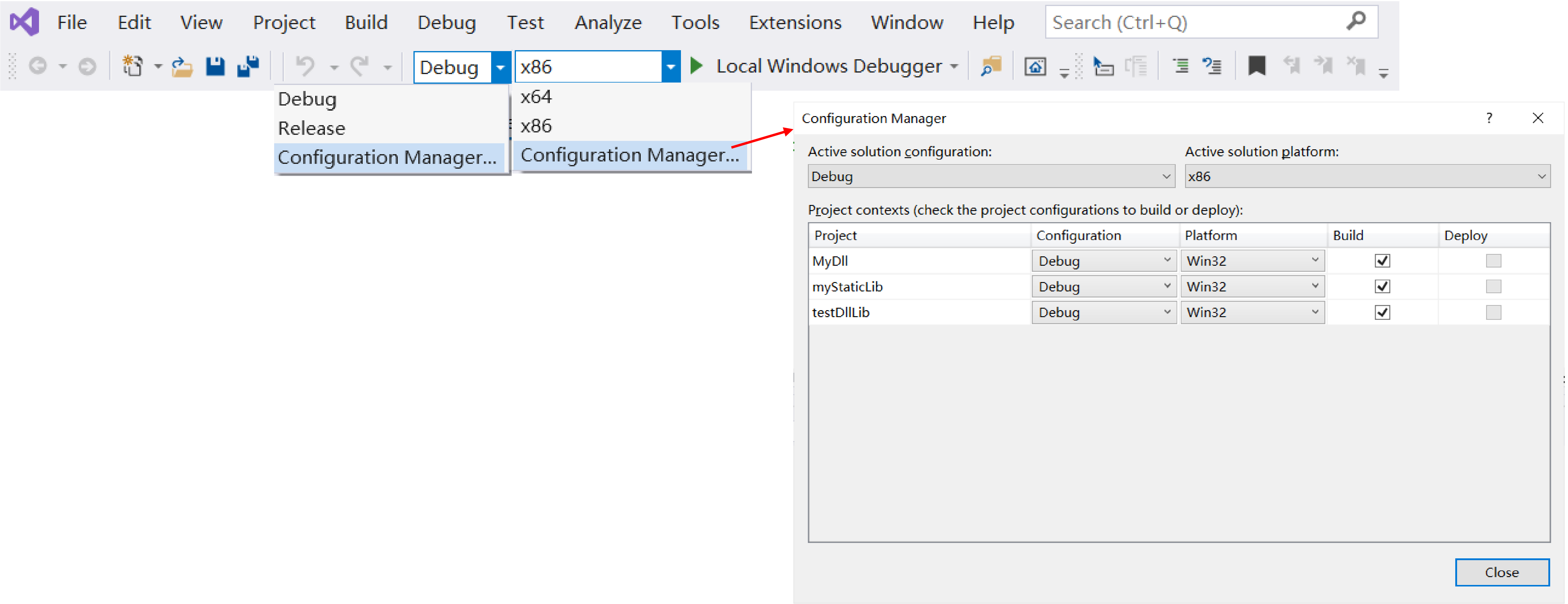
Static Library
Dynamic Library

1. Debug version and Release version

Debug version program contains a lot of debugging information, the final execution file is large, poor performance.
Release version program is usually optimized to make the program the best in terms of code size and running speed.



Active solution configuration: Debug

Active solution platform: x86

Project contexts (check the project configurations to build or deploy):

Project	Configuration	Platform	Build	Deploy
MyDll	Debug	Win32	☑	□
myStaticLib	Debug	Win32	☑	□
testDllLib	Debug	Win32	☑	□

2. Static library and Dynamic library

Static library: In the link phase, the library will be packaged with the object file.o compiled by the target to produce an executable file. Becomes part of the executable, and then the library can disappear. This means that at the last step of compilation (the linking phase), if the program needs to use a static library, it will all be packaged into an executable at this stage.

Dynamic library: The dynamic library does not have any action in the link stage, only when the program is running will be loaded, that is, the link of the dynamic library occurs in the program run time, it is separated from the executable file, only the executable file in a certain period of time to call it.

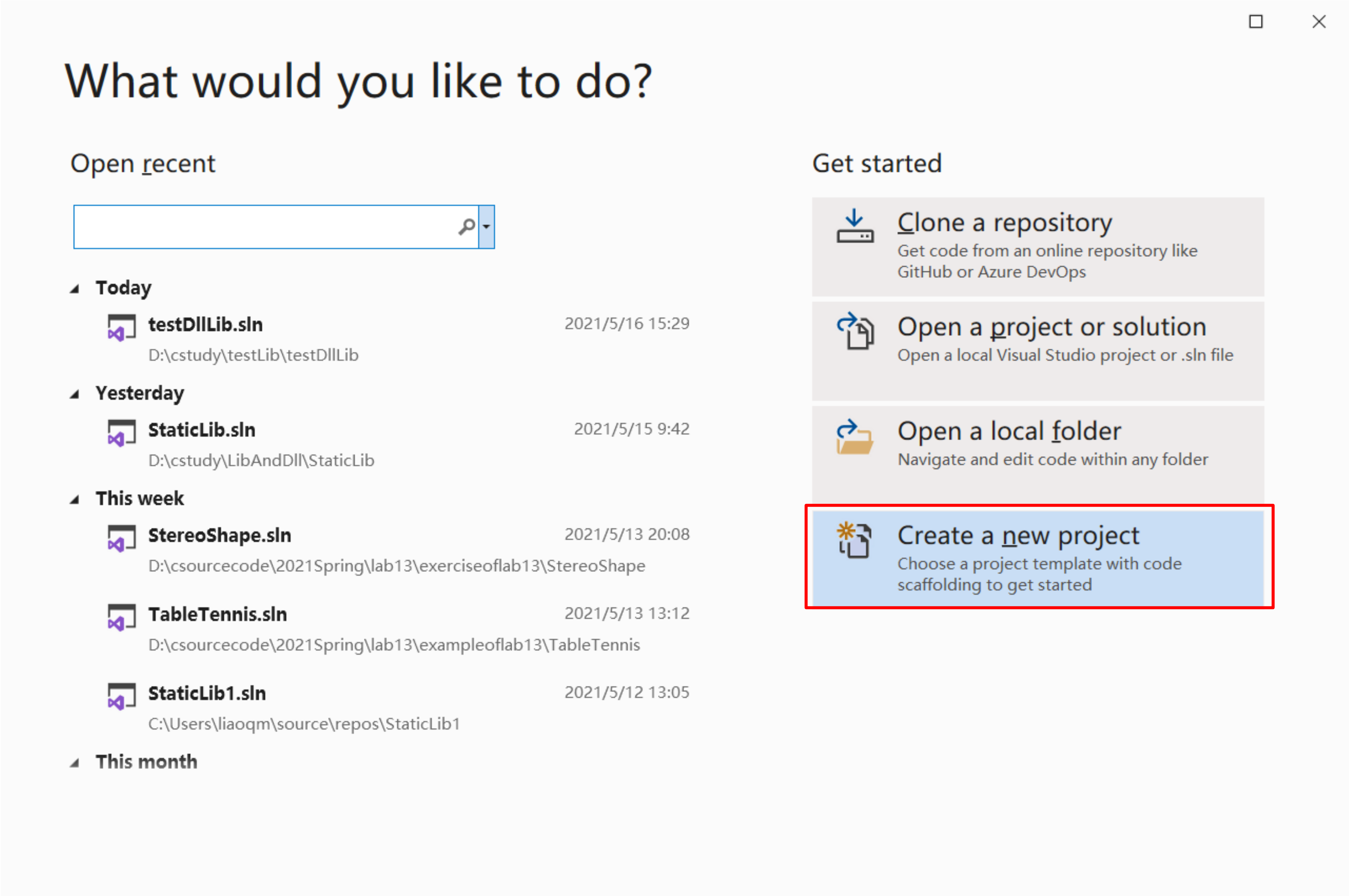
Advantages and disadvantages of Static library and Dynamic library

	advantages	disadvantages
Static Library	1. Make the executable has fewer dependencies, has been packaged into the executable file. 2. The link is completed in the compilation stage, and the code is loaded quickly during execution.	1. Make the executable file larger. 2. Being a library dependent on another library will result in redundant copies because it must be packaged with the target file. 3. Upgrade is not convenient. Upgrade must be recompiled.
Dynamic Library	1.Dynamic library can achieve resource sharing between processes, there can be only one library file. 2. The upgrade procedure is simple, do not need to recompile.	1. Loading during runtime will slow down the execution speed of code. 2. Add program dependencies that must be accompanied by an executable file.

	Windows	Linux
Static Library	.lib	libxxx.a
Dynamic Library	.lib and .dll	libxxx.so

3. Create Static Library

(1) Create a new project in VS 2019



What would you like to do?

Open recent

- Today
  - testDllLib.sln 2021/5/16 15:29
    D:\cstudy\testLib\testDllLib
- Yesterday
  - StaticLib.sln 2021/5/15 9:42
    D:\cstudy\LibAndDll\StaticLib
- This week
  - StereoShape.sln 2021/5/13 20:08
    D:\csourcecode\2021Spring\lab13\exerciseoflab13\StereoShape
  - TableTennis.sln 2021/5/13 13:12
    D:\csourcecode\2021Spring\lab13\exampleoflab13\TableTennis
  - StaticLib1.sln 2021/5/12 13:05
    C:\Users\liaoqm\source\repos\StaticLib1
- This month

Get started

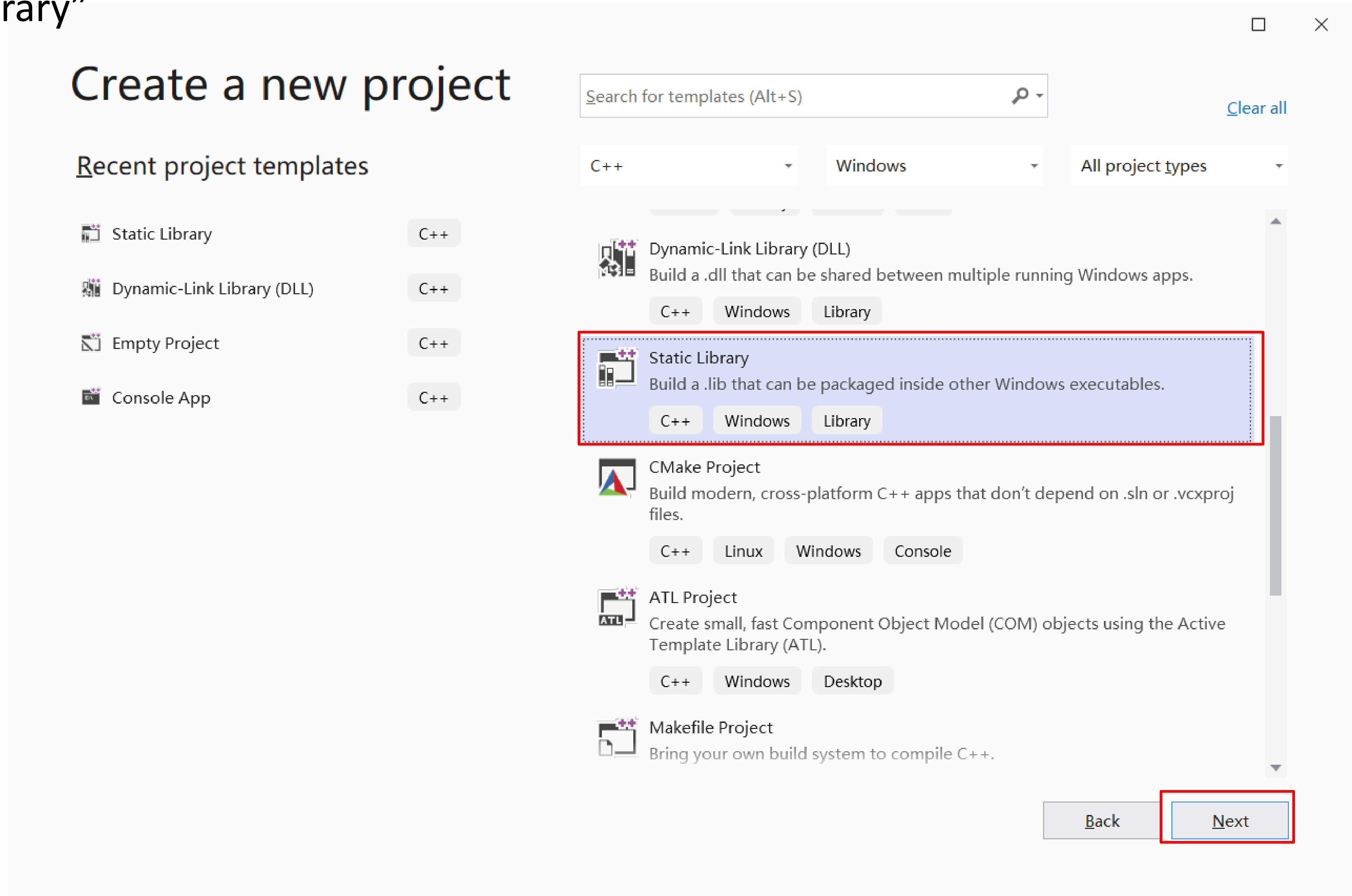
Clone a repository
Get code from an online repository like GitHub or Azure DevOps

Open a project or solution
Open a local Visual Studio project or .sln file

Open a local folder
Navigate and edit code within any folder

Create a new project
Choose a project template with code scaffolding to get started

(2) Select “Static Library”



Create a new project

Search for templates (Alt+S)

Clear all

C++ | Windows | All project types

Recent project templates

Static Library	C++
Dynamic-Link Library (DLL)	C++
Empty Project	C++
Console App	C++

Dynamic-Link Library (DLL)
Build a .dll that can be shared between multiple running Windows apps.
C++ Windows Library

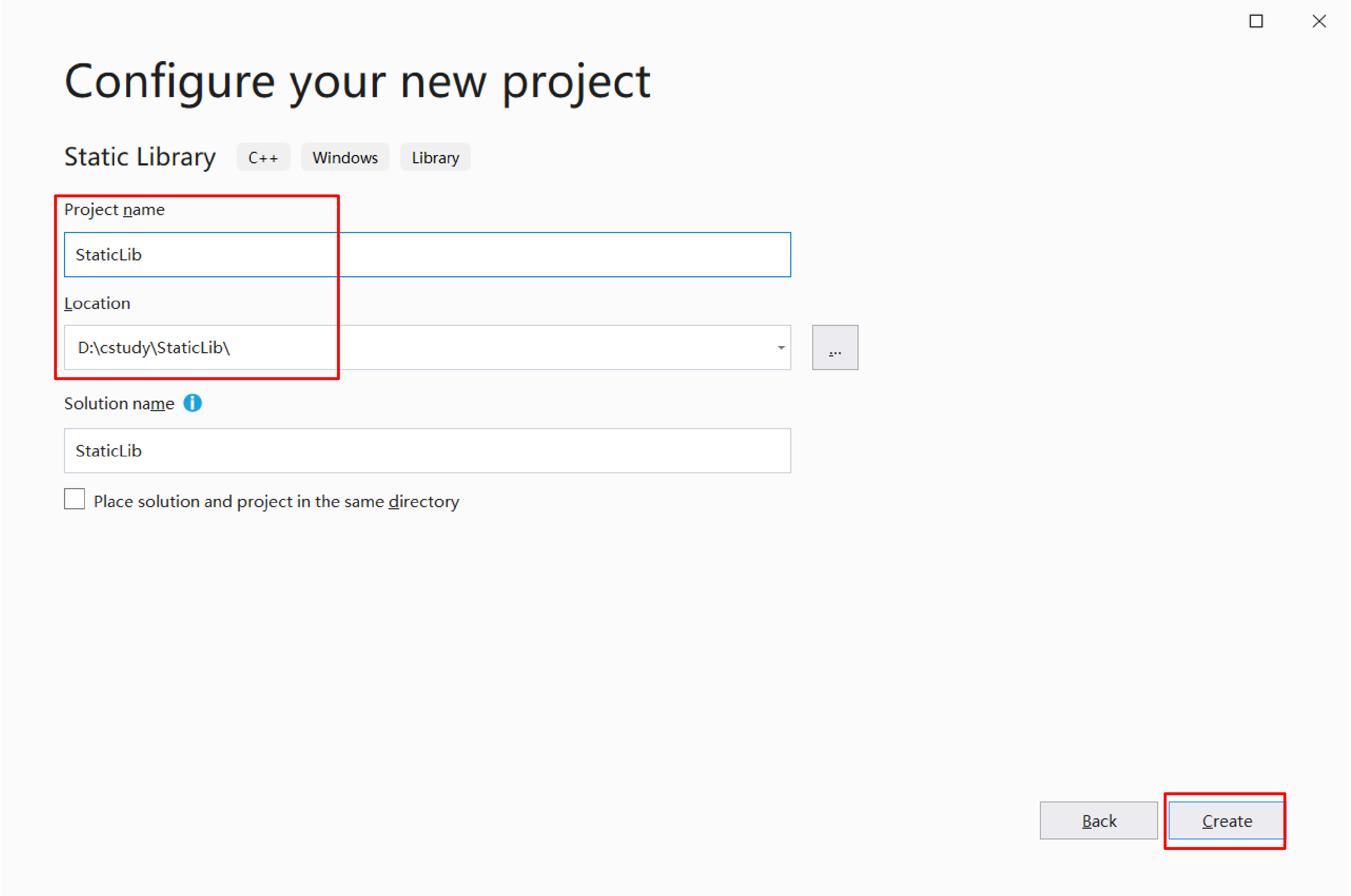
Static Library
Build a .lib that can be packaged inside other Windows executables.
C++ Windows Library

CMake Project
Build modern, cross-platform C++ apps that don’t depend on .sln or .vcxproj files.
C++ Linux Windows Console

ATL Project
Create small, fast Component Object Model (COM) objects using the Active Template Library (ATL).
C++ Windows Desktop

Makefile Project
Bring your own build system to compile C++.

