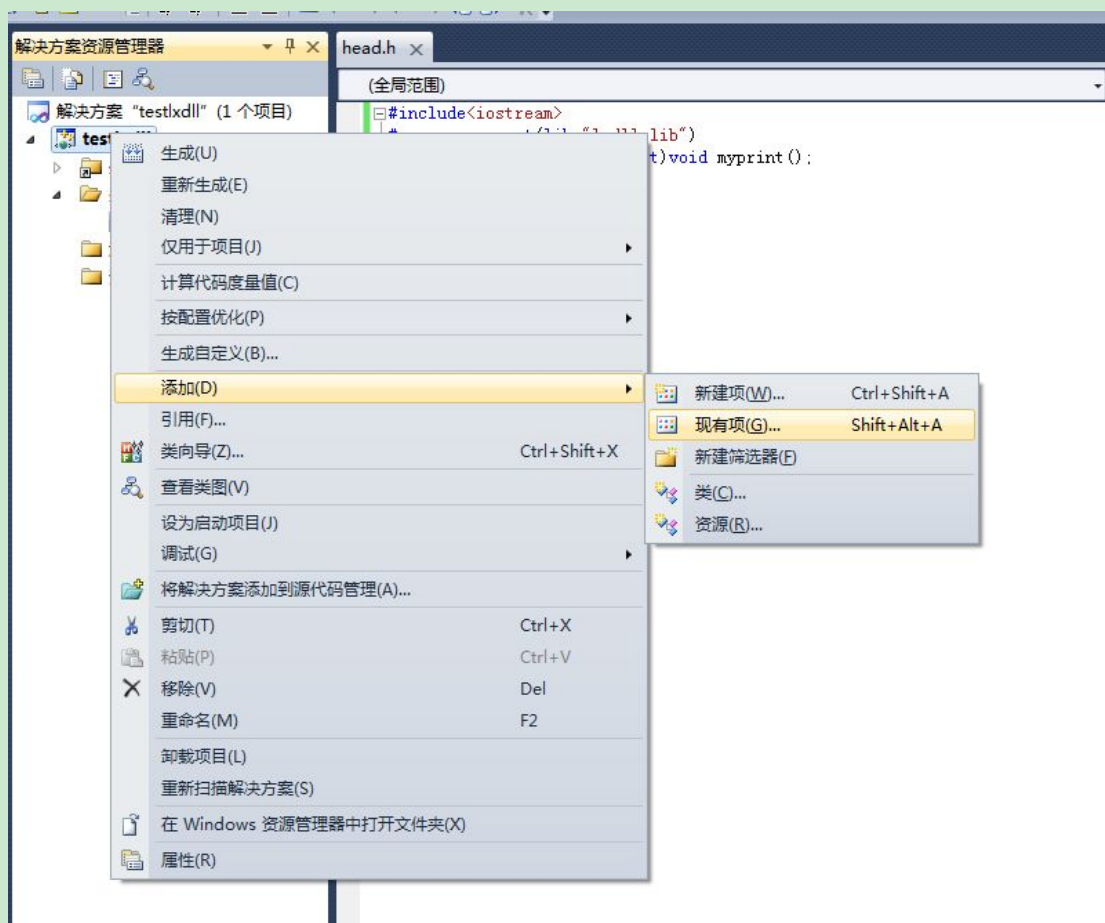
如果有 `dll` 文件，那么 `lib` 一般是一些索引信息，记录了 `dll` 中函数的入口和位置，`dll` 中是函数的具体内容；如果只有 `lib` 文件，那么这个 `lib` 文件是静态编译出来的，索引和实现都在其中。使用静态编译的 `lib` 文件，在运行程序时不需要再挂动态库，缺点是导致应用程序比较大，而且失去了动态库的灵活性，发布新版本时要发布新的应用程序才行。

