Fundamentals of Computer Graphics

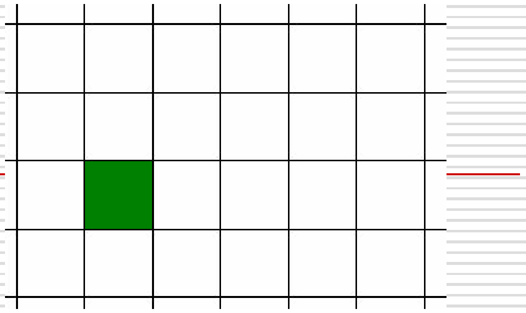
Lecture 3. My first OpenGL program

Yong-Jin Liu
liuyongjin@tsinghua.edu.cn

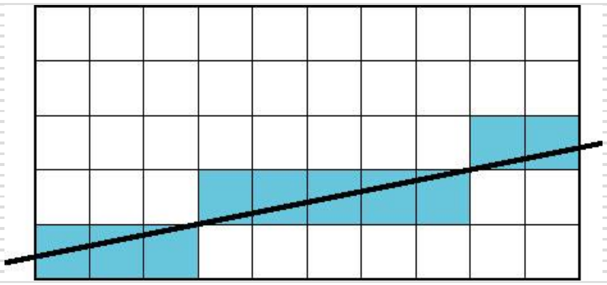
Outline

- OpenGL:
  draw an elemental shape
- Event-driven programming
- Use GLUT advanced library
- The first OpenGL program

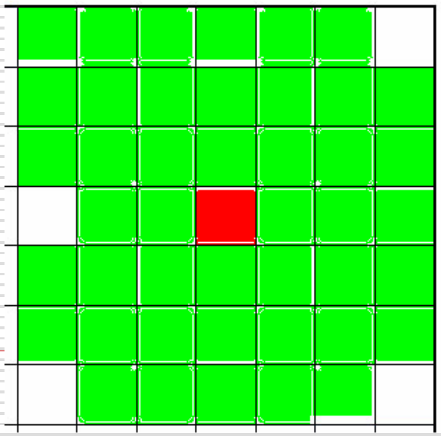
Begin OpenGL



Draw point, line, face:

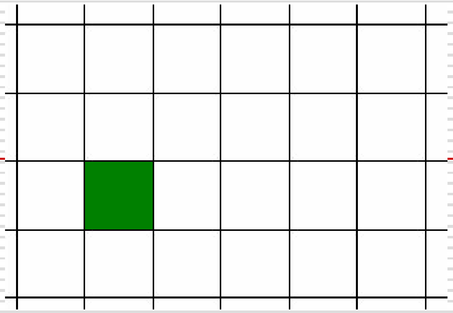


```
glBegin(parameter);

  … …

glEnd();
```