(3) Select the project location and input your project name



Configure your new project

Static Library C++ Windows Library

Project name

StaticLib

Location

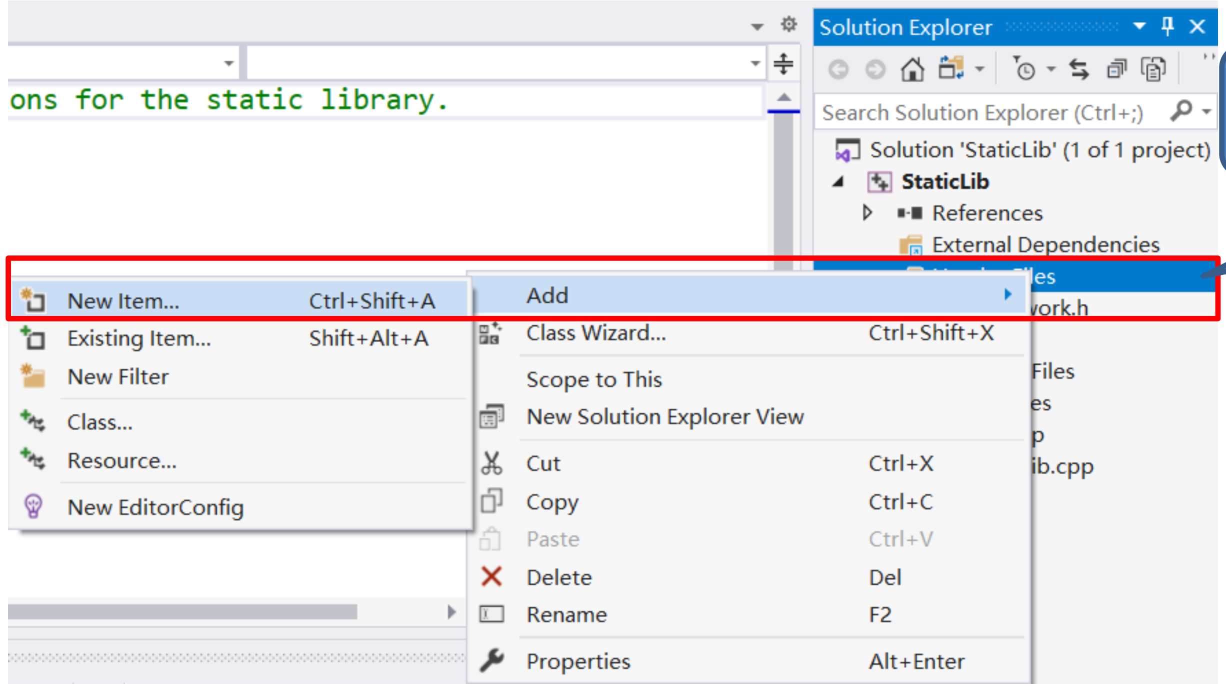
D:\cstudy\StaticLib\

Solution name

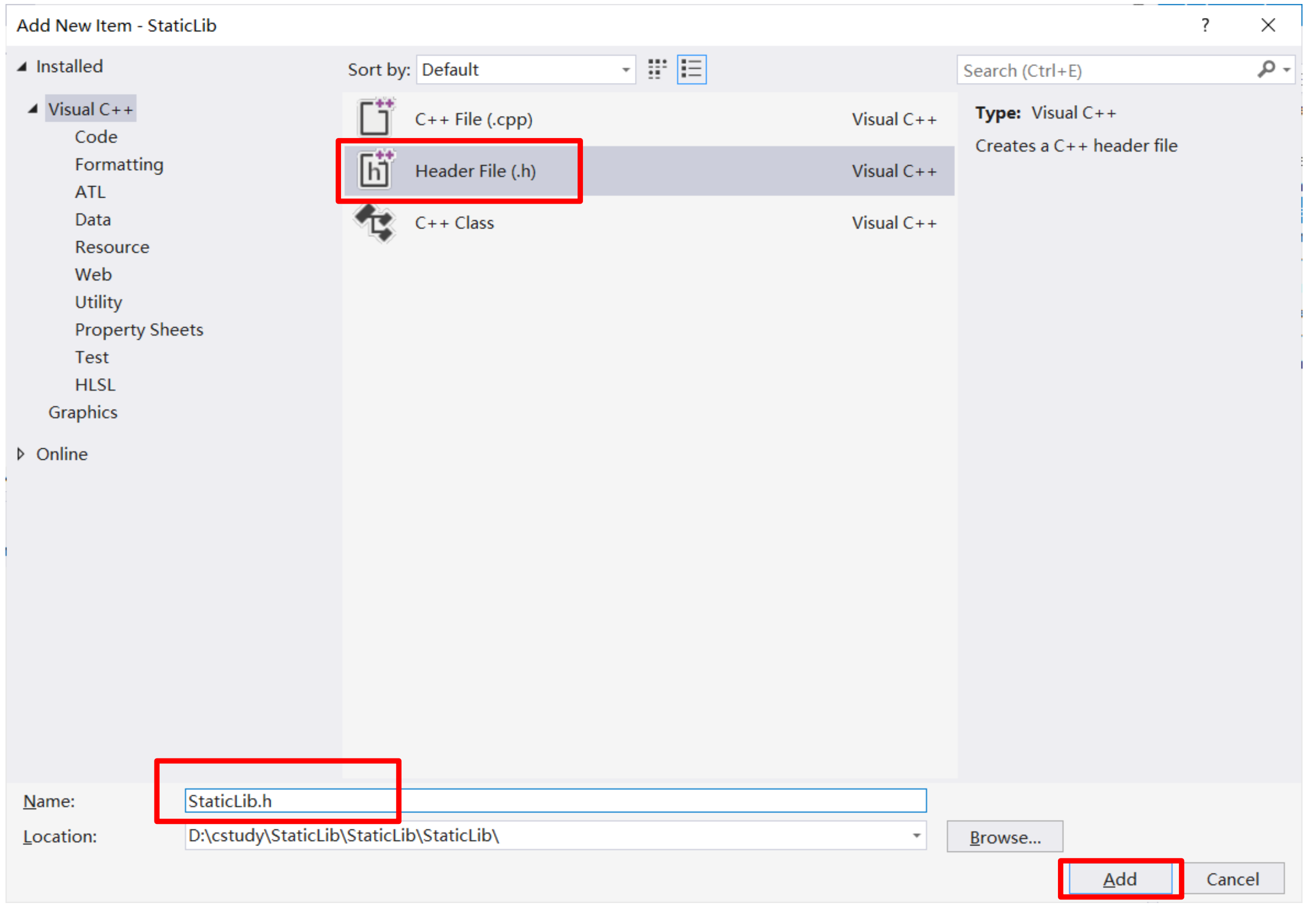
StaticLib

□ Place solution and project in the same directory

(4) Create a header file, declare and implement functions in the static library

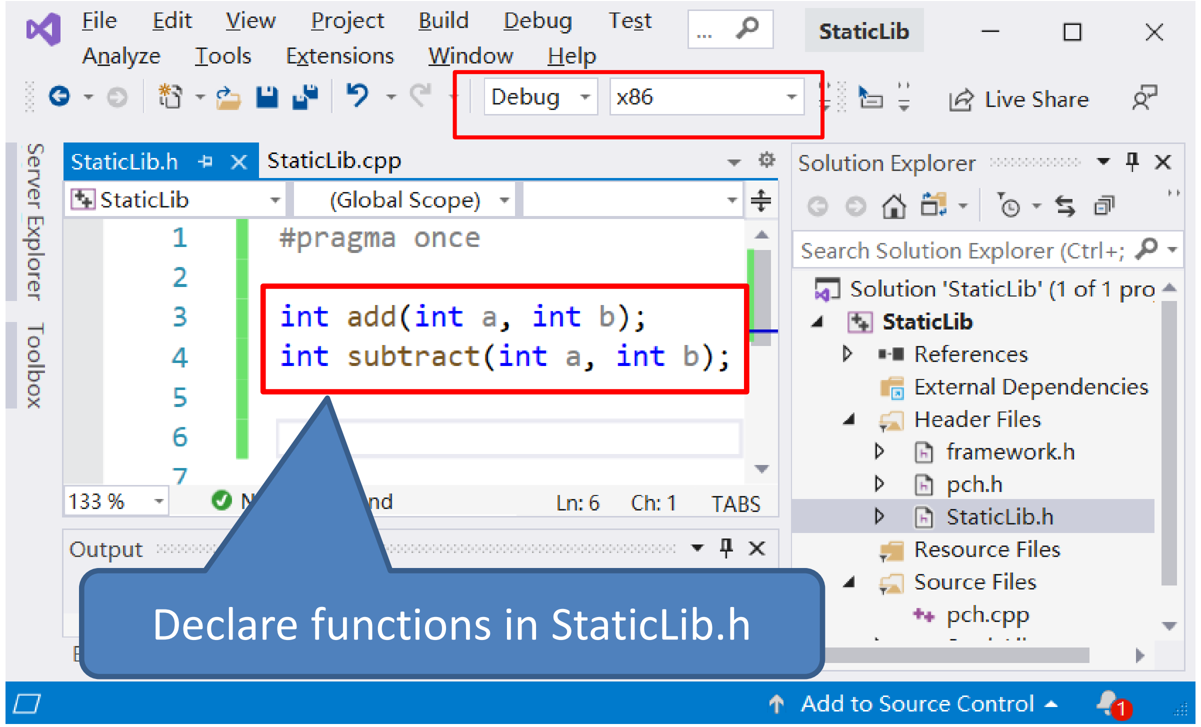


Set the cursor at “Header Files”, and right click the mouse button, select “Add”→”New Item…”



Name: StaticLib.h

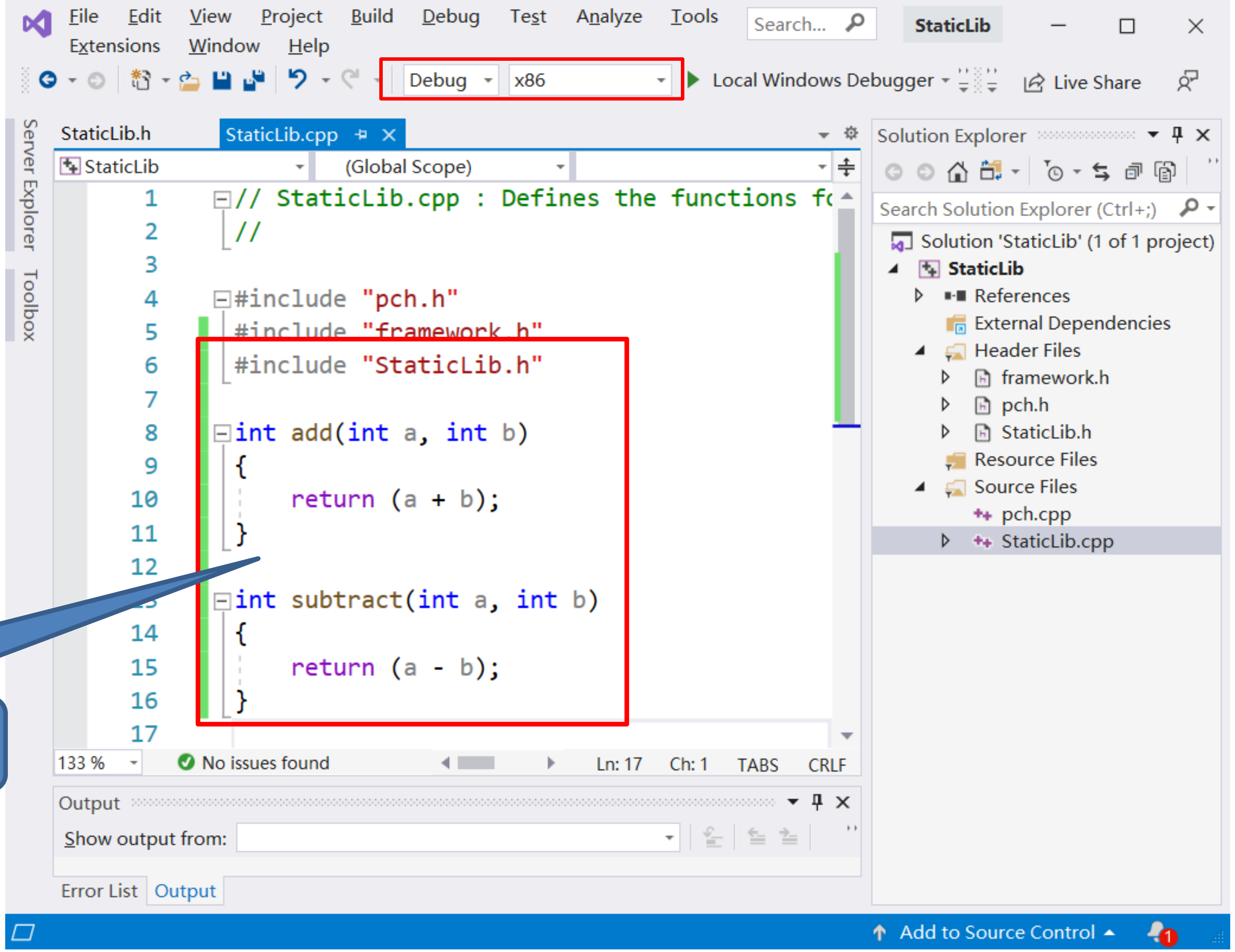
Location: D:\cstudy\StaticLib\StaticLib\StaticLib\



Debug x86

```
#pragma once

int add(int a, int b);
int subtract(int a, int b);
```

Declare functions in StaticLib.h

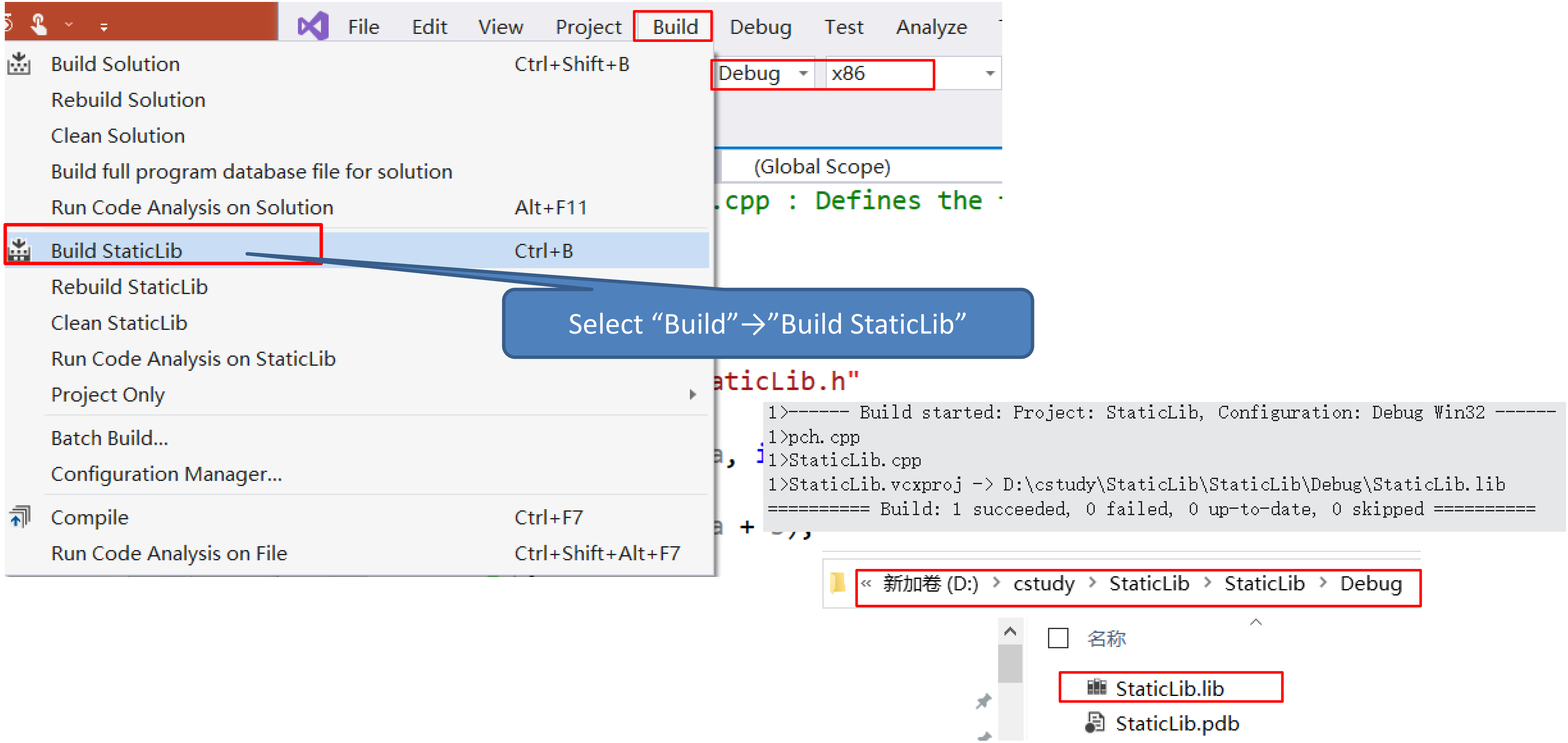


Debug x86

```
// StaticLib.cpp : Defines the functions fo
//

#include "pch.h"
#include "framework.h"
#include "StaticLib.h"

int add(int a, int b)
{
    return (a + b);
}

int subtract(int a, int b)
{
    return (a - b);
}
```

Implement functions in StaticLib.cpp

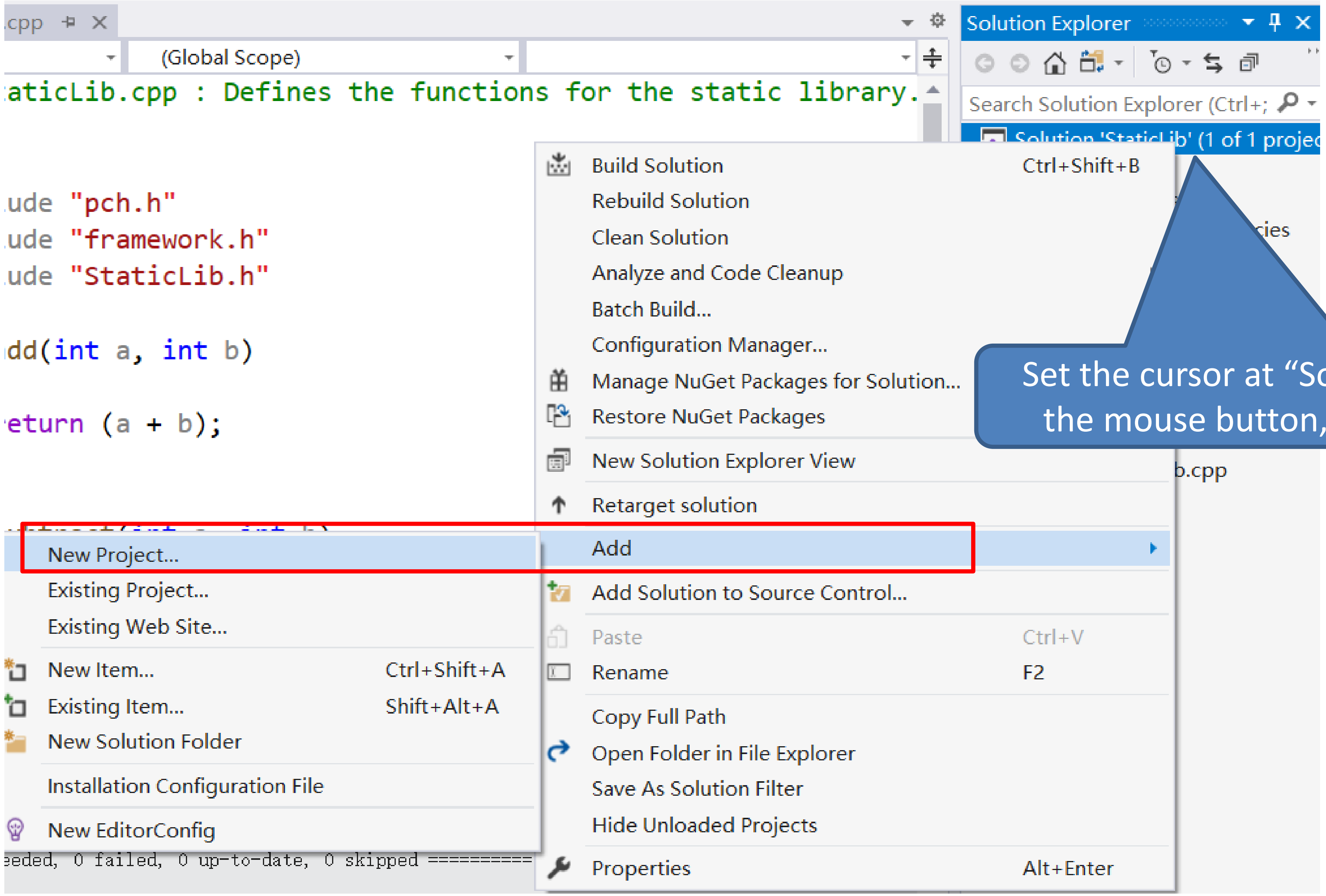
(5) Build the static library



Select “Build”→”Build StaticLib”

```
1>------ Build started: Project: StaticLib, Configuration: Debug Win32 ------
1>pch.cpp
1>StaticLib.cpp
1>StaticLib.vcxproj -> D:\cstudy\StaticLib\StaticLib\Debug\StaticLib.lib
========== Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped ==========
```

(6) Create a new project to test the static library

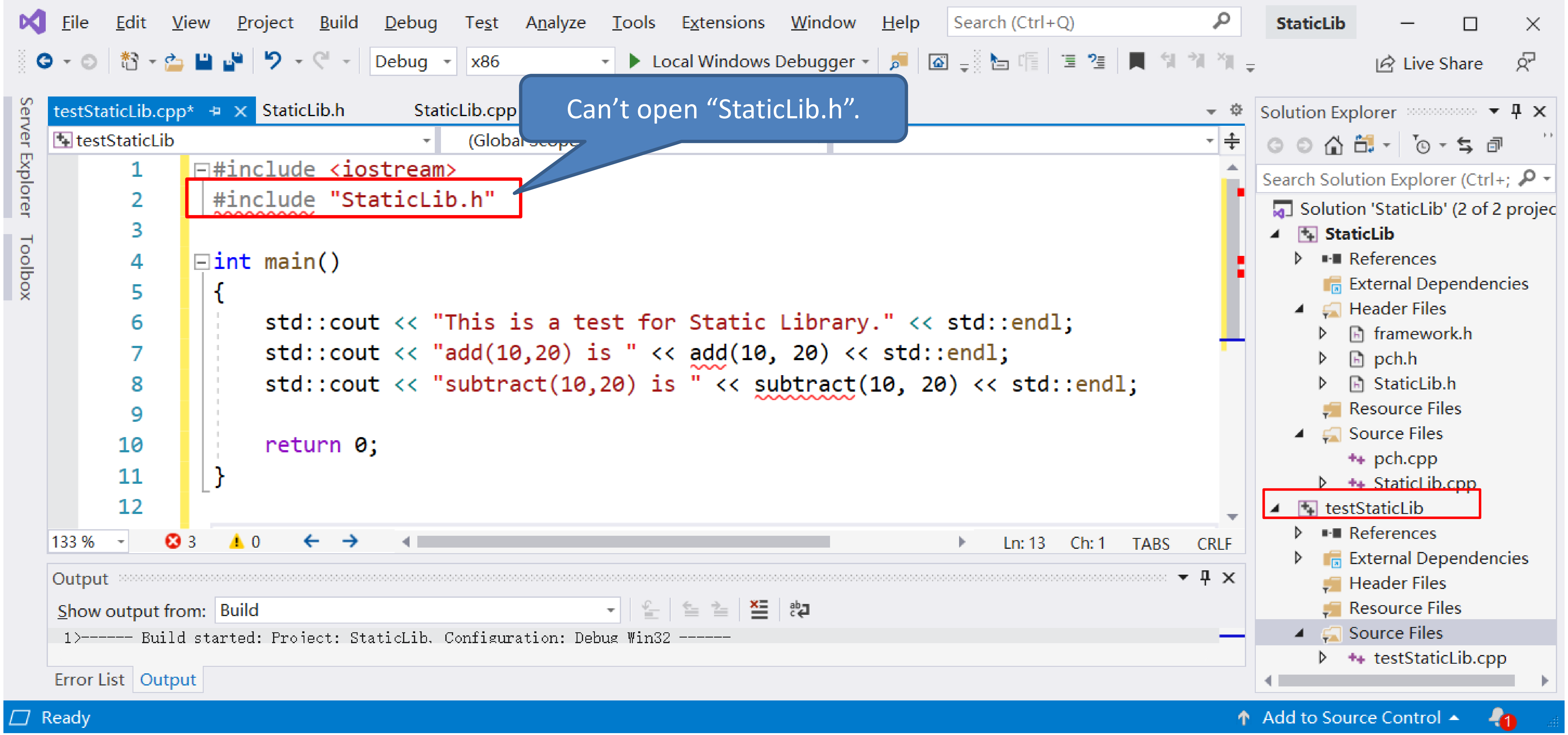


```
aticLib.cpp : Defines the functions for the static library.

ude "pch.h"
ude "framework.h"
ude "StaticLib.h"

dd(int a, int b)

eturn (a + b);
```

```
eeded, 0 failed, 0 up-to-date, 0 skipped ==========
```

Set the cursor at “Solution ‘StaticLib’”, and right click the mouse button, select “Add”→”New Project…”