看来我们的 `lib` 就是记录记录了 `dll` 中函数的入口和位置

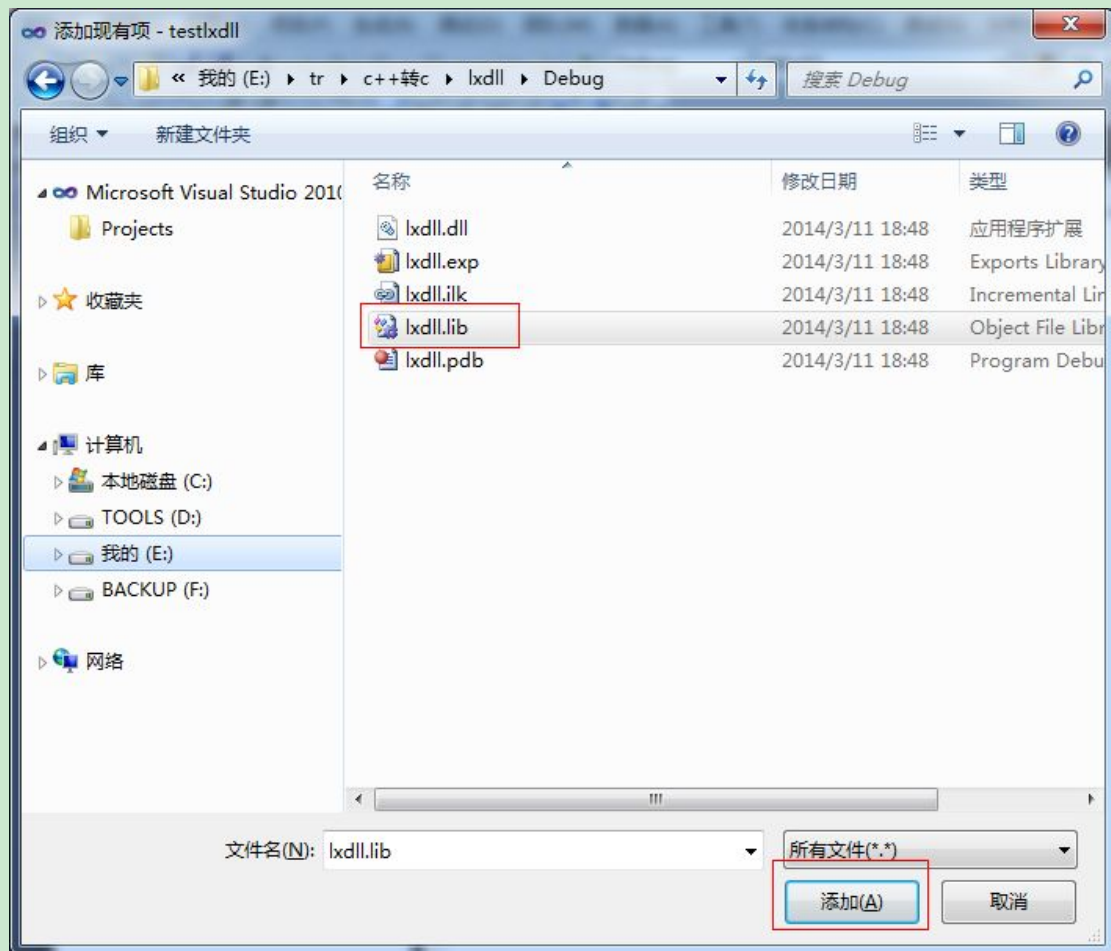
现在我们程序还不能打开 `lib` ， 因为程序中没有，我们要导入它

步骤如下

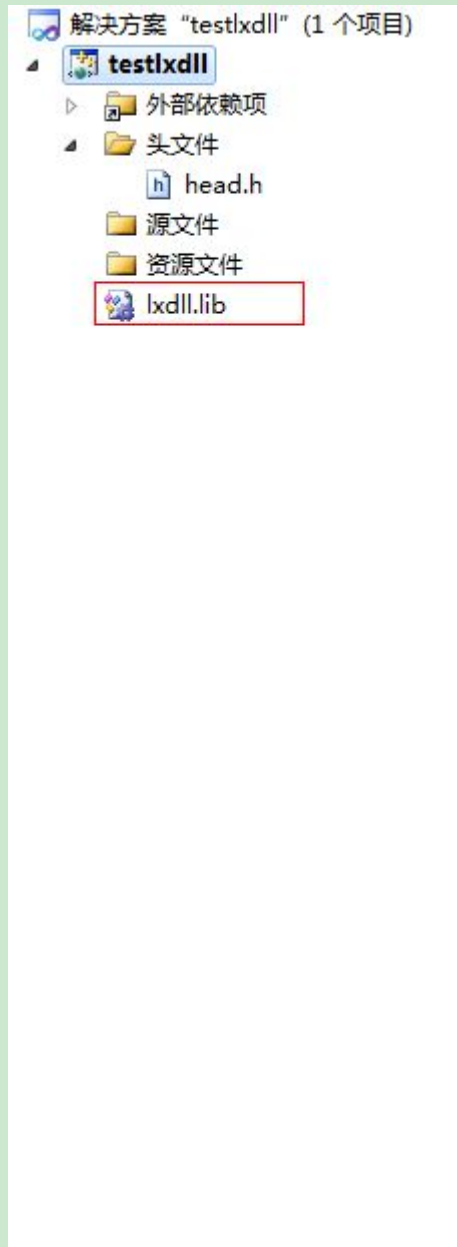
右击项目→添加→现有项



找到你生成的.lib 文件点击添加



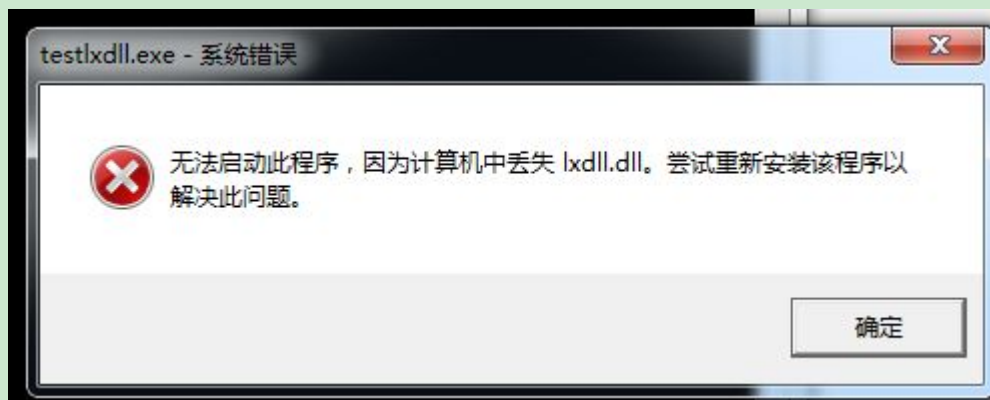
添加成功后我们项目栏中会有 lxdll.lib 文件



接下来我们添加源文件 记住扩展名为.c （我起的为 testlxdll.c）内容如下

```
(主函数范围)
#include "head.h"
int main(int argc, char* argv[])
{