parameter: GL_POINTS, GL_LINES,
GL_POLYGON, GL_TRIANGLES

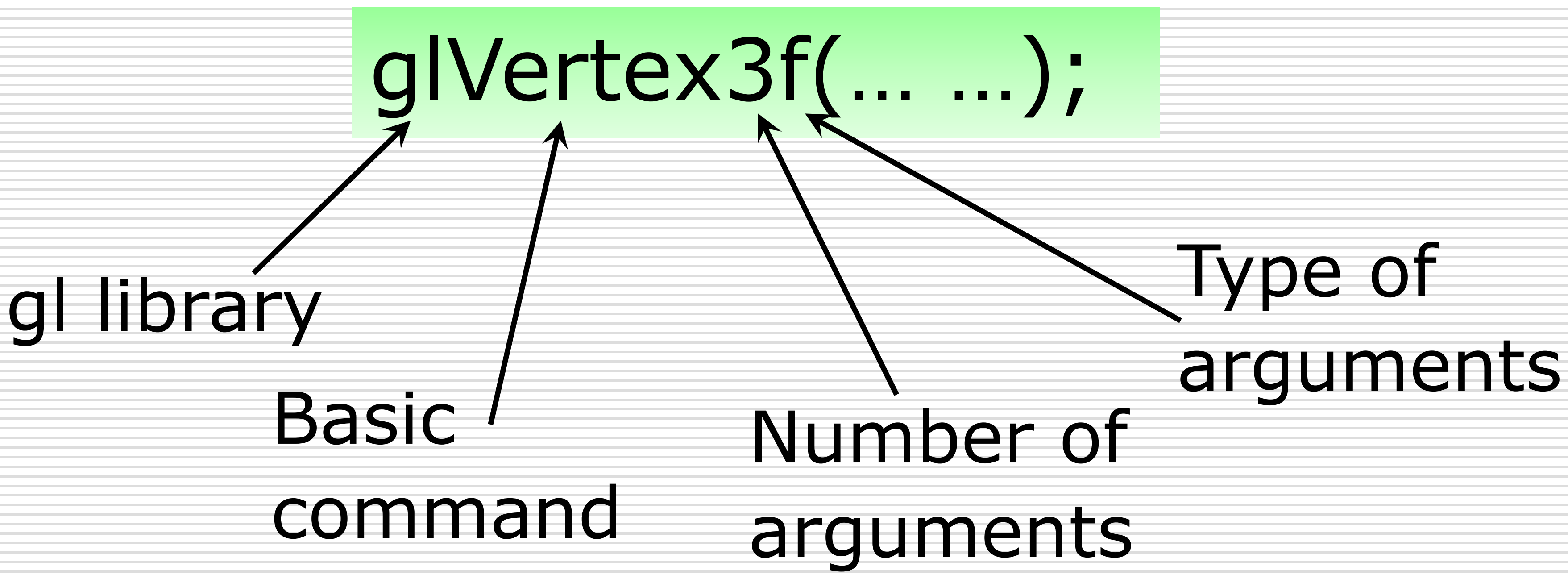
Begin OpenGL



Draw points:

```
glBegin(GL_POINTS);
  glVertex3f(-0.5,-0.5,0.0);
  glVertex3f(0.5,0.0,0.0);
  glVertex3f(0.0,0.5,0.0);
  … …
glEnd();
```

GL Function Construction



Example of construction

- glVertex2i(⋯) takes integer values
- glVertex2d(⋯) takes floating point values

- OpenGL has its own data types to make graphics device-independent
  Use these types instead of standard ones

Open-GL Data Types

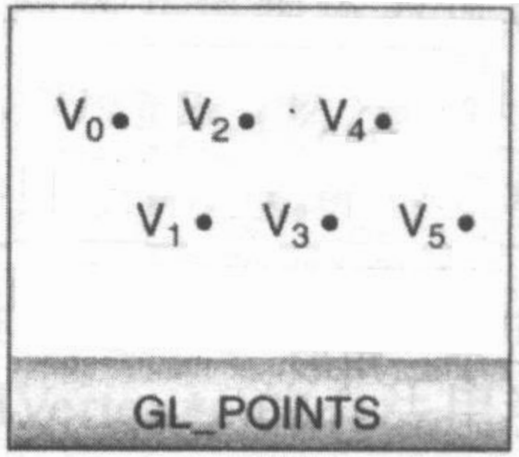
suffix	data type	C/C++ type	OpenGL type name
b	**8-bit integer**	**signed char**	**GLbyte**
s	**16-bit integer**	**Short**	**GLshort**
i	**32-bit integer**	**int or long**	**GLint, GLsizei**
f	**32-bit float**	**Float**	**GLfloat, GLclampf**
d	**64-bit float**	**Double**	**GLdouble,GLclampd**
ub	**8-bit unsigned number**	**unsigned char**	**GLubyte,GLboolean**
us	**16-bit unsigned number**	**unsigned short**	**GLushort**
ui	**32-bit unsigned number**	**unsigned int or unsigned long**	**GLuint,Glenum,GLbitfield**

Begin OpenGL

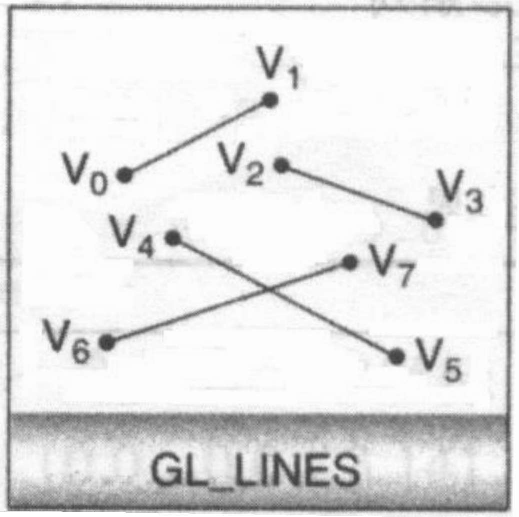
Draw all lines at one time:

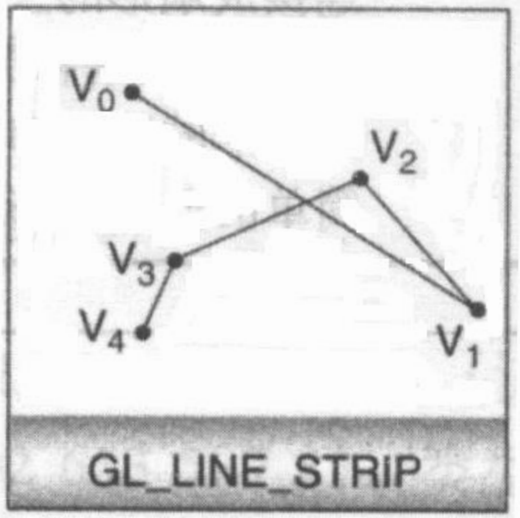
```
glBegin(GL_LINES);
  glVertex3f(-0.5,-0.5,0.0);  }
  glVertex3f(0.5,0.0,0.0);    }
  glVertex3f(0.0,0.5,0.0);    }
  glVertex3f(0.0,0.0,0.5);    }
  … …
glEnd();
```