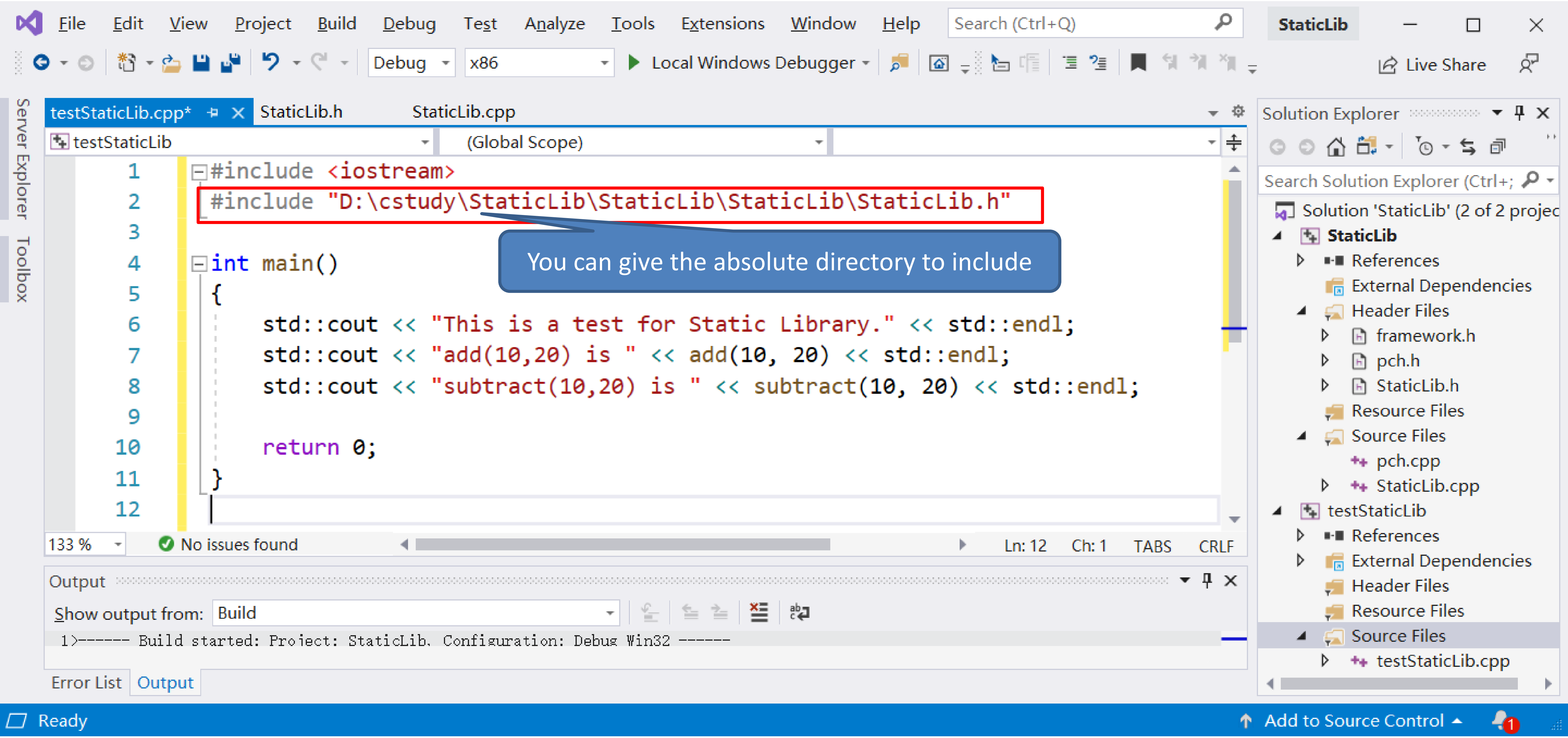
(7) Input code to invoke the static library functions



Can't open "StaticLib.h".

```
#include <iostream>
#include "StaticLib.h"

int main()
{
    std::cout << "This is a test for Static Library." << std::endl;
    std::cout << "add(10,20) is " << add(10, 20) << std::endl;
    std::cout << "subtract(10,20) is " << subtract(10, 20) << std::endl;

    return 0;
}
```

```
1>------ Build started: Project: StaticLib, Configuration: Debug Win32 ------
```

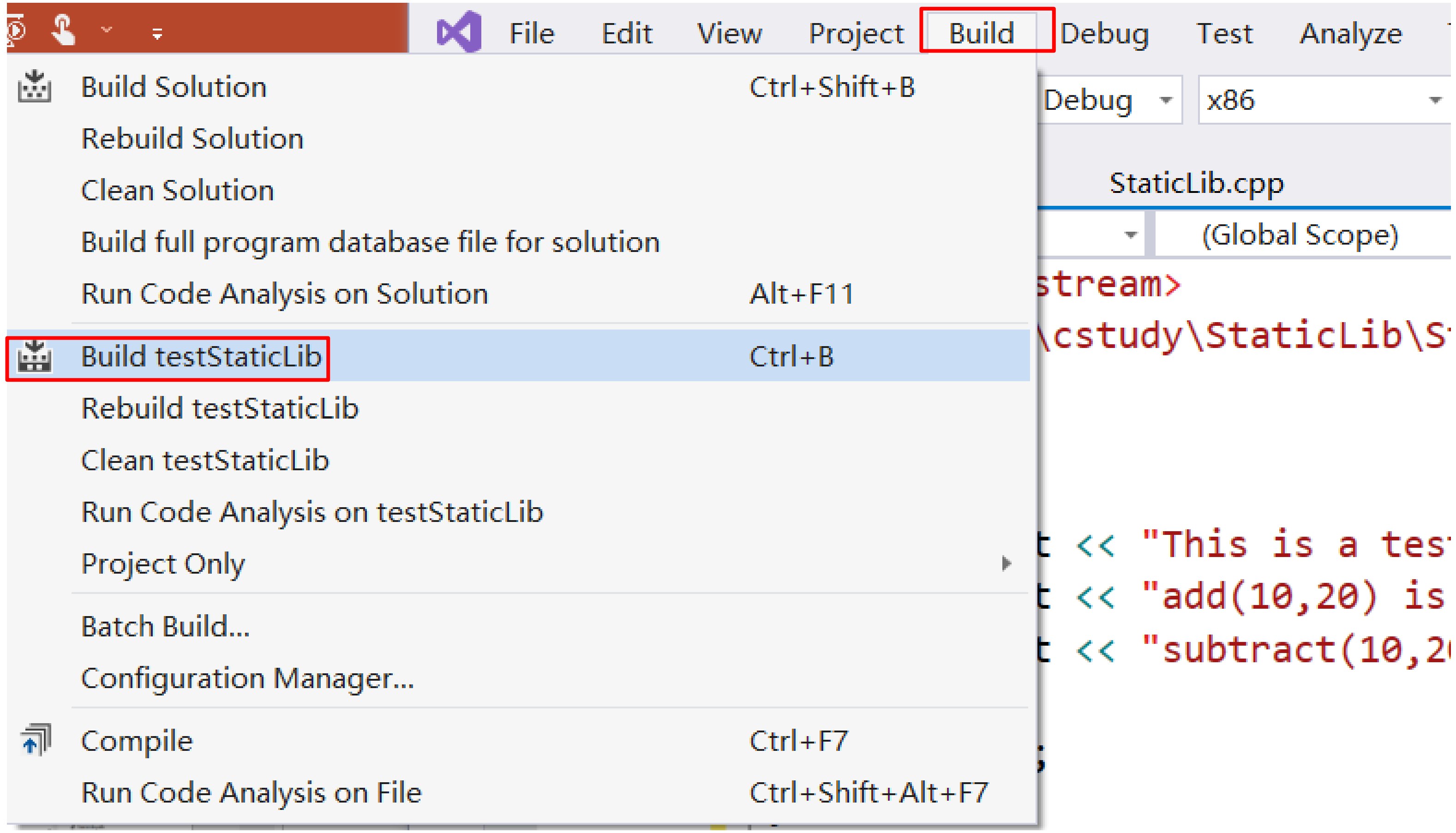


```
#include <iostream>
#include "D:\cstudy\StaticLib\StaticLib\StaticLib\StaticLib.h"

int main()
{
    std::cout << "This is a test for Static Library." << std::endl;
    std::cout << "add(10,20) is " << add(10, 20) << std::endl;
    std::cout << "subtract(10,20) is " << subtract(10, 20) << std::endl;

    return 0;
}
```

You can give the absolute directory to include

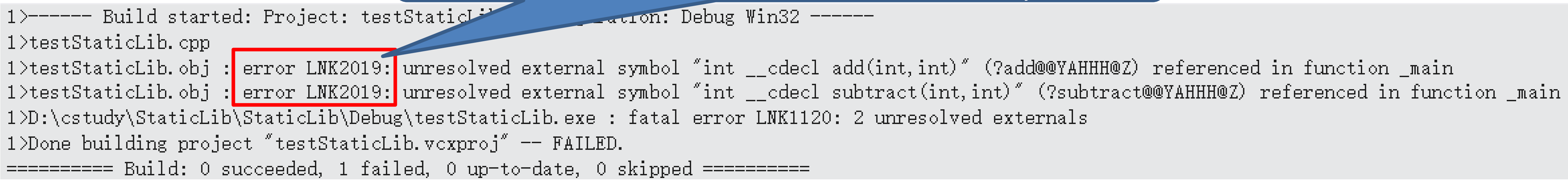
```
1>------ Build started: Project: StaticLib, Configuration: Debug Win32 ------
```

(8) Build the static library


File
Edit
View
Project
Build
Debug
Test
Analyze
Build Solution Ctrl+Shift+B
Rebuild Solution
Clean Solution
Build full program database file for solution
Run Code Analysis on Solution Alt+F11
Build testStaticLib Ctrl+B
Rebuild testStaticLib
Clean testStaticLib
Run Code Analysis on testStaticLib
Project Only
Batch Build...
Configuration Manager...
Compile Ctrl+F7
Run Code Analysis on File Ctrl+Shift+Alt+F7
Debug
x86
StaticLib.cpp
(Global Scope)

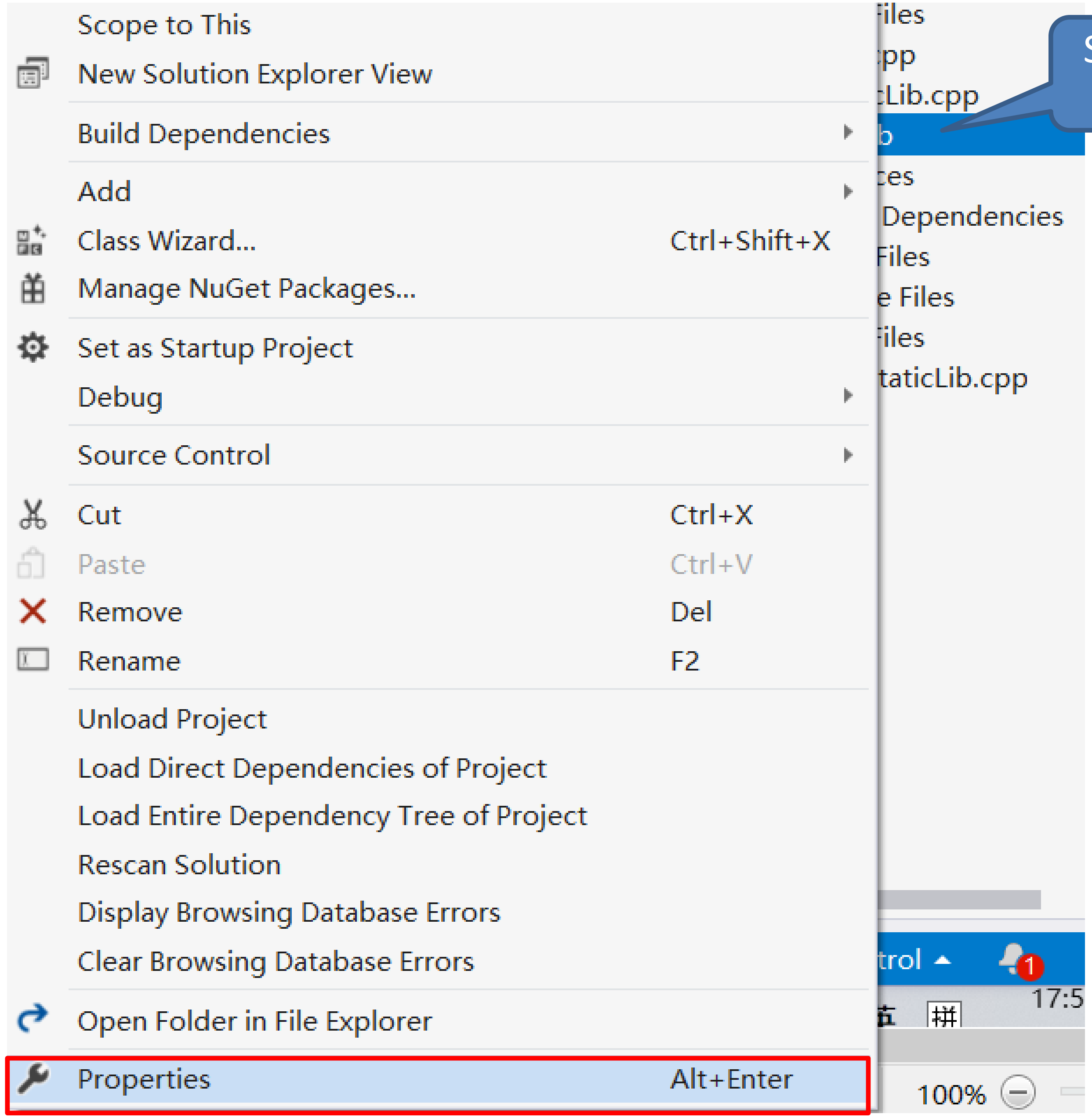
```
stream>
\cstudy\StaticLib\S

t << "This is a tes
t << "add(10,20) is
t << "subtract(10,2

;
```

Build the project, link errors appeared. Just include the header file can not link the static library.

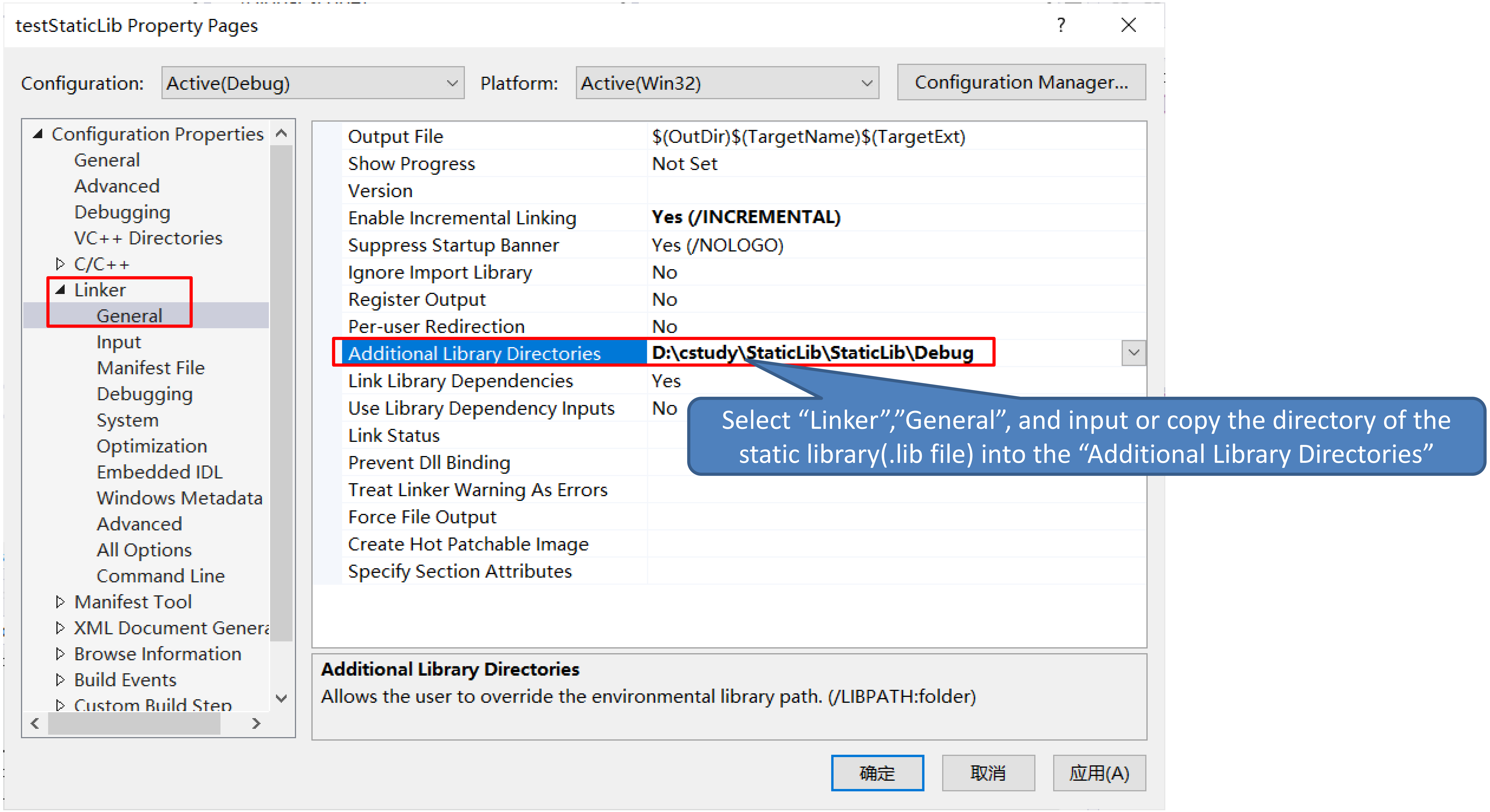


```
1>------ Build started: Project: testStaticLi          ration: Debug Win32 ------
1>testStaticLib.cpp
1>testStaticLib.obj : error LNK2019: unresolved external symbol "int __cdecl add(int,int)" (?add@@YAHHH@Z) referenced in function _main
1>testStaticLib.obj : error LNK2019: unresolved external symbol "int __cdecl subtract(int,int)" (?subtract@@YAHHH@Z) referenced in function _main
1>D:\cstudy\StaticLib\StaticLib\Debug\testStaticLib.exe : fatal error LNK1120: 2 unresolved externals
1>Done building project "testStaticLib.vcxproj" -- FAILED.
========== Build: 0 succeeded, 1 failed, 0 up-to-date, 0 skipped ==========
```

(9) In vs2019, use its linker to load the static library



Set the cursor at “testStaticLib’” project, and right click the mouse button, select “Properties”



Configuration: Active(Debug)

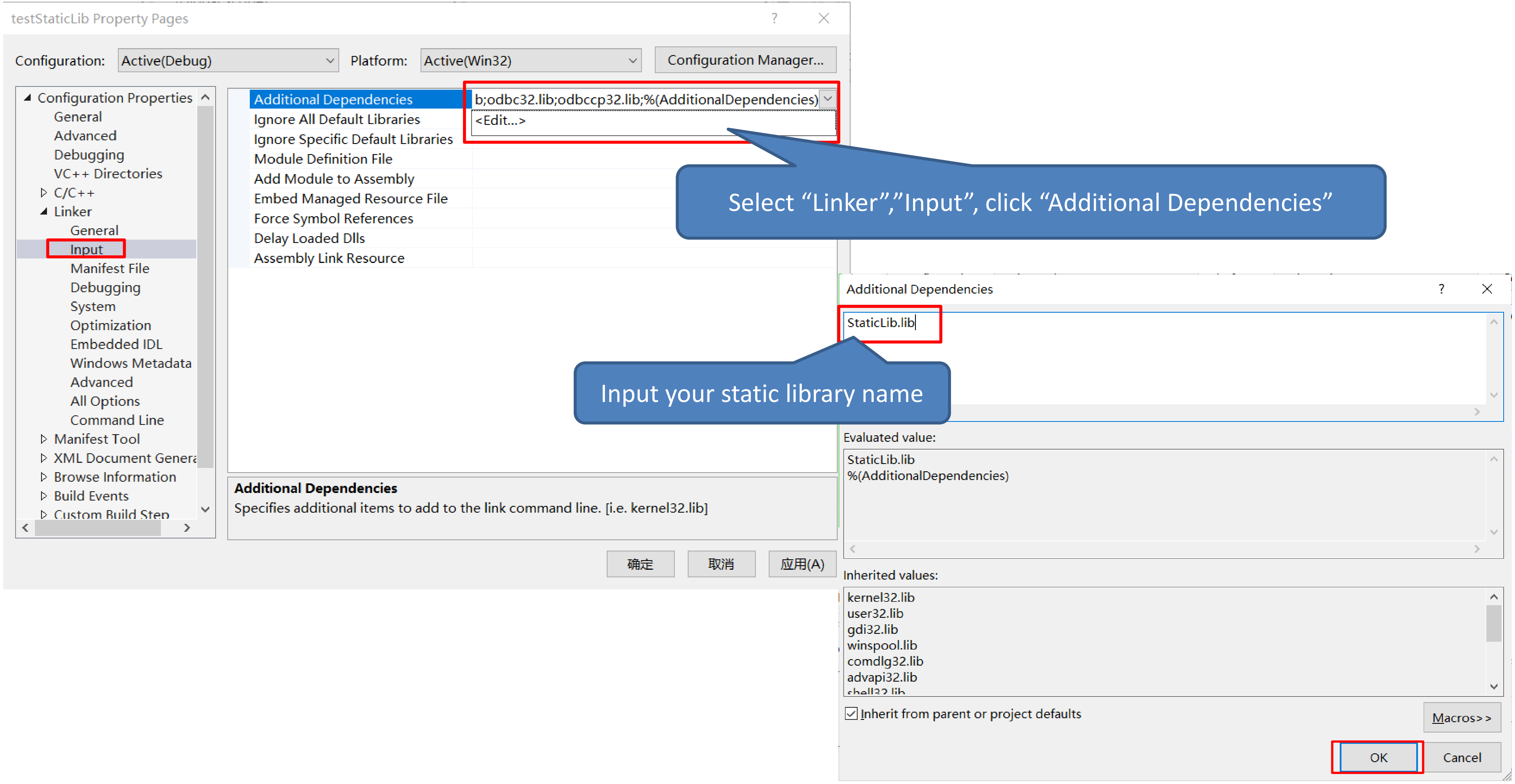
Platform: Active(Win32)