    myprint();
    getch();
    return 0;
}
```

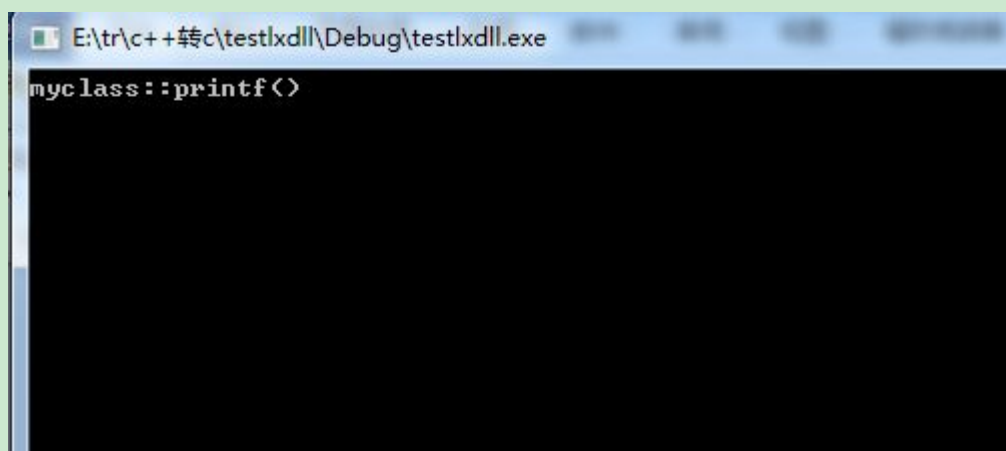
好我们开始运行 他会提示



很明显 程序没有 lxdll.dll 文件 我们要手动添加 找到你项目的位置
有个 debug 目录 记住是有 exe 文件的目录 如我的

名称	修改日期	类型	大小
lxdll.dll	2014/3/11 18:48	应用程序扩展	36 KB
testlxdll.exe	2014/3/12 9:57	应用程序	28 KB
testlxdll.ilc	2014/3/12 9:57	Incremental Link...	296 KB
testlxdll.pdb	2014/3/12 9:57	Program Debug...	347 KB