Output File	$(OutDir)$(TargetName)$(TargetExt)
Show Progress	Not Set
Version	
Enable Incremental Linking	**Yes (/INCREMENTAL)**
Suppress Startup Banner	Yes (/NOLOGO)
Ignore Import Library	No
Register Output	No
Per-user Redirection	No
Additional Library Directories	**D:\cstudy\StaticLib\StaticLib\Debug**
Link Library Dependencies	Yes
Use Library Dependency Inputs	No
Link Status	
Prevent Dll Binding	
Treat Linker Warning As Errors	
Force File Output	
Create Hot Patchable Image	
Specify Section Attributes	

Additional Library Directories

Allows the user to override the environmental library path. (/LIBPATH:folder)

Select “Linker”,”General”, and input or copy the directory of the static library(.lib file) into the “Additional Library Directories”



Configuration: Active(Debug)

Platform: Active(Win32)

Additional Dependencies: b;odbc32.lib;odbccp32.lib;%(AdditionalDependencies)

Additional Dependencies
Specifies additional items to add to the link command line. [i.e. kernel32.lib]

Select “Linker”,”Input”, click “Additional Dependencies”

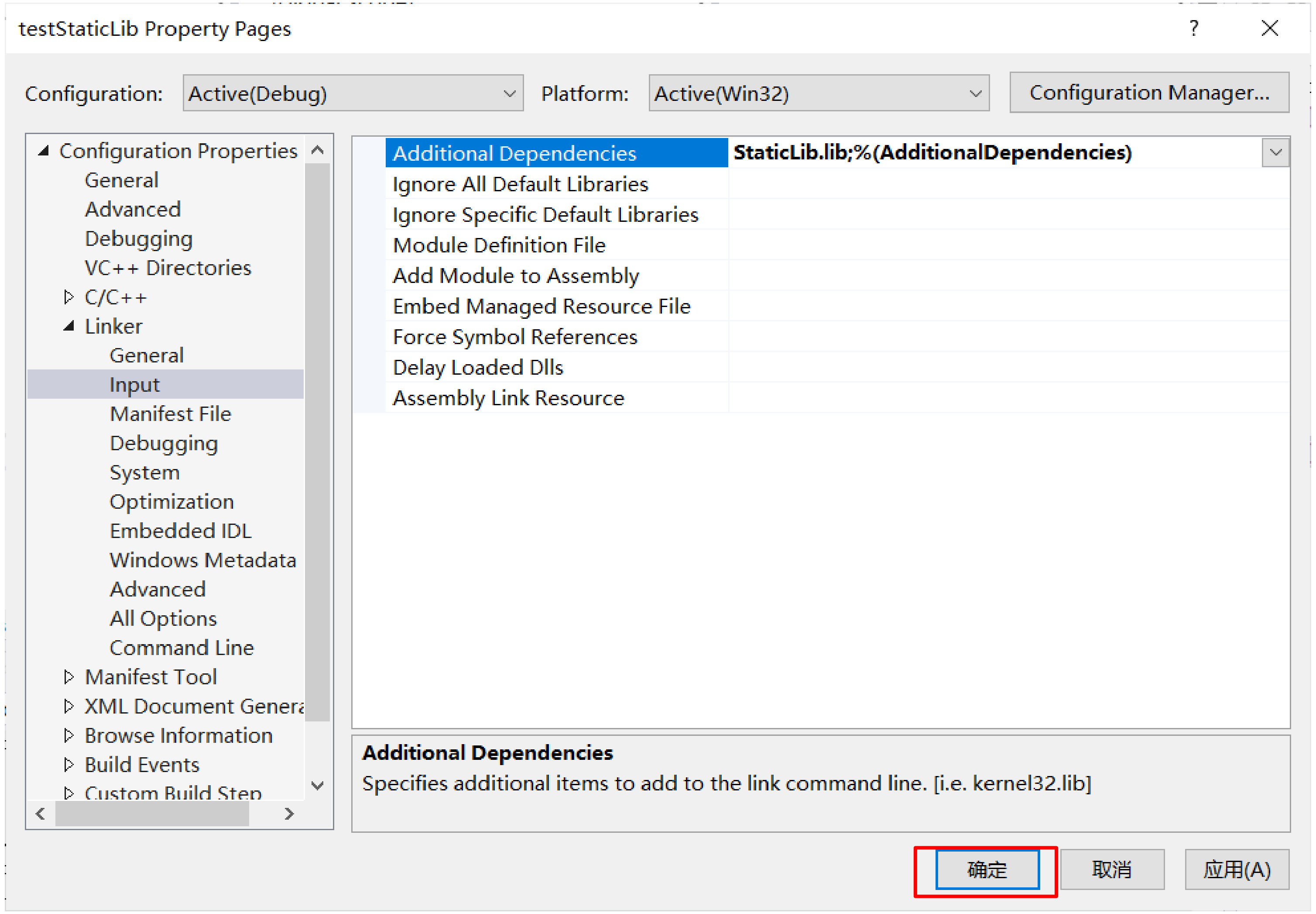
StaticLib.lib

Input your static library name

Evaluated value:
StaticLib.lib
%(AdditionalDependencies)

Inherited values:
kernel32.lib
user32.lib
gdi32.lib
winspool.lib
comdlg32.lib
advapi32.lib
shell32.lib

☑ Inherit from parent or project defaults



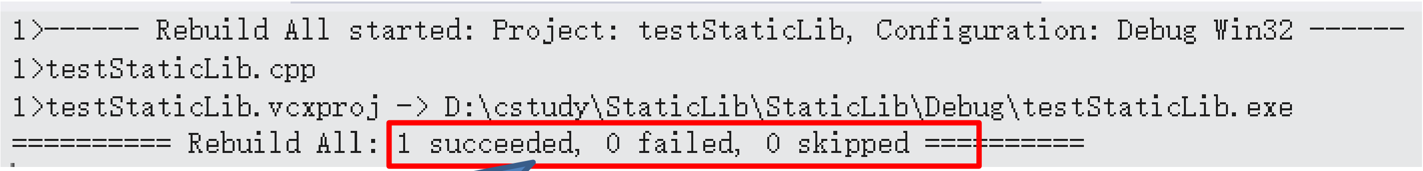
testStaticLib Property Pages

Configuration: Active(Debug)   Platform: Active(Win32)

Additional Dependencies	StaticLib.lib;%(AdditionalDependencies)
Ignore All Default Libraries	
Ignore Specific Default Libraries	
Module Definition File	
Add Module to Assembly	
Embed Managed Resource File	
Force Symbol References	
Delay Loaded Dlls	
Assembly Link Resource	

Additional Dependencies

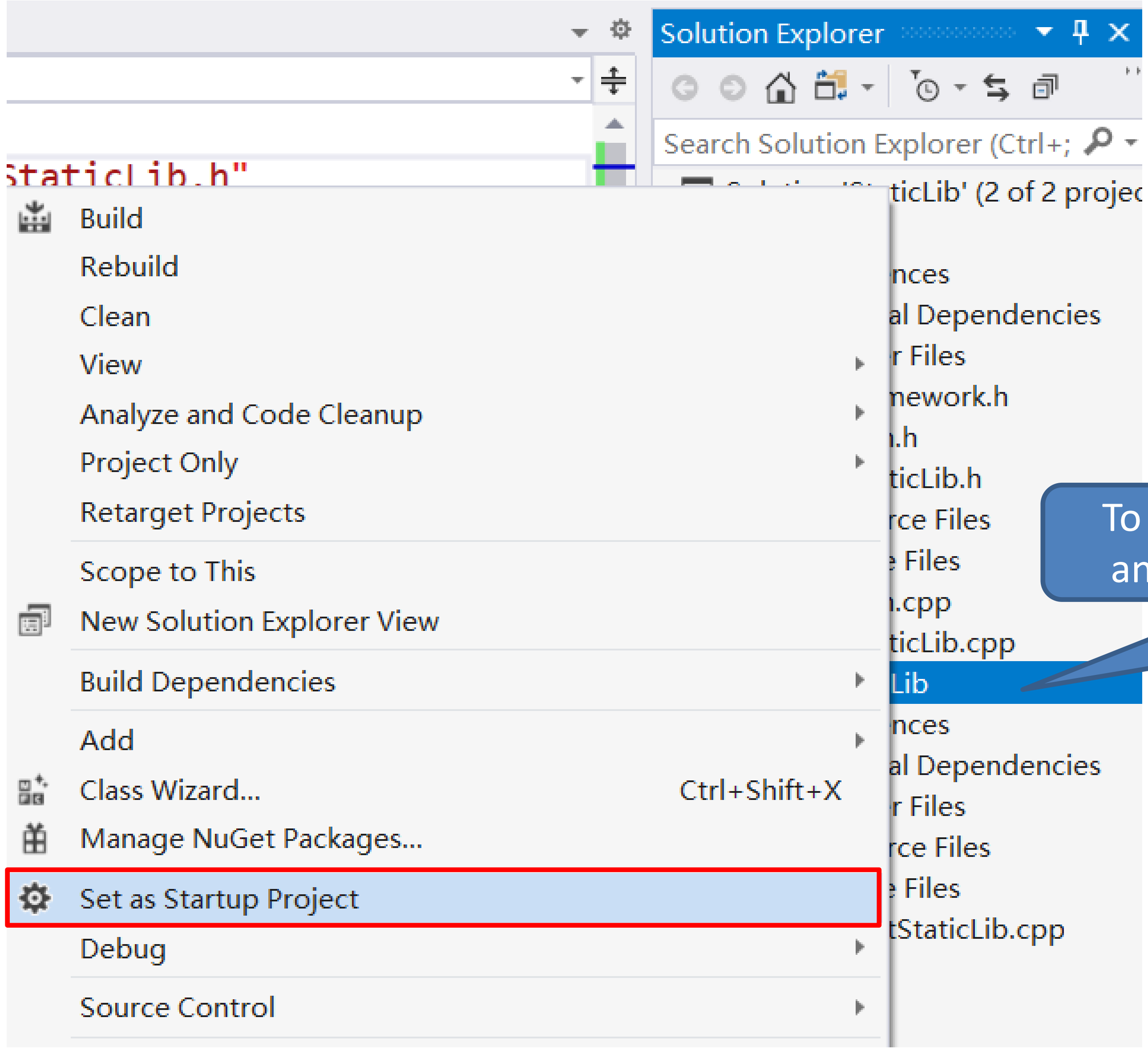
Specifies additional items to add to the link command line. [i.e. kernel32.lib]



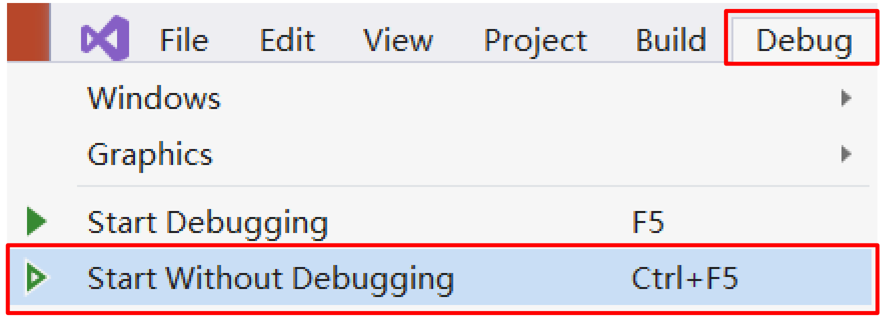
```
1>------ Rebuild All started: Project: testStaticLib, Configuration: Debug Win32 ------
1>testStaticLib.cpp
1>testStaticLib.vcxproj -> D:\cstudy\StaticLib\StaticLib\Debug\testStaticLib.exe
========== Rebuild All: 1 succeeded, 0 failed, 0 skipped ==========
```

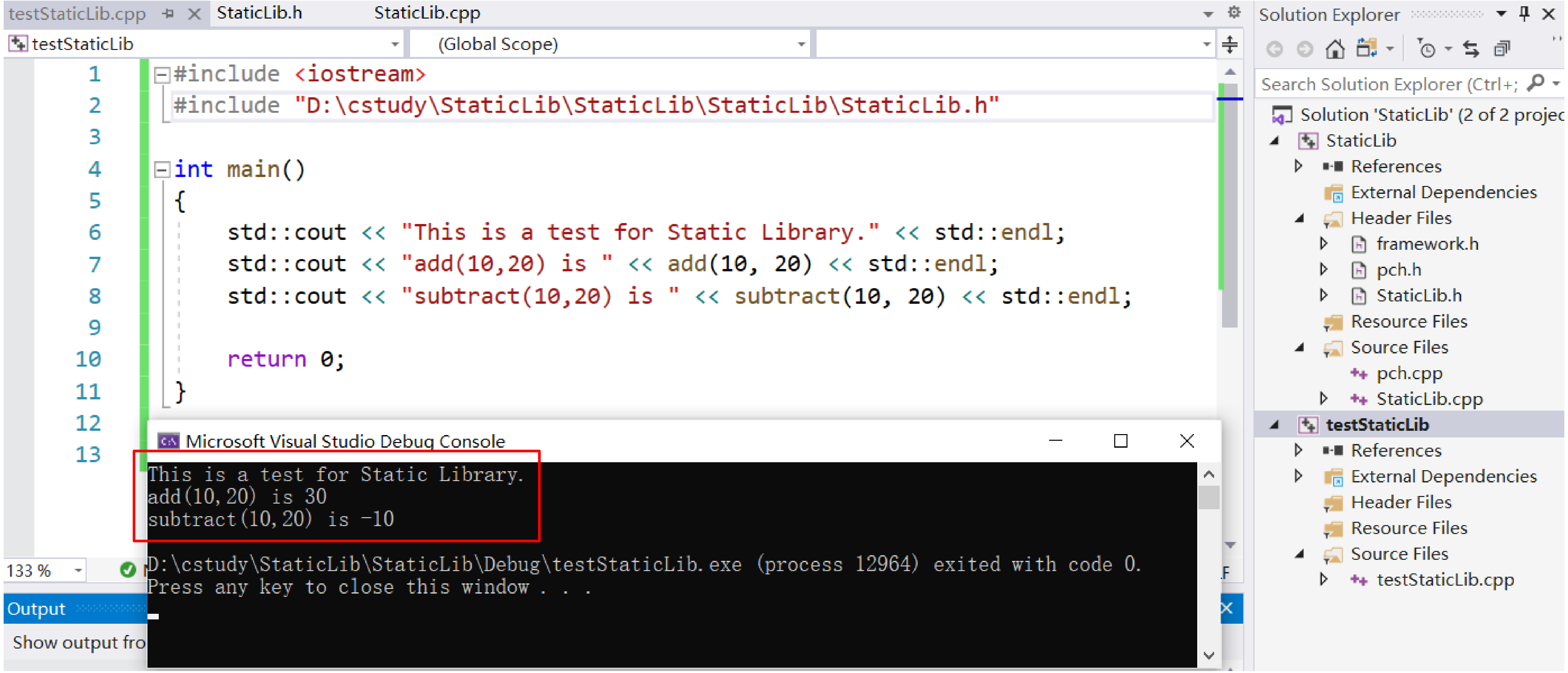
Rebuild your project, this time succeeded.

(10) Run the project



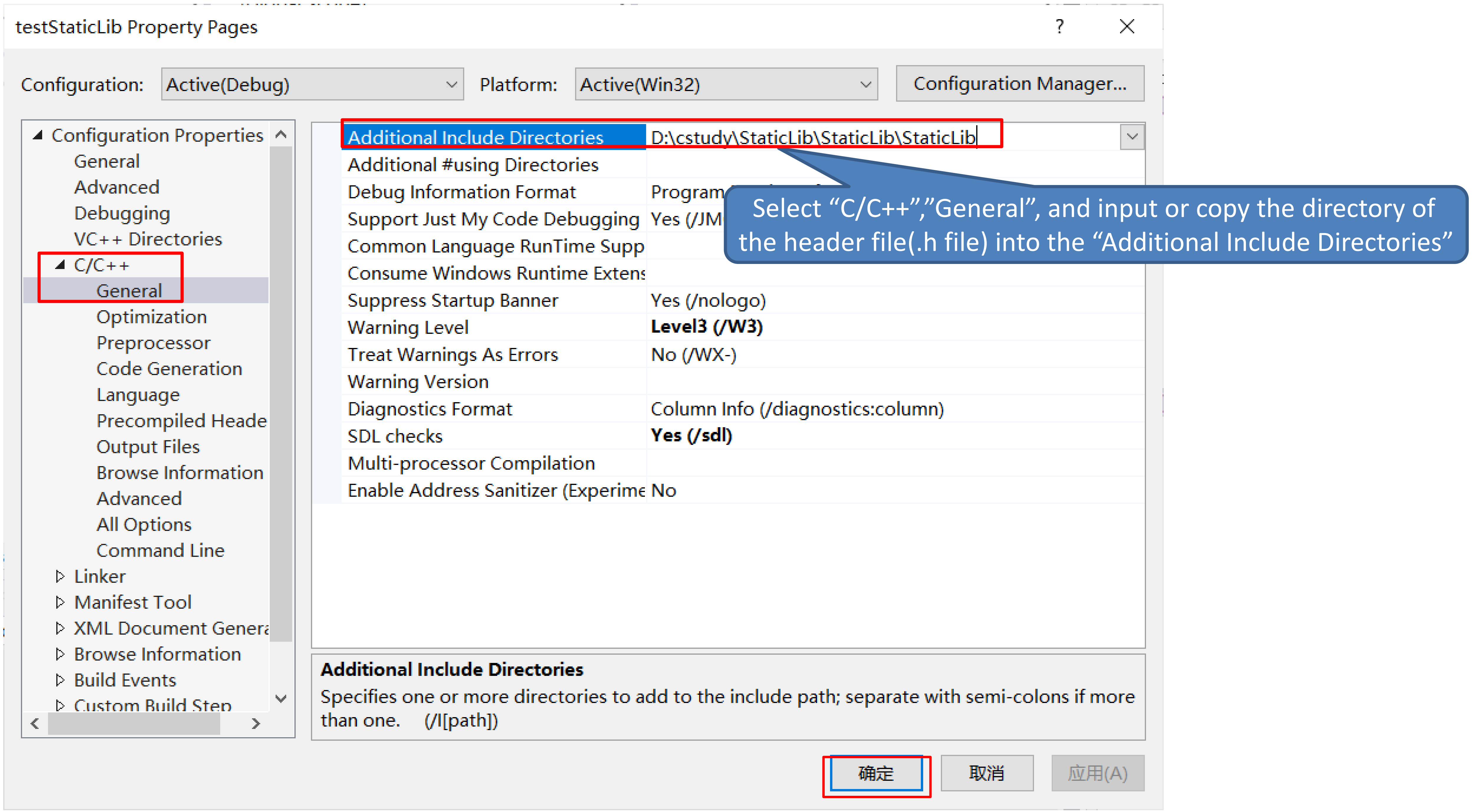
To run your project, first set the cursor at “testStaticLib’” project, and right click the mouse button, select “Set as Startup Project”





```cpp
#include <iostream>
#include "D:\cstudy\StaticLib\StaticLib\StaticLib\StaticLib.h"

int main()
{
    std::cout << "This is a test for Static Library." << std::endl;
    std::cout << "add(10,20) is " << add(10, 20) << std::endl;
    std::cout << "subtract(10,20) is " << subtract(10, 20) << std::endl;

    return 0;
}
```

```
This is a test for Static Library.
add(10,20) is 30
subtract(10,20) is -10

D:\cstudy\StaticLib\StaticLib\Debug\testStaticLib.exe (process 12964) exited with code 0.
Press any key to close this window . . .
```

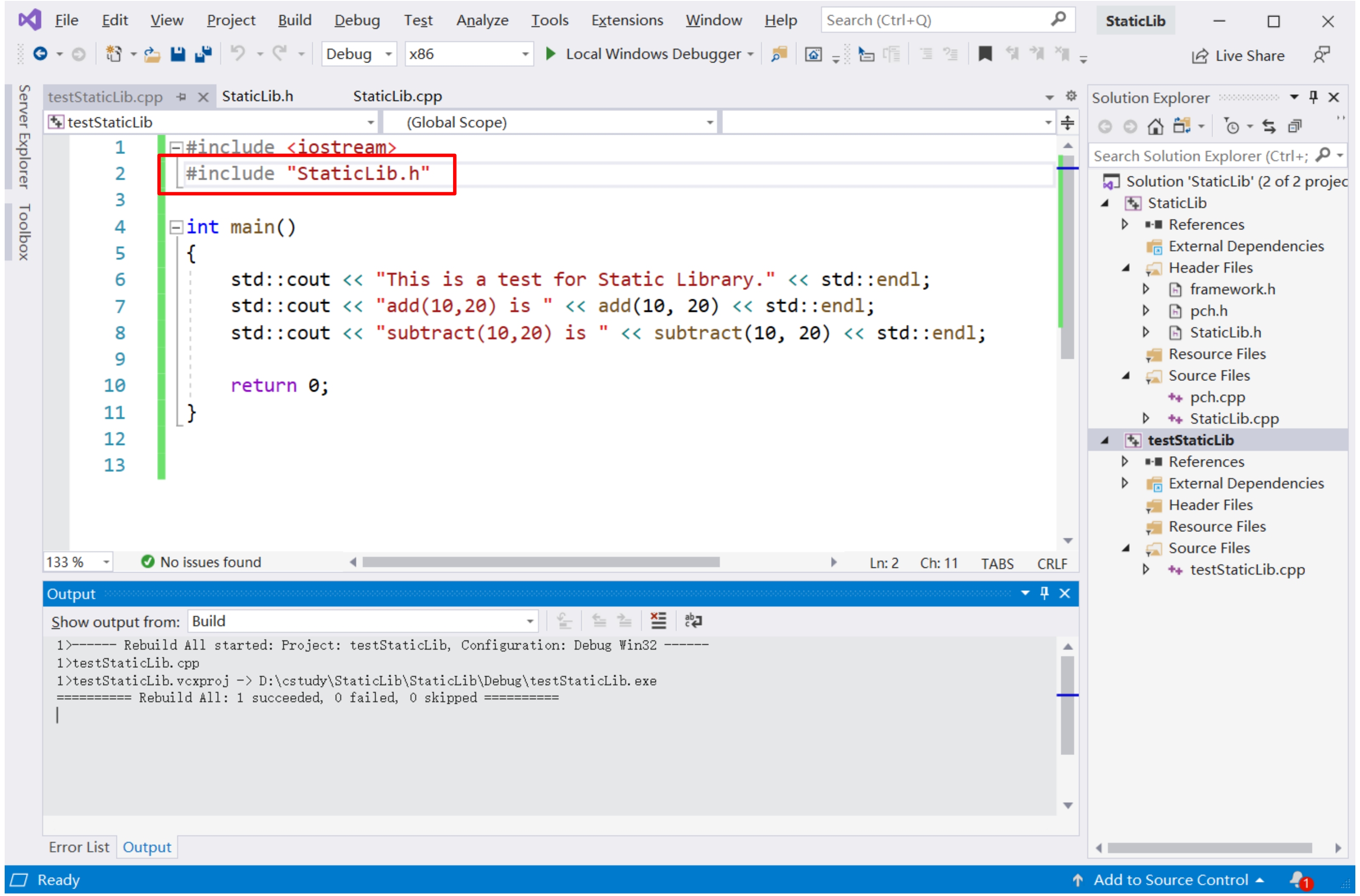


Additional Include Directories	D:\cstudy\StaticLib\StaticLib\StaticLib
Additional #using Directories	
Debug Information Format	Program
Support Just My Code Debugging	Yes (/JM
Common Language RunTime Supp	
Consume Windows Runtime Extens	
Suppress Startup Banner	Yes (/nologo)
Warning Level	**Level3 (/W3)**
Treat Warnings As Errors	No (/WX-)
Warning Version	
Diagnostics Format	Column Info (/diagnostics:column)
SDL checks	**Yes (/sdl)**
Multi-processor Compilation	
Enable Address Sanitizer (Experime	No

Additional Include Directories

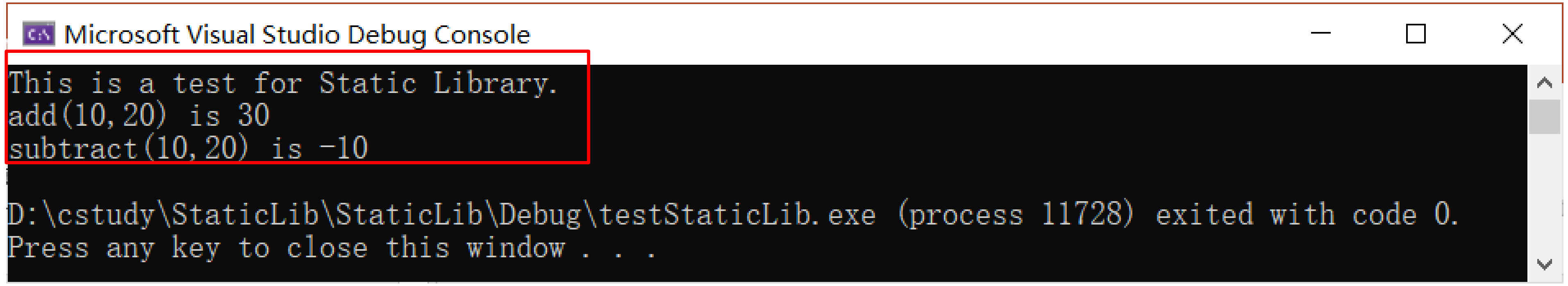
Specifies one or more directories to add to the include path; separate with semi-colons if more than one. (/I[path])

Select “C/C++”,”General”, and input or copy the directory of the header file(.h file) into the “Additional Include Directories”



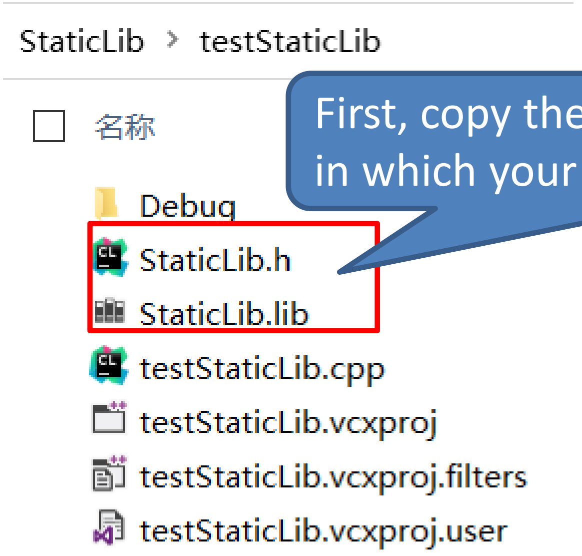
```
#include <iostream>
#include "StaticLib.h"

int main()
{
    std::cout << "This is a test for Static Library." << std::endl;
    std::cout << "add(10,20) is " << add(10, 20) << std::endl;
    std::cout << "subtract(10,20) is " << subtract(10, 20) << std::endl;

    return 0;
}
```

```
1>------ Rebuild All started: Project: testStaticLib, Configuration: Debug Win32 ------
1>testStaticLib.cpp
1>testStaticLib.vcxproj -> D:\cstudy\StaticLib\StaticLib\Debug\testStaticLib.exe
========== Rebuild All: 1 succeeded, 0 failed, 0 skipped ==========
```



```
This is a test for Static Library.
add(10,20) is 30
subtract(10,20) is -10

D:\cstudy\StaticLib\StaticLib\Debug\testStaticLib.exe (process 11728) exited with code 0.
Press any key to close this window . . .
```

You can use `#pragma comment(lib, "XXXXXX.lib") without setting the linker.`

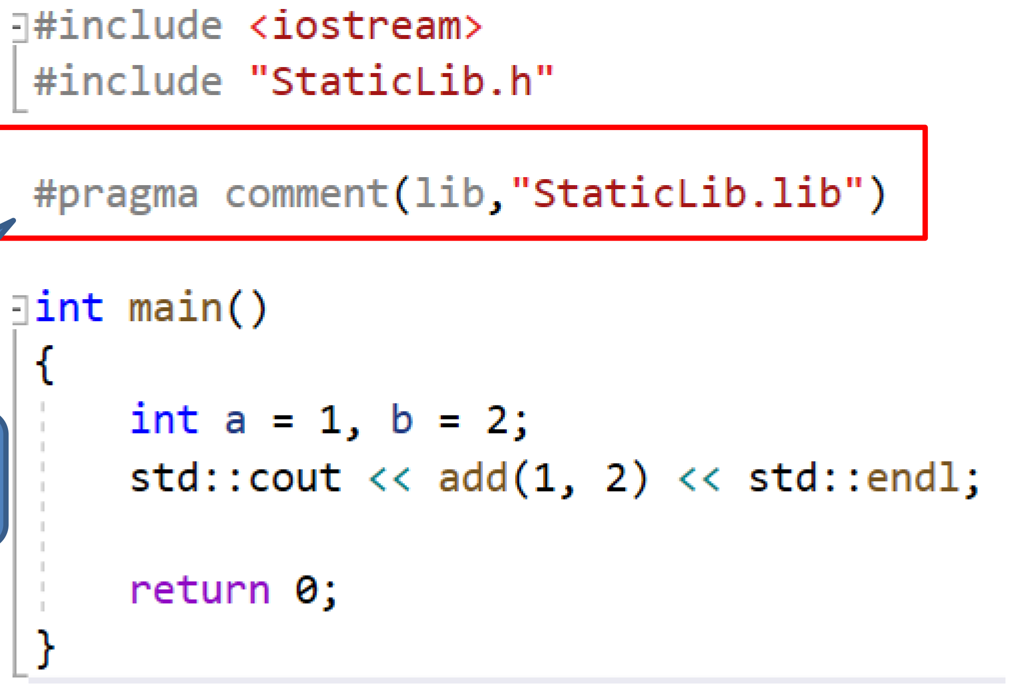


StaticLib › testStaticLib

□ 名称

- Debug
- StaticLib.h
- StaticLib.lib
- testStaticLib.cpp
- testStaticLib.vcxproj
- testStaticLib.vcxproj.filters
- testStaticLib.vcxproj.user

First, copy the **.h** and **.lib** files to the folder in which your main program is.



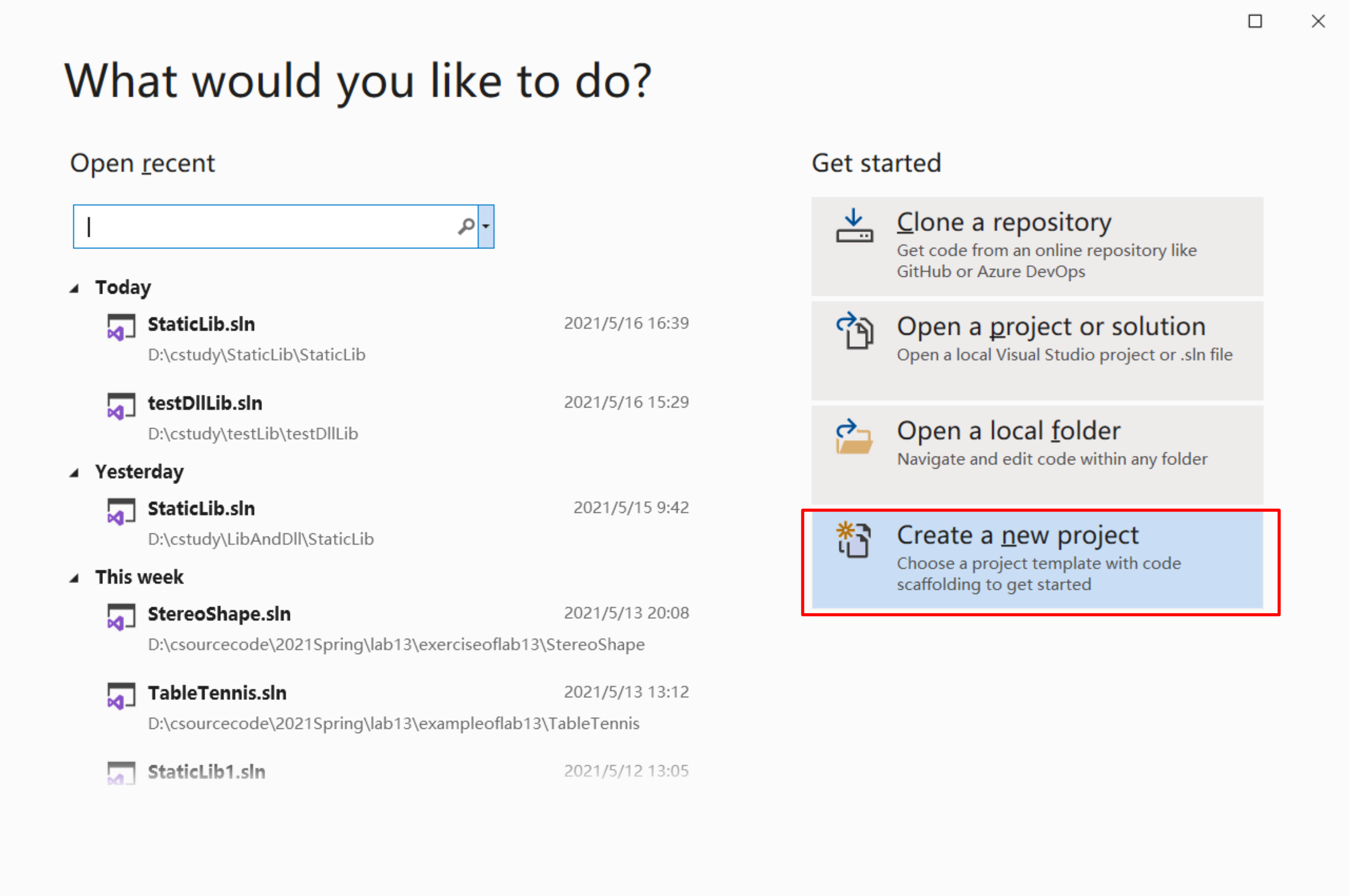
```
#include <iostream>
#include "StaticLib.h"

#pragma comment(lib,"StaticLib.lib")

int main()
{
    int a = 1, b = 2;
    std::cout << add(1, 2) << std::endl;

    return 0;
}
```

Second, include the .h and use the command #pragma comment in the main program.

https://docs.microsoft.com/en-us/cpp/preprocessor/comment-c-cpp?view=msvc-160

4. Create Dynamic Library

(1) Create a new project in VS 2019



What would you like to do?

Open recent

- Today
  - StaticLib.sln 2021/5/16 16:39 D:\cstudy\StaticLib\StaticLib
  - testDllLib.sln 2021/5/16 15:29 D:\cstudy\testLib\testDllLib
- Yesterday
  - StaticLib.sln 2021/5/15 9:42 D:\cstudy\LibAndDll\StaticLib
- This week
  - StereoShape.sln 2021/5/13 20:08 D:\csourcecode\2021Spring\lab13\exerciseoflab13\StereoShape
  - TableTennis.sln 2021/5/13 13:12 D:\csourcecode\2021Spring\lab13\exampleoflab13\TableTennis
  - StaticLib1.sln 2021/5/12 13:05

Get started

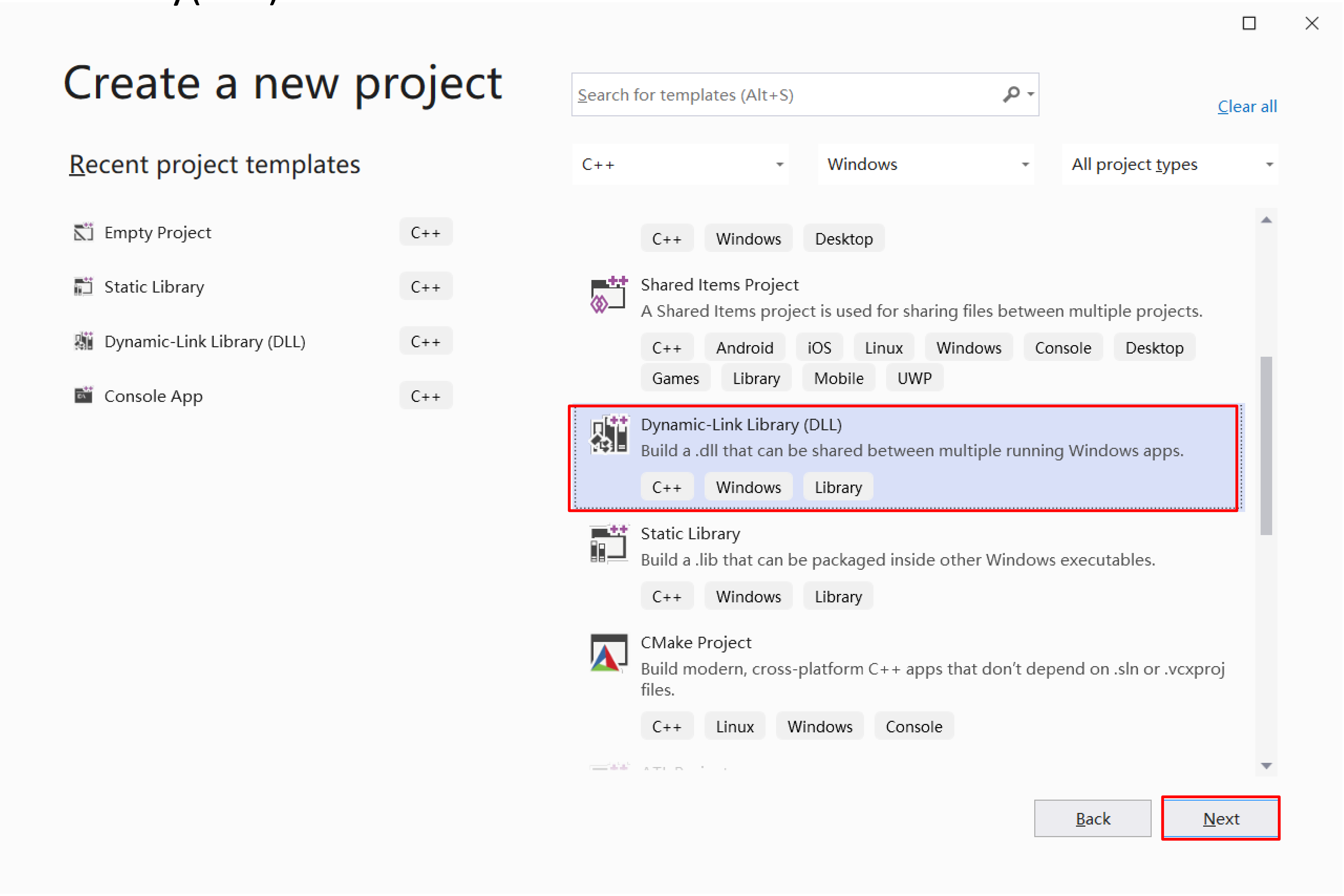
Clone a repository
Get code from an online repository like GitHub or Azure DevOps

Open a project or solution
Open a local Visual Studio project or .sln file

Open a local folder
Navigate and edit code within any folder

Create a new project
Choose a project template with code scaffolding to get started

(2) Select Dynamic-Link Library(DLL)



Create a new project

Search for templates (Alt+S)

Clear all

Recent project templates

Empty Project	C++
Static Library	C++
Dynamic-Link Library (DLL)	C++
Console App	C++

C++ ▾ Windows ▾ All project types ▾

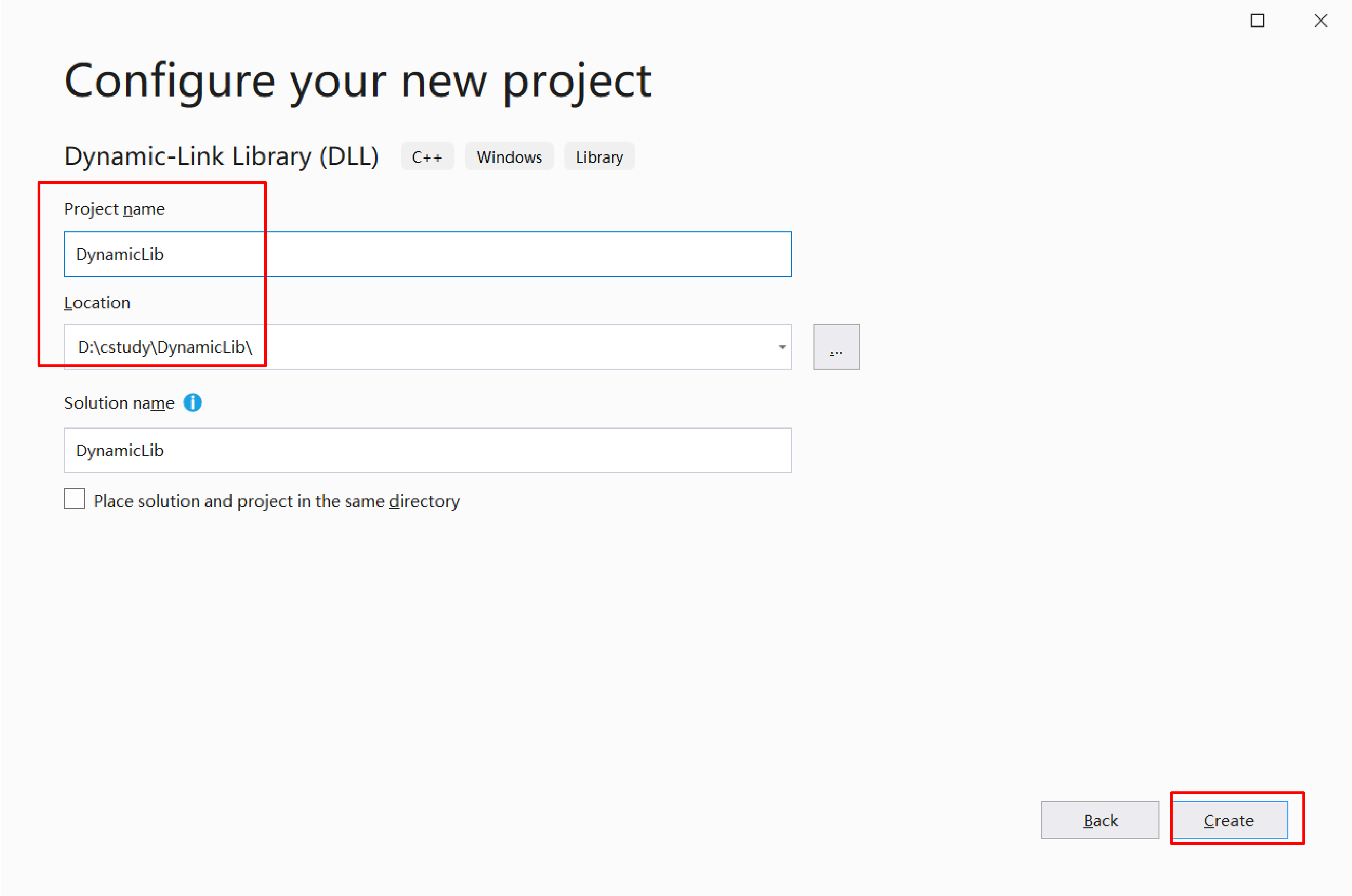
C++ Windows Desktop

Shared Items Project
A Shared Items project is used for sharing files between multiple projects.
C++ Android iOS Linux Windows Console Desktop Games Library Mobile UWP

Dynamic-Link Library (DLL)
Build a .dll that can be shared between multiple running Windows apps.
C++ Windows Library

Static Library
Build a .lib that can be packaged inside other Windows executables.
C++ Windows Library

CMake Project
Build modern, cross-platform C++ apps that don't depend on .sln or .vcxproj files.
C++ Linux Windows Console