添加完成 我们在点击运行 就成功了 结果如下

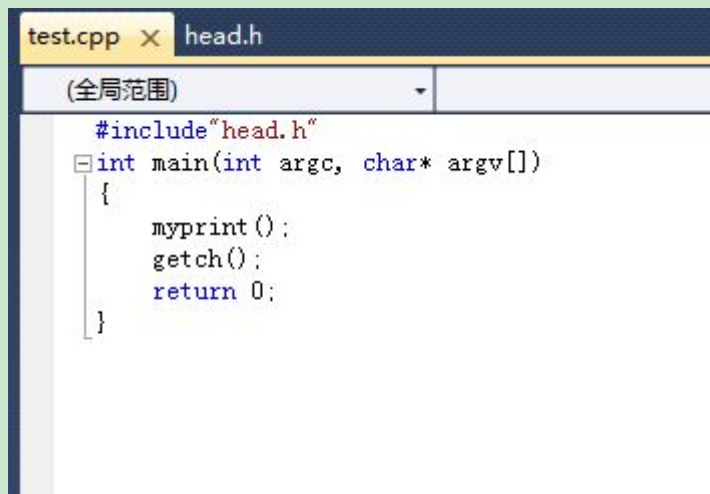


完成 c 调用 dll 我们来学习 c++调用 dll

五、C++调用 DLL

我在当前工程下更改的 将以前的源文件删除 添加新的源文件 注意
我们添加的源文件后缀是.cpp 我添加的是 test.cpp 内容没有改变

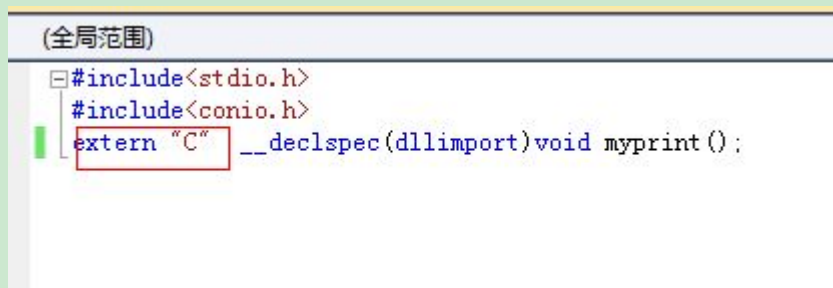
就和.c 内容一样如下



```
test.cpp x head.h
(全局范围)
#include "head.h"
int main(int argc, char* argv[])
{
    myprint();
    getch();
    return 0;
}
```

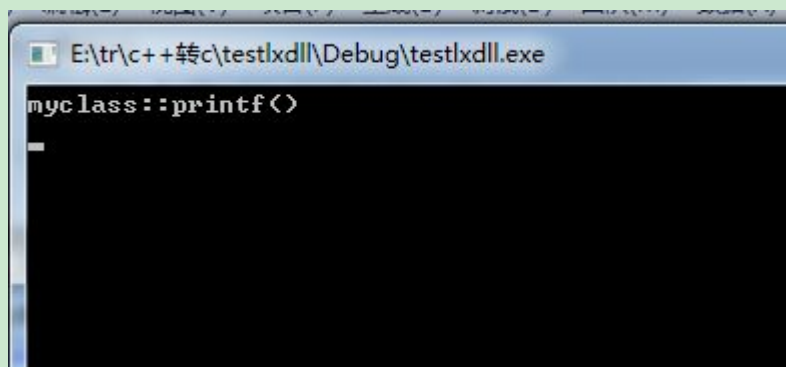
头文件我们还要更改一下 防止 c++编译器给函数偷换名称

只改动一个地方



```
(全局范围)
#include <stdio.h>
#include <conio.h>
extern "C" __declspec(dllimport) void myprint();
```

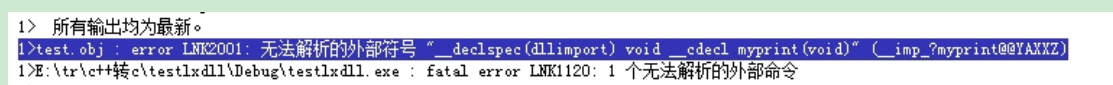
我们测试下



```
E:\tr\c++\转c\testlxdll\Debug\testlxdll.exe
myclass::printf<>
```

OK 测试通过

如我们不改，会出现一下问题



```
1> 所有输出均为最新。
1>test.obj : error LNK2001: 无法解析的外部符号 "__declspec(dllimport) void __cdecl myprint(void)" (imp_?myprint@YAXXZ)
1>E:\tr\c++\转c\testlxdll\Debug\testlxdll.exe : fatal error LNK1120: 1 个无法解析的外部命令
1>
```