Configure your new project

Dynamic-Link Library (DLL) C++ Windows Library

Project name

DynamicLib

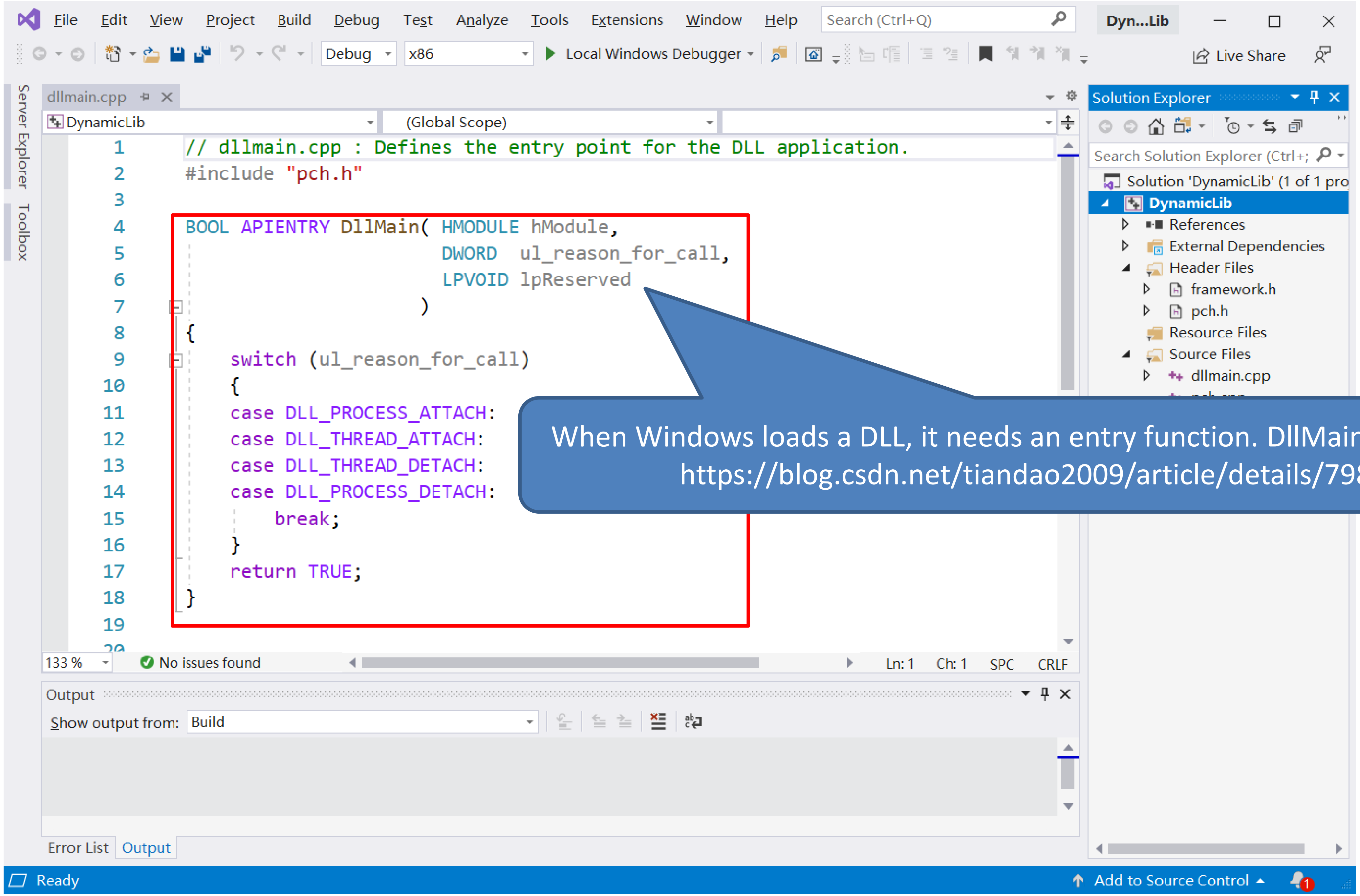
Location

D:\cstudy\DynamicLib\

Solution name

DynamicLib

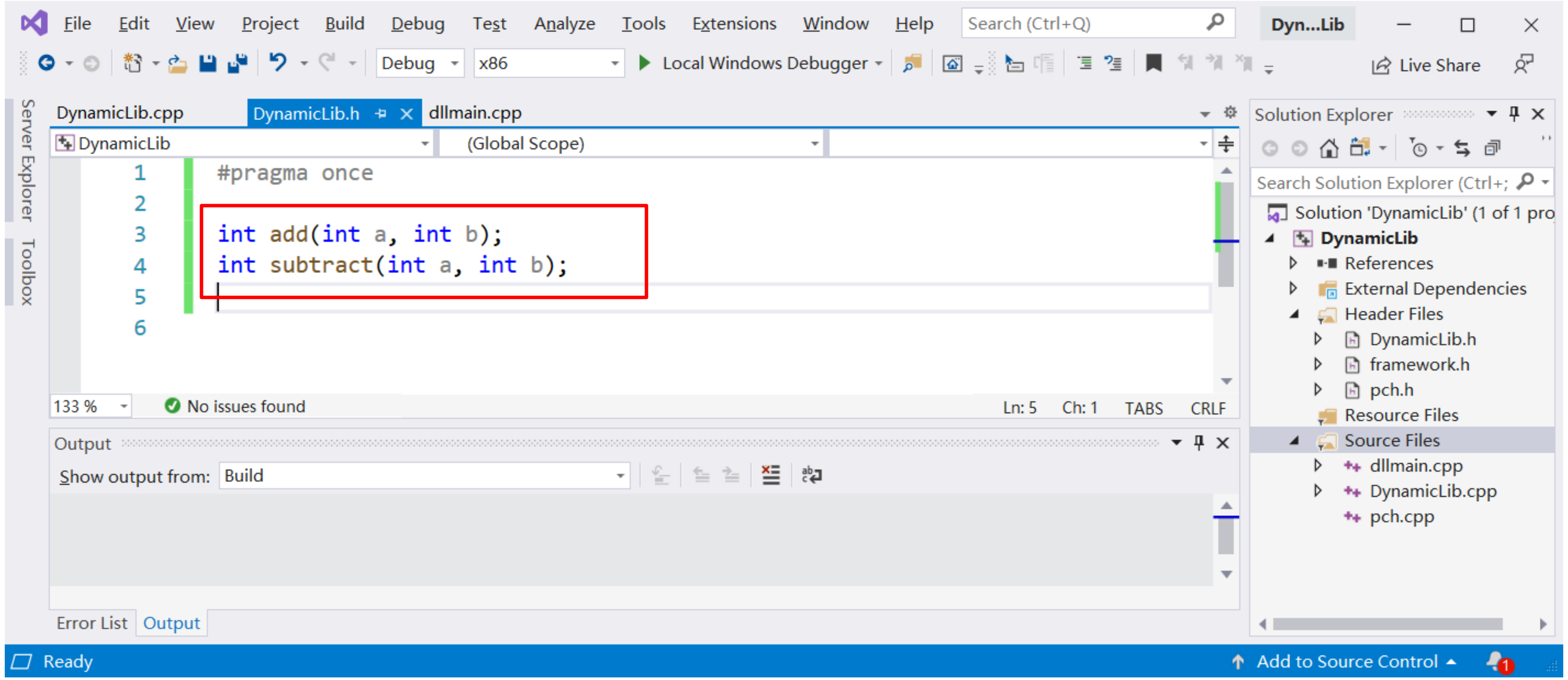
□ Place solution and project in the same directory



```
// dllmain.cpp : Defines the entry point for the DLL application.
#include "pch.h"

BOOL APIENTRY DllMain( HMODULE hModule,
                       DWORD  ul_reason_for_call,
                       LPVOID lpReserved
                     )
{
    switch (ul_reason_for_call)
    {
    case DLL_PROCESS_ATTACH:
    case DLL_THREAD_ATTACH:
    case DLL_THREAD_DETACH:
    case DLL_PROCESS_DETACH:
        break;
    }
    return TRUE;
}
```

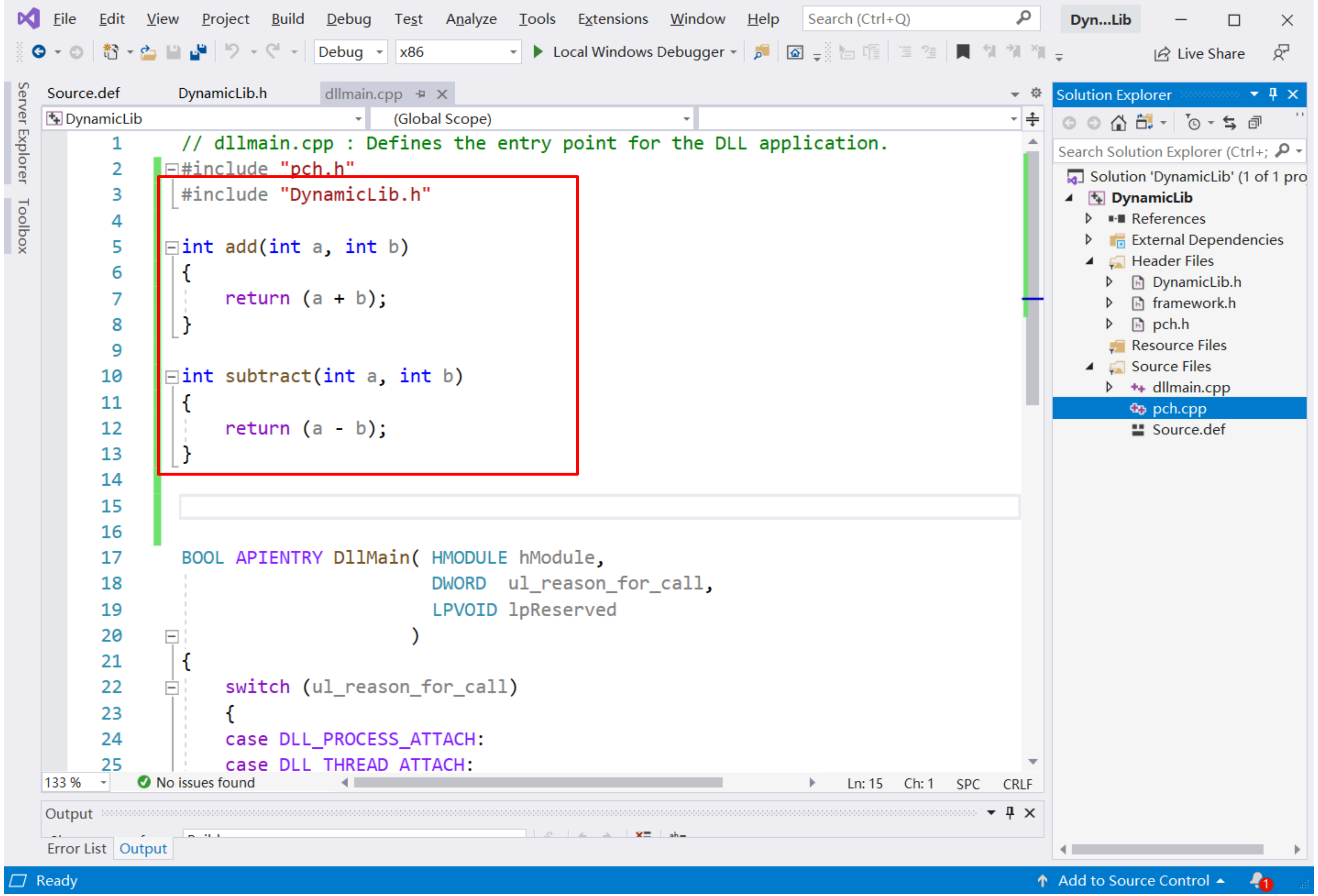
When Windows loads a DLL, it needs an entry function. DllMain is such function.
https://blog.csdn.net/tiandao2009/article/details/79839182

(3) Create .h to declare functions



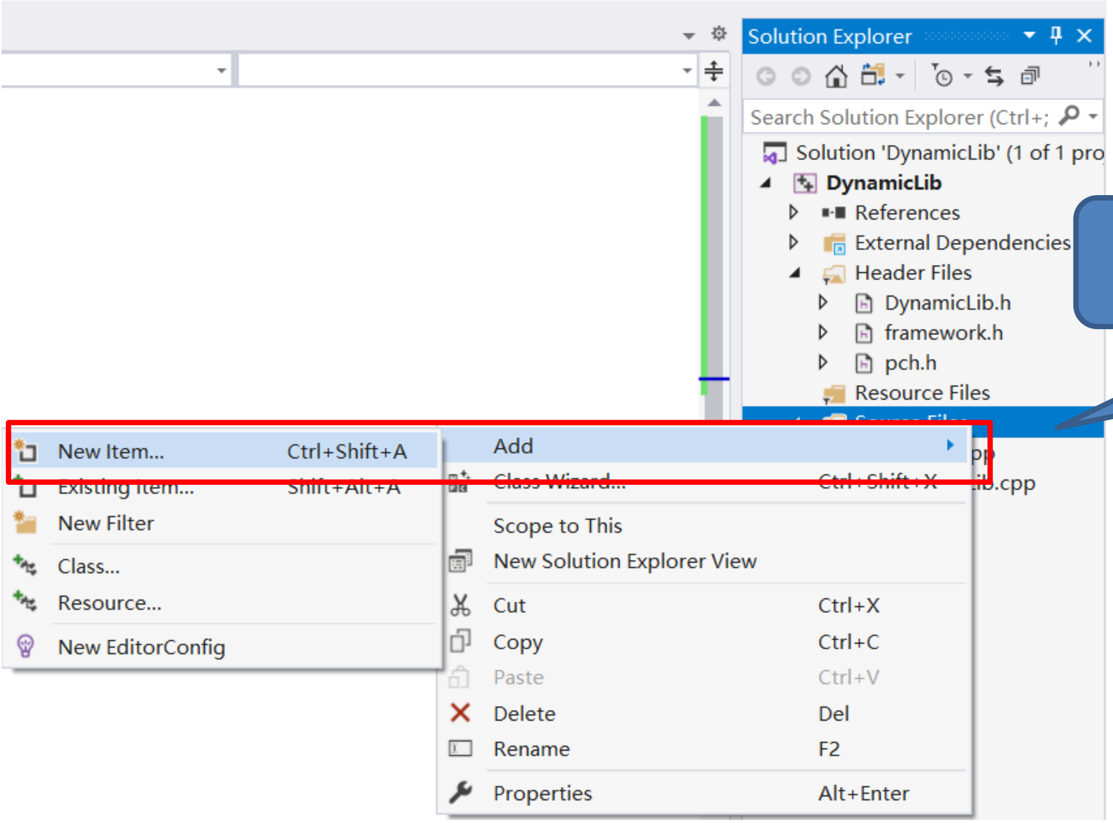
```
#pragma once

int add(int a, int b);
int subtract(int a, int b);
```

(4) Input functions implementation in DllMain.cpp

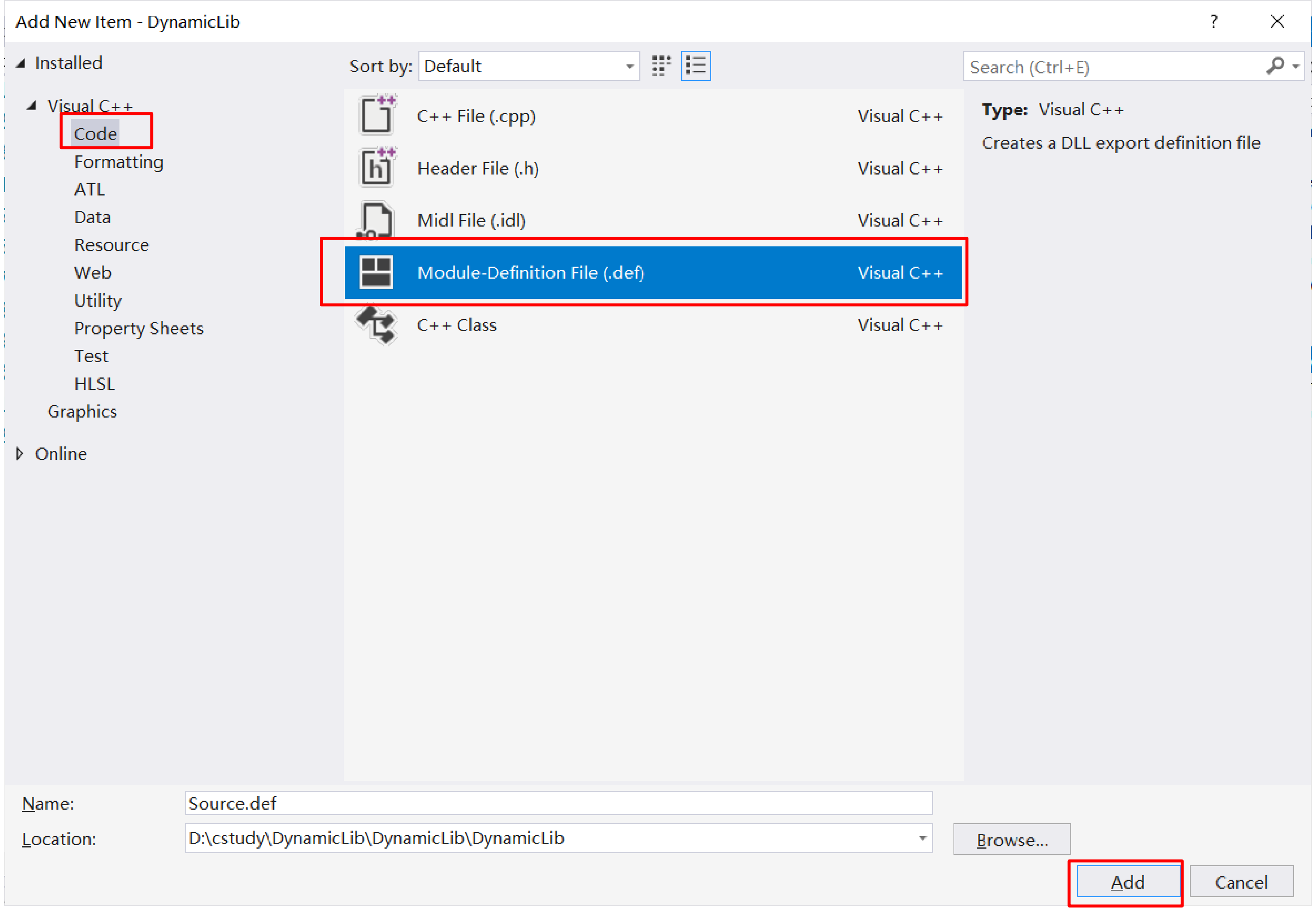


```
// dllmain.cpp : Defines the entry point for the DLL application.
#include "pch.h"
#include "DynamicLib.h"

int add(int a, int b)
{
    return (a + b);
}

int subtract(int a, int b)
{
    return (a - b);
}



BOOL APIENTRY DllMain( HMODULE hModule,
                       DWORD  ul_reason_for_call,
                       LPVOID lpReserved
                     )
{
    switch (ul_reason_for_call)
    {
    case DLL_PROCESS_ATTACH:
    case DLL_THREAD_ATTACH:
```

(5) Create .def file for exporting functions



Set the cursor at “Source Files’”, and right click the mouse button, select “Add”→”New Item…”

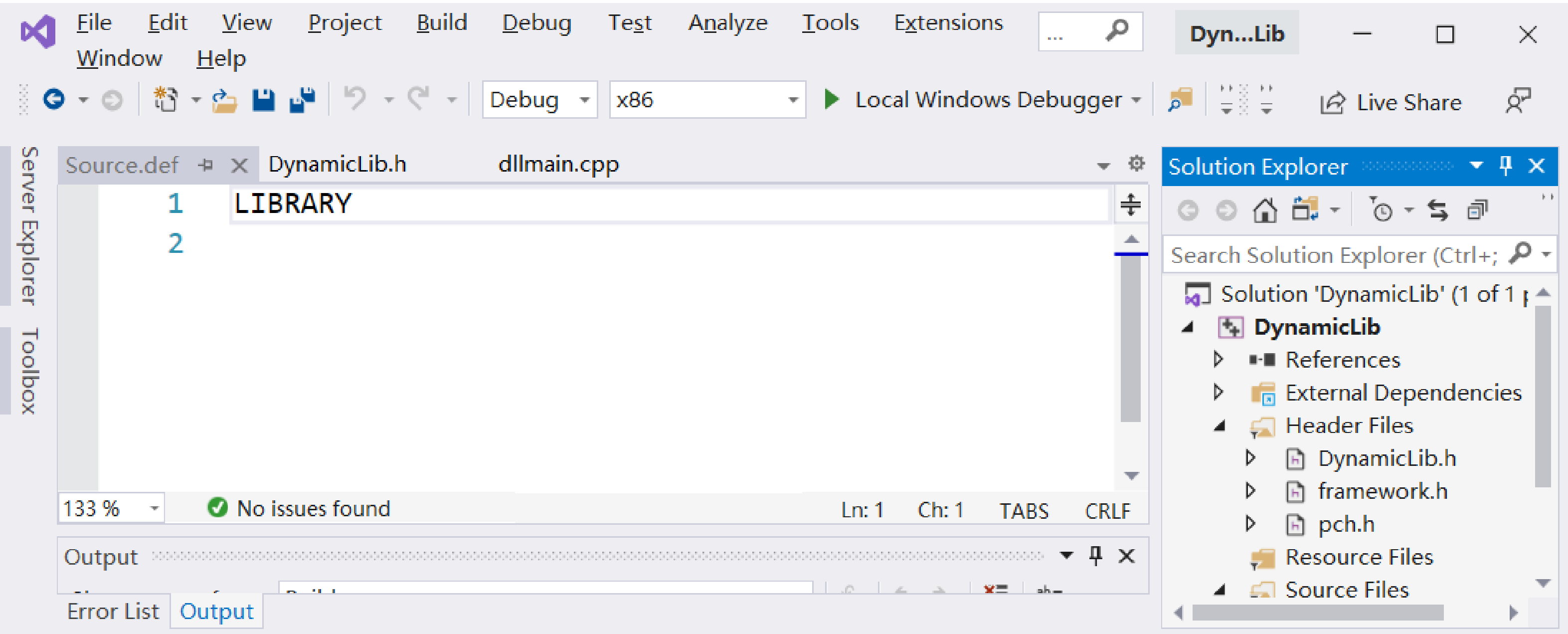


Type: Visual C++

Creates a DLL export definition file

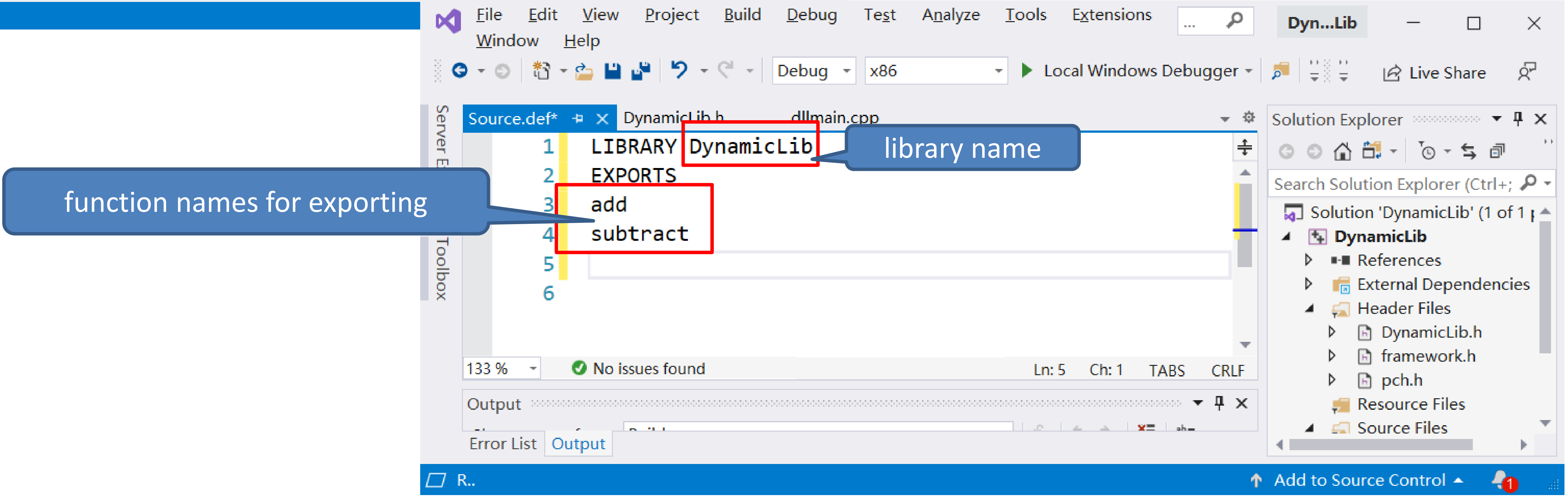
Name: Source.def

Location: D:\cstudy\DynamicLib\DynamicLib\DynamicLib



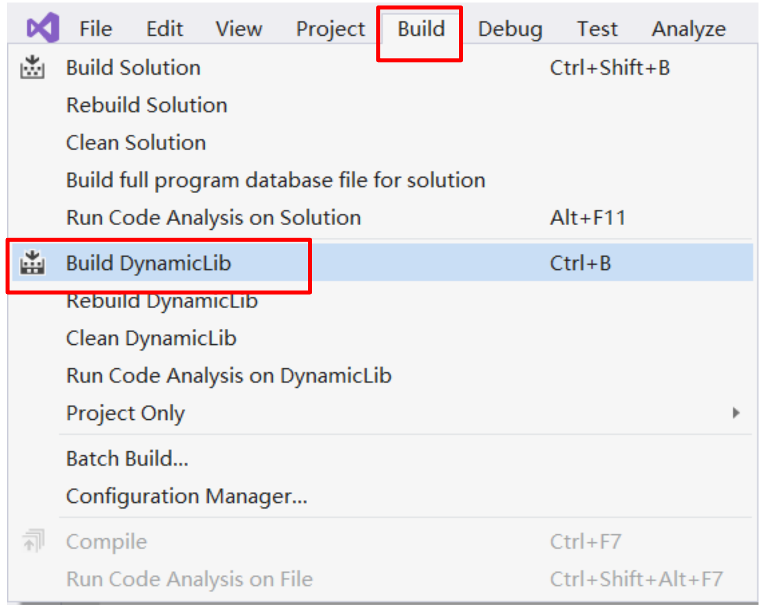
```
LIBRARY
```

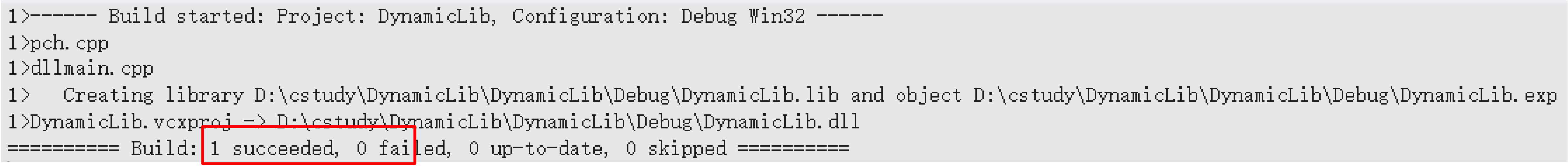
If you don't create .def file, you must define __declspec(dllexport) for exporting by using macro in .h file https://blog.csdn.net/a369189453/article/details/81124685



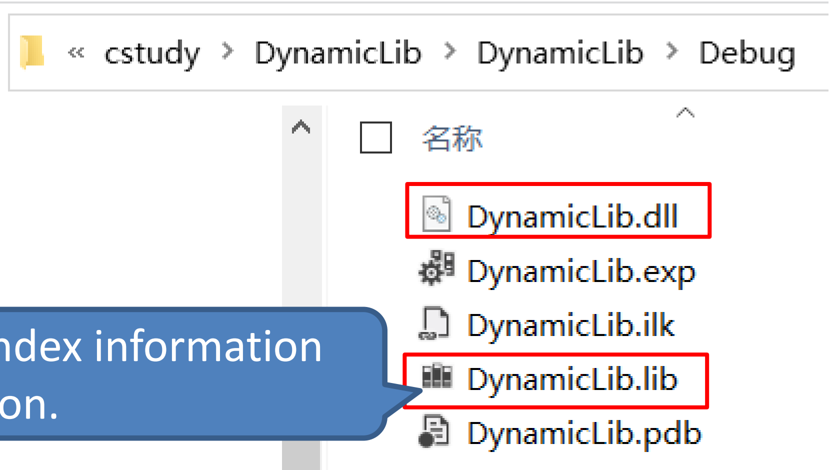
```
LIBRARY DynamicLib
EXPORTS
add
subtract
```

(6)Build dynamic library



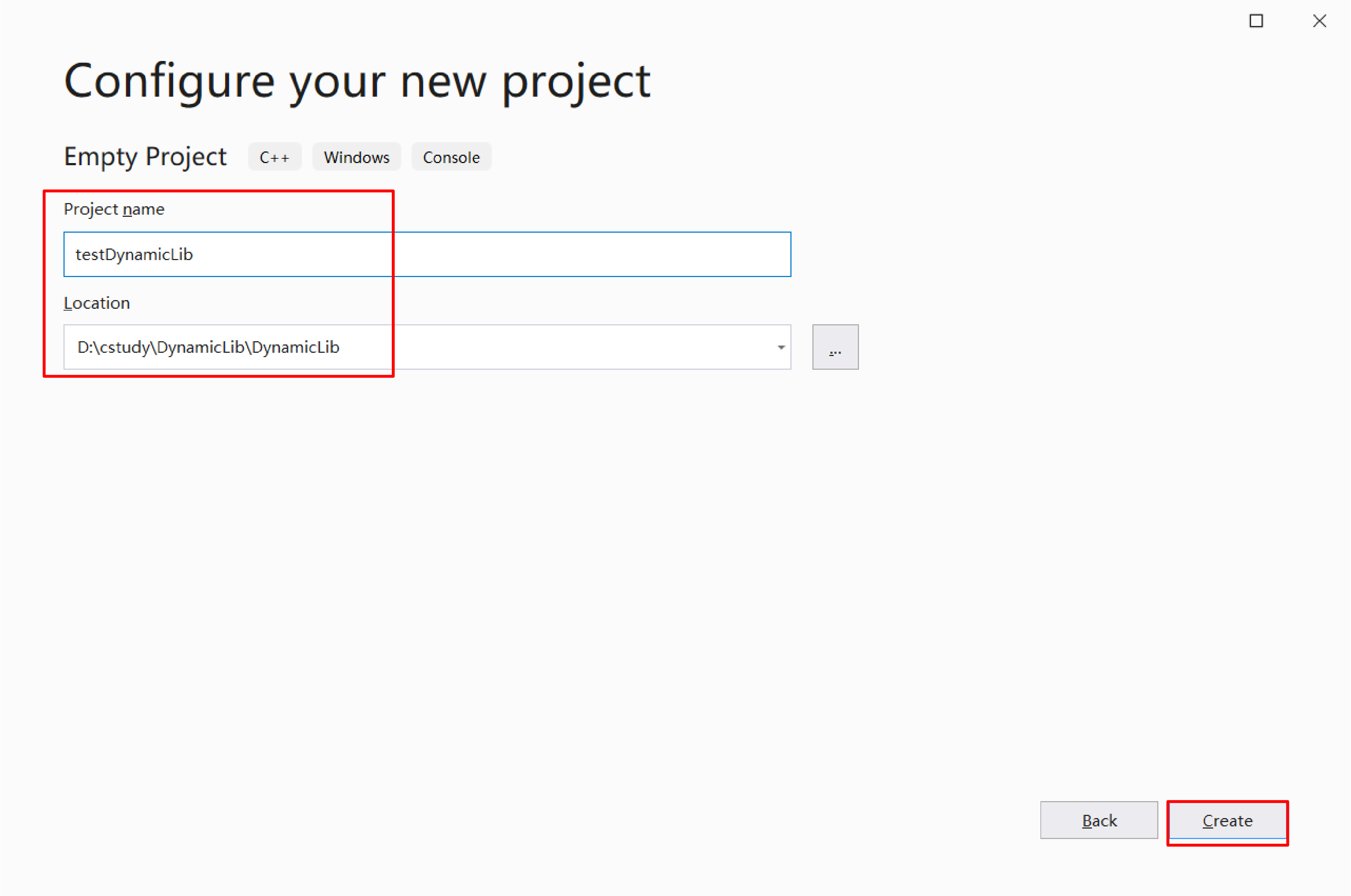


```
1>------ Build started: Project: DynamicLib, Configuration: Debug Win32 ------
1>pch.cpp
1>dllmain.cpp
1>   Creating library D:\cstudy\DynamicLib\DynamicLib\Debug\DynamicLib.lib and object D:\cstudy\DynamicLib\DynamicLib\Debug\DynamicLib.exp
1>DynamicLib.vcxproj -> D:\cstudy\DynamicLib\DynamicLib\Debug\DynamicLib.dll
========== Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped ==========
```



This .lib file is not the same as the static library. This .lib contains only index information such as function entry location rather than implementation.

(7)Create a new empty project to apply the dynamic library



Configure your new project

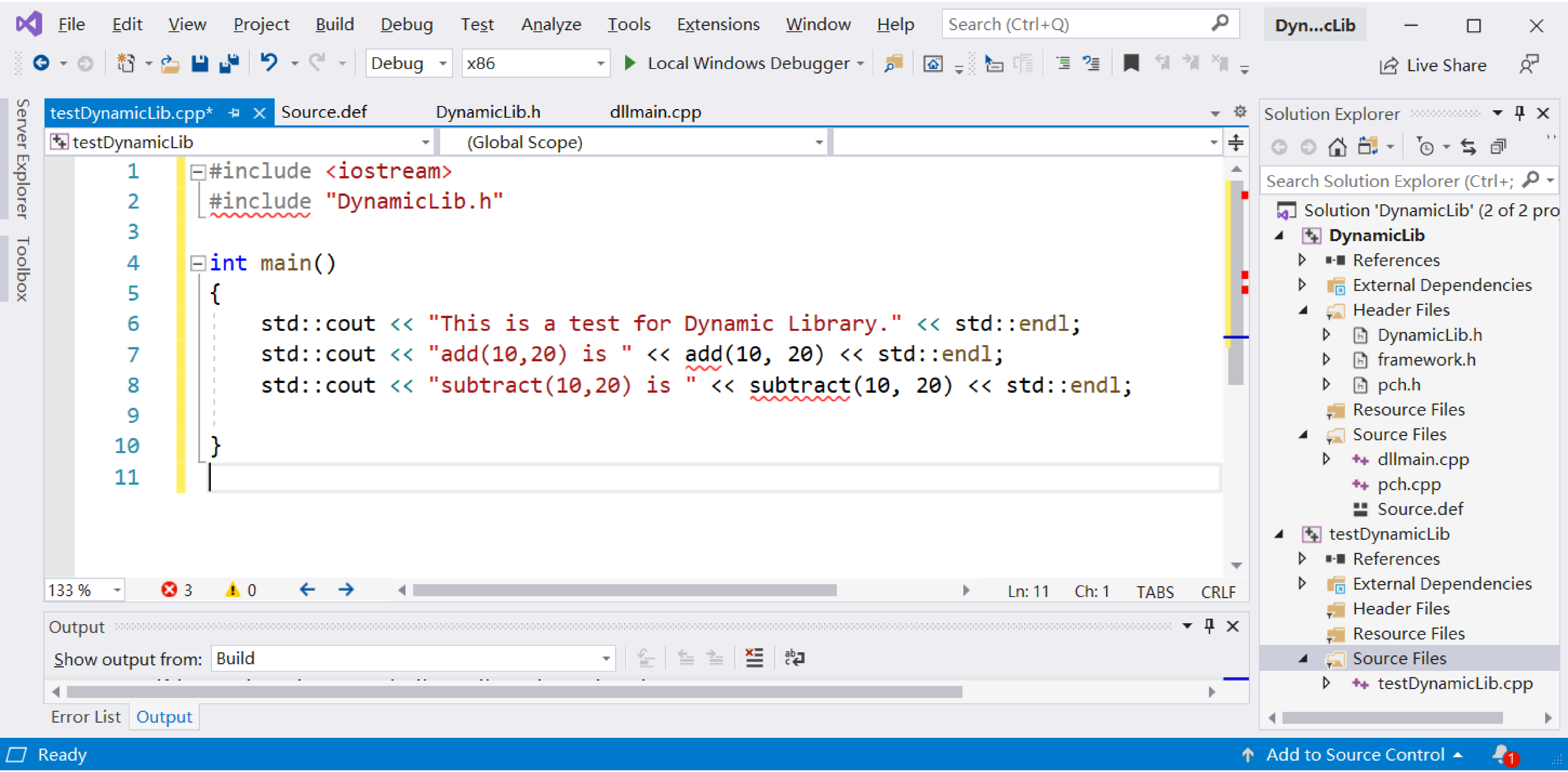
Empty Project C++ Windows Console

Project name

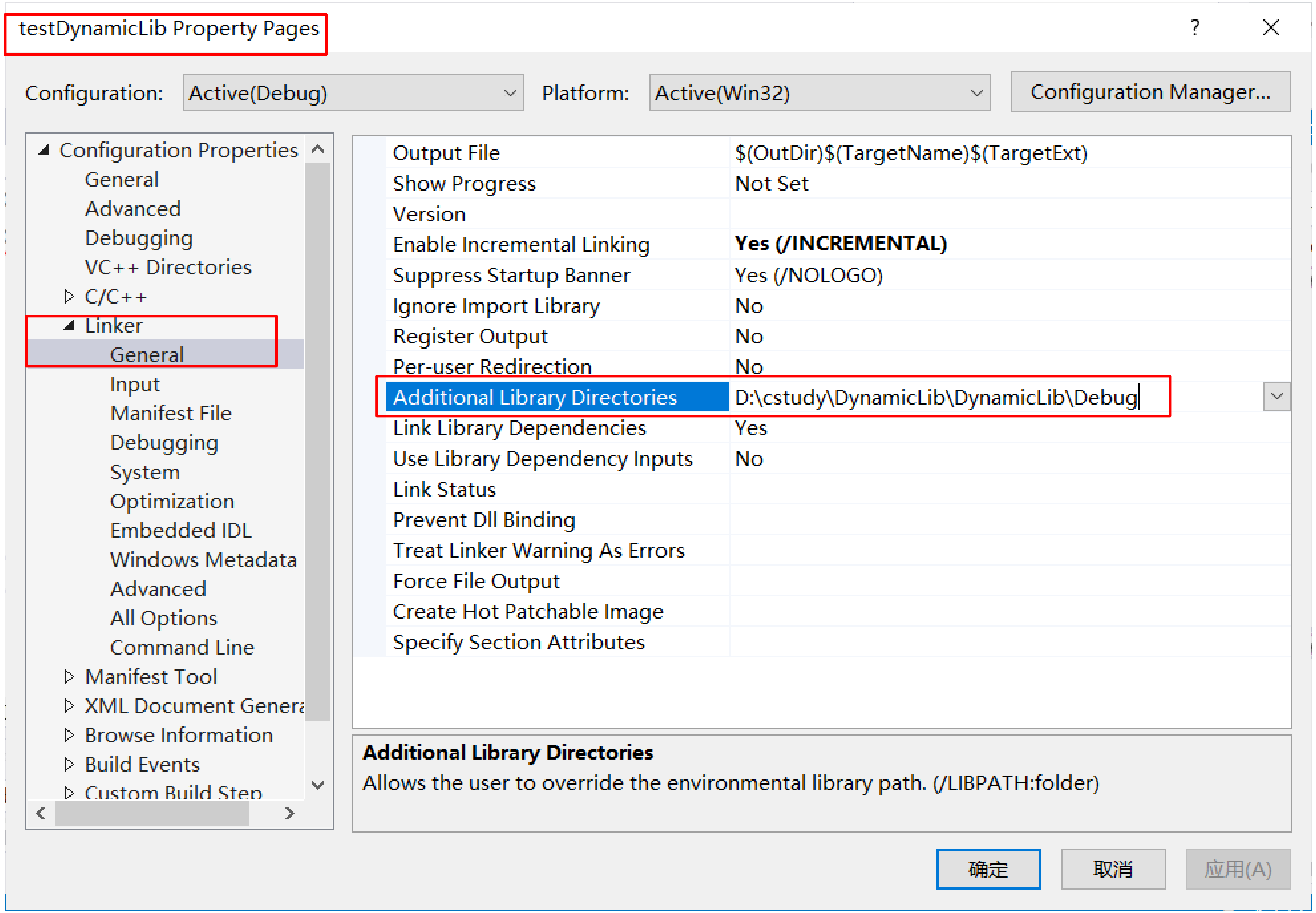
testDynamicLib

Location

D:\cstudy\DynamicLib\DynamicLib



```
#include <iostream>
#include "DynamicLib.h"

int main()
{
    std::cout << "This is a test for Dynamic Library." << std::endl;
    std::cout << "add(10,20) is " << add(10, 20) << std::endl;
    std::cout << "subtract(10,20) is " << subtract(10, 20) << std::endl;

}

```



testDynamicLib Property Pages

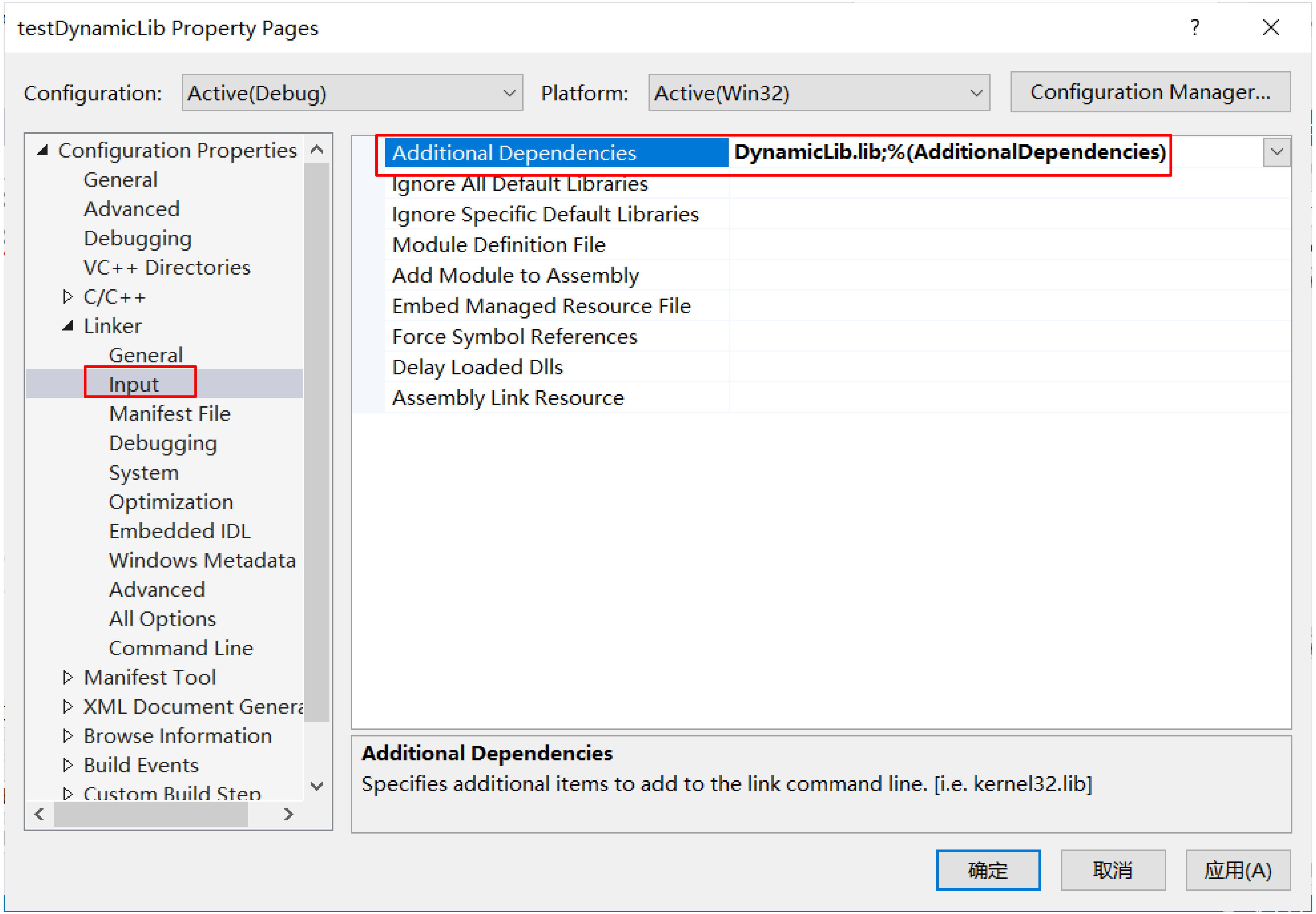
Configuration: Active(Debug)

Platform: Active(Win32)

Output File	$(OutDir)$(TargetName)$(TargetExt)
Show Progress	Not Set
Version	
Enable Incremental Linking	**Yes (/INCREMENTAL)**
Suppress Startup Banner	Yes (/NOLOGO)
Ignore Import Library	No
Register Output	No
Per-user Redirection	No
Additional Library Directories	D:\cstudy\DynamicLib\DynamicLib\Debug
Link Library Dependencies	Yes
Use Library Dependency Inputs	No
Link Status	
Prevent Dll Binding	
Treat Linker Warning As Errors	
Force File Output	
Create Hot Patchable Image	
Specify Section Attributes	

Additional Library Directories

Allows the user to override the environmental library path. (/LIBPATH:folder)



testDynamicLib Property Pages

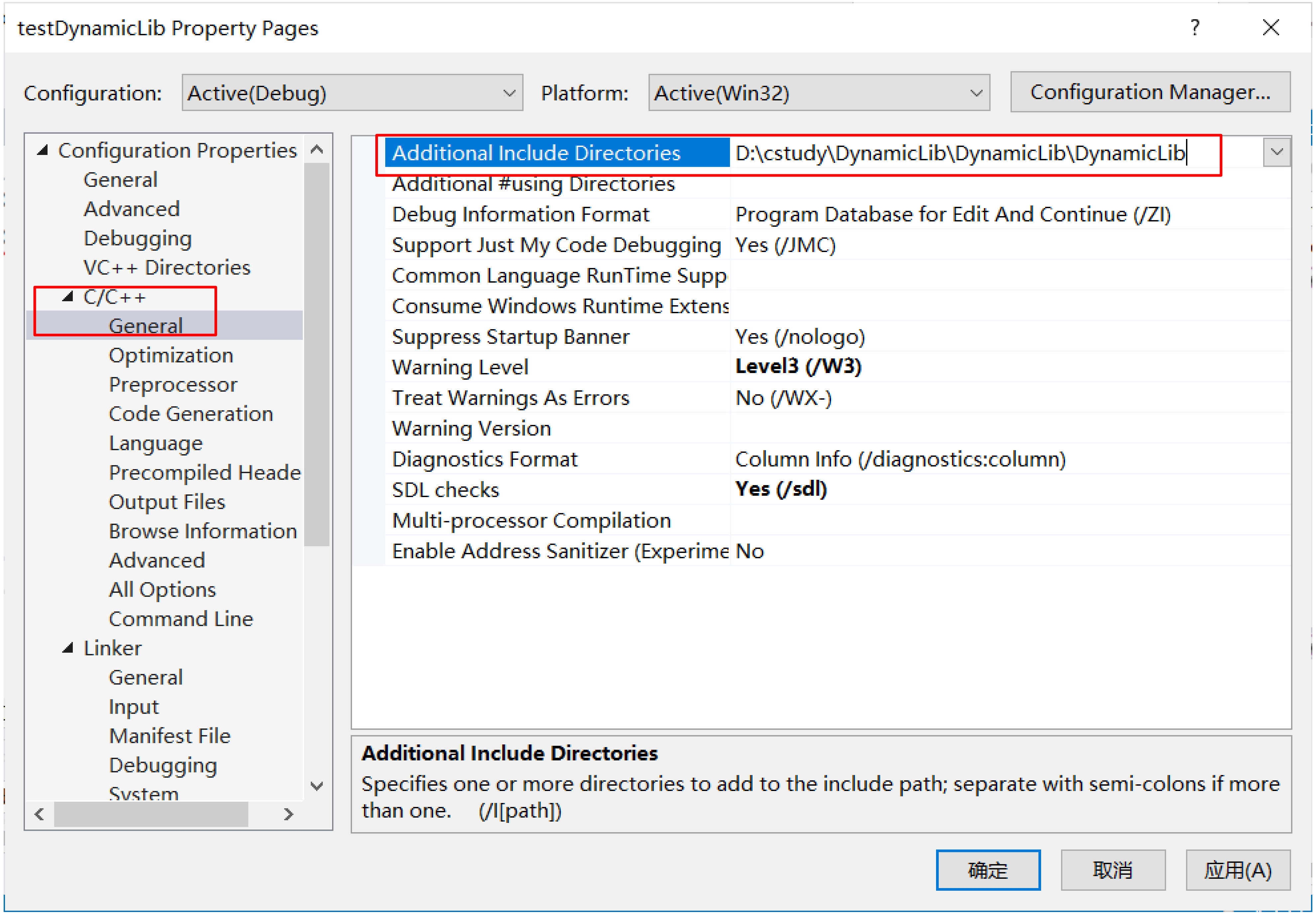
Configuration: Active(Debug)

Platform: Active(Win32)

Additional Dependencies: DynamicLib.lib;%(AdditionalDependencies)

Additional Dependencies

Specifies additional items to add to the link command line. [i.e. kernel32.lib]



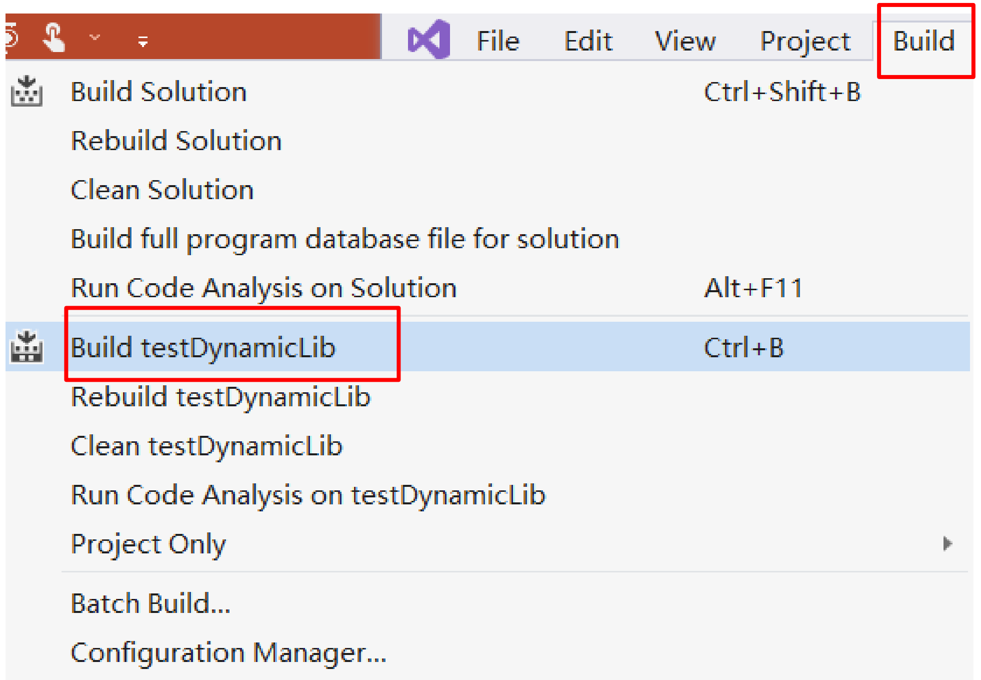
testDynamicLib Property Pages

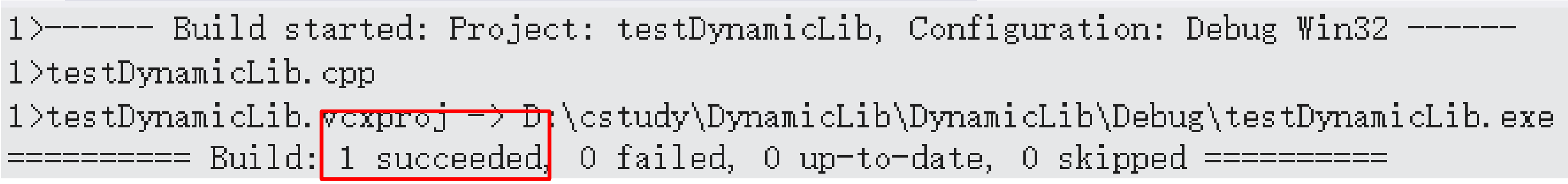
Configuration: Active(Debug)　Platform: Active(Win32)　Configuration Manager...

Additional Include Directories	D:\cstudy\DynamicLib\DynamicLib\DynamicLib
Additional #using Directories	
Debug Information Format	Program Database for Edit And Continue (/ZI)
Support Just My Code Debugging	Yes (/JMC)
Common Language RunTime Supp	
Consume Windows Runtime Extens	
Suppress Startup Banner	Yes (/nologo)
Warning Level	**Level3 (/W3)**
Treat Warnings As Errors	No (/WX-)
Warning Version	
Diagnostics Format	Column Info (/diagnostics:column)
SDL checks	**Yes (/sdl)**
Multi-processor Compilation	
Enable Address Sanitizer (Experime	No

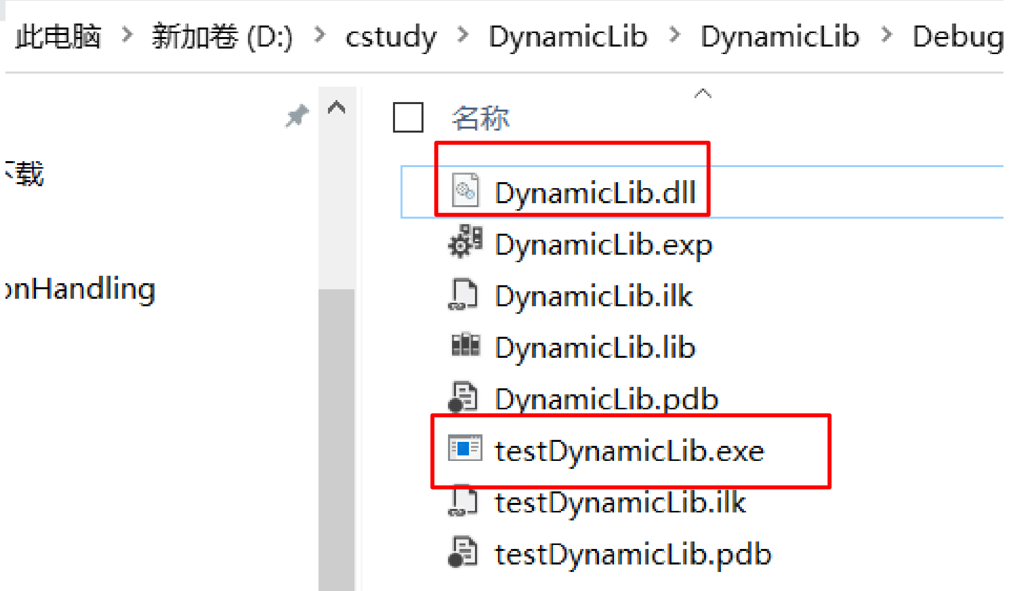
Additional Include Directories

Specifies one or more directories to add to the include path; separate with semi-colons if more than one. (/I[path])

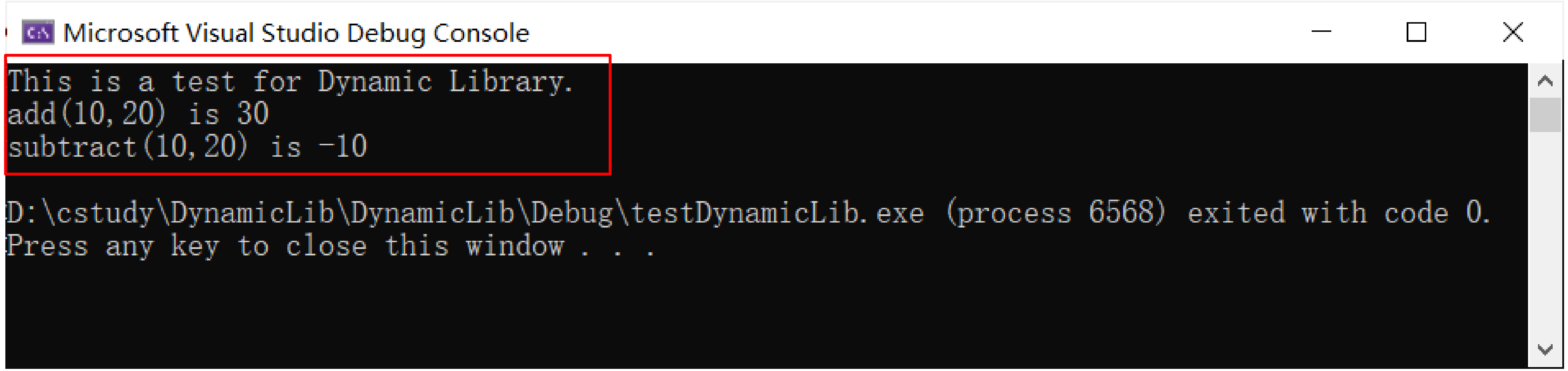

File
Edit
View
Project
Build
Build Solution Ctrl+Shift+B
Rebuild Solution
Clean Solution
Build full program database file for solution
Run Code Analysis on Solution Alt+F11
Build testDynamicLib Ctrl+B
Rebuild testDynamicLib
Clean testDynamicLib
Run Code Analysis on testDynamicLib
Project Only
Batch Build...
Configuration Manager...



```
1>------ Build started: Project: testDynamicLib, Configuration: Debug Win32 ------
1>testDynamicLib.cpp
1>testDynamicLib.vcxproj -> D:\cstudy\DynamicLib\DynamicLib\Debug\testDynamicLib.exe
========== Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped ==========
```


此电脑 › 新加卷 (D:) › cstudy › DynamicLib › DynamicLib › Debug
载
onHandling
名称

DynamicLib.dll
DynamicLib.exp
DynamicLib.ilk
DynamicLib.lib
DynamicLib.pdb
testDynamicLib.exe
testDynamicLib.ilk
testDynamicLib.pdb

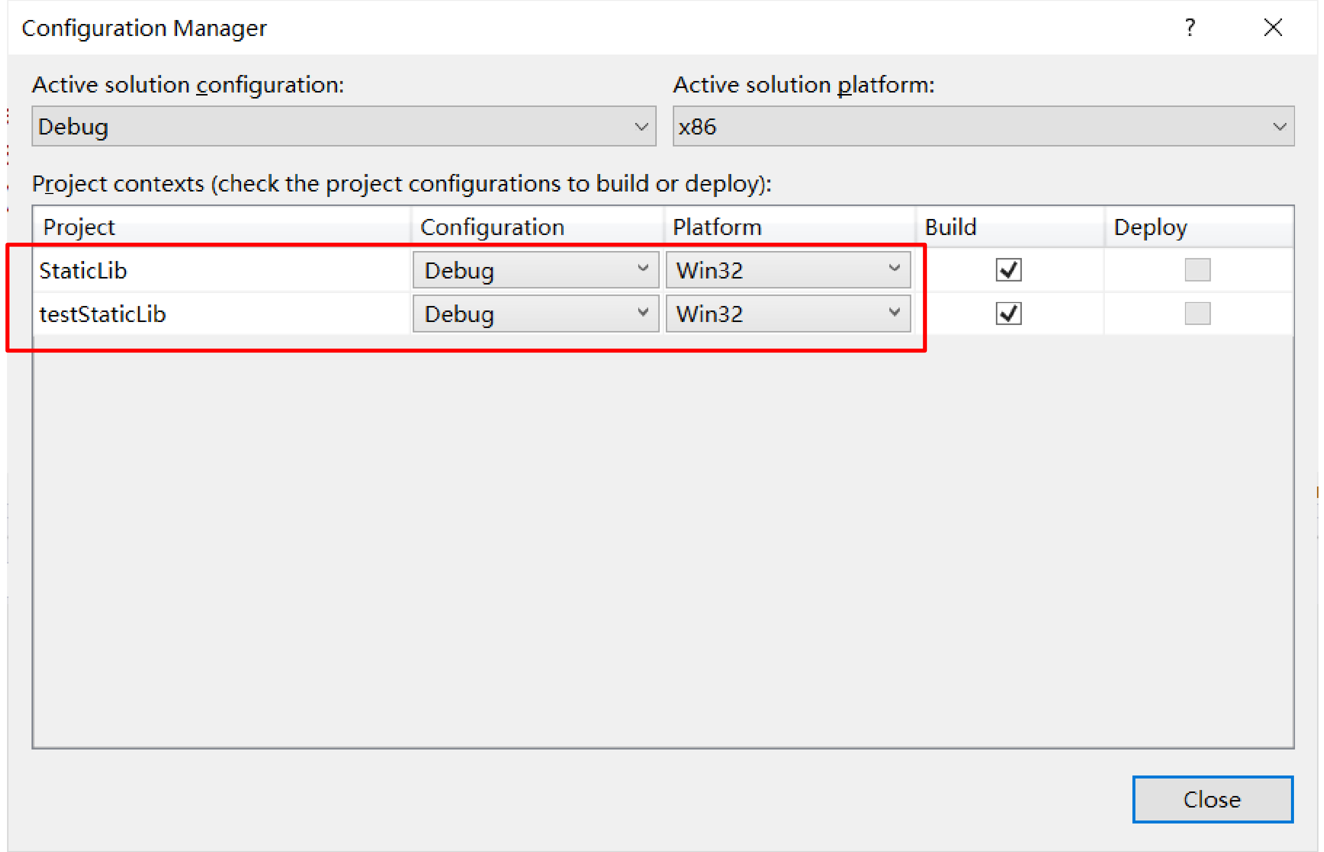


```
This is a test for Dynamic Library.
add(10,20) is 30
subtract(10,20) is -10

D:\cstudy\DynamicLib\DynamicLib\Debug\testDynamicLib.exe (process 6568) exited with code 0.
Press any key to close this window . . .
```

Create static and dynamic library in Linux
https://blog.csdn.net/kai_zone/article/details/93907101

NOTE:

The configuration and platform of both library and testproject must be matched.



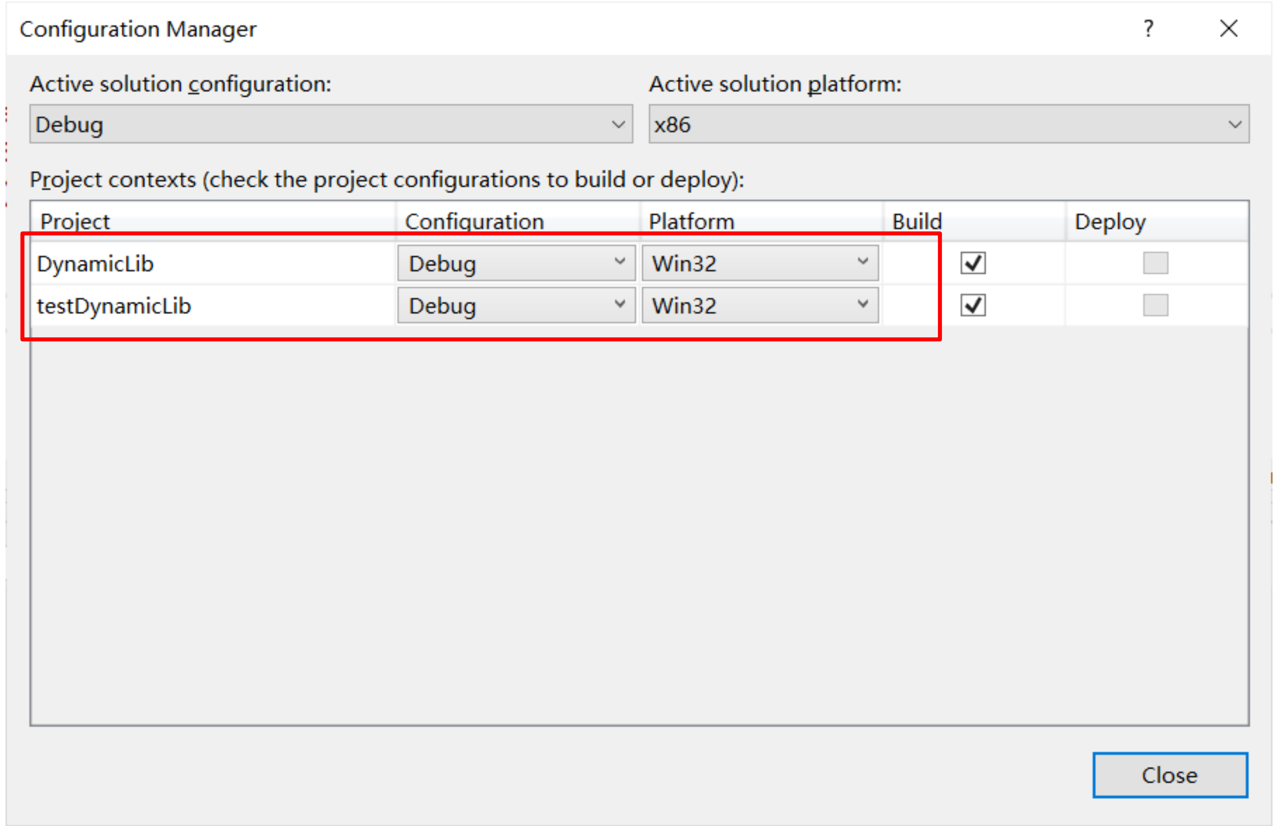
Configuration Manager

Active solution configuration: Debug

Active solution platform: x86

Project contexts (check the project configurations to build or deploy):

Project	Configuration	Platform	Build	Deploy
StaticLib	Debug	Win32	☑	□
testStaticLib	Debug	Win32	☑	□



Configuration Manager

Active solution configuration: Debug

Active solution platform: x86

Project contexts (check the project configurations to build or deploy):

Project	Configuration	Platform	Build	Deploy
DynamicLib	Debug	Win32	☑	□
testDynamicLib	Debug	Win32	☑	□