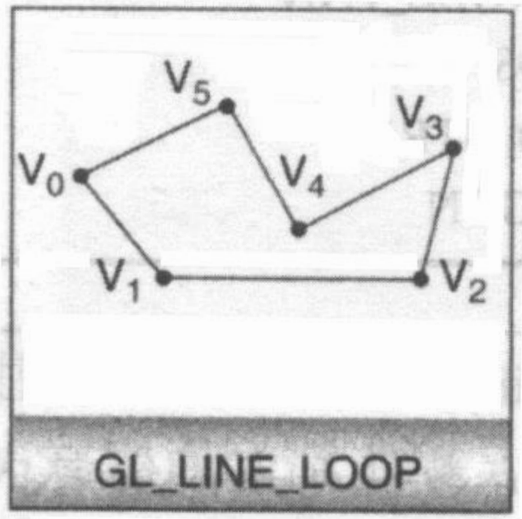
Begin OpenGL



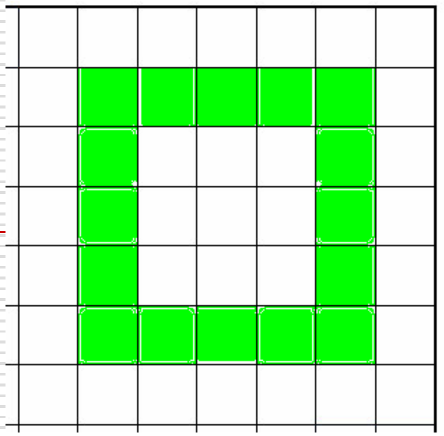
OpenGL programming guide, 8th edtion







Begin OpenGL



Draw polygon:

```
glBegin(GL_POLYGON);
  glVertex3f(-0.5,-0.5,0.0);
  glVertex3f(0.5,0.0,0.0);
  glVertex3f(0.0,0.5,0.0);
  glVertex3f(0.0,0.0,0.5);
  … …
glEnd();
```

Must be convex polygon

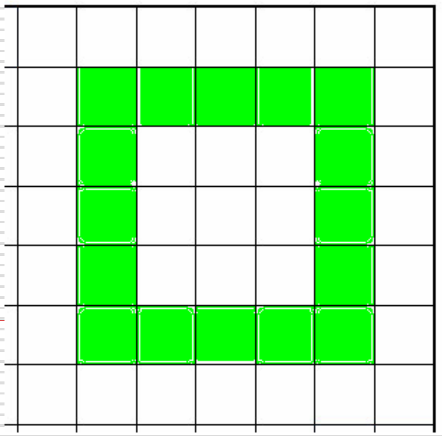
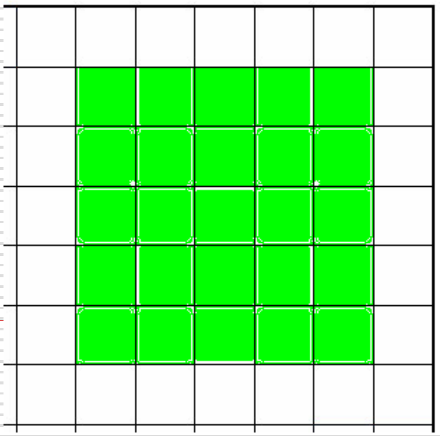
Begin OpenGL

Draw polygon - Region filling:

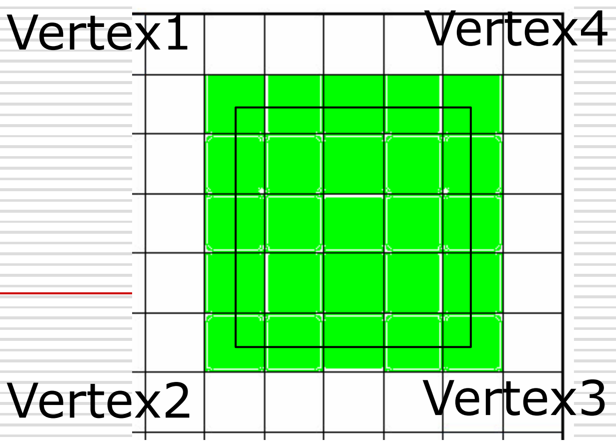
```
glPolygonMode(parameter1, parameter2);
```

parameter2: GL_LINE, GL_FILL

parameter1: GL_FRONT, GL_BACK

Begin OpenGL



Draw polygon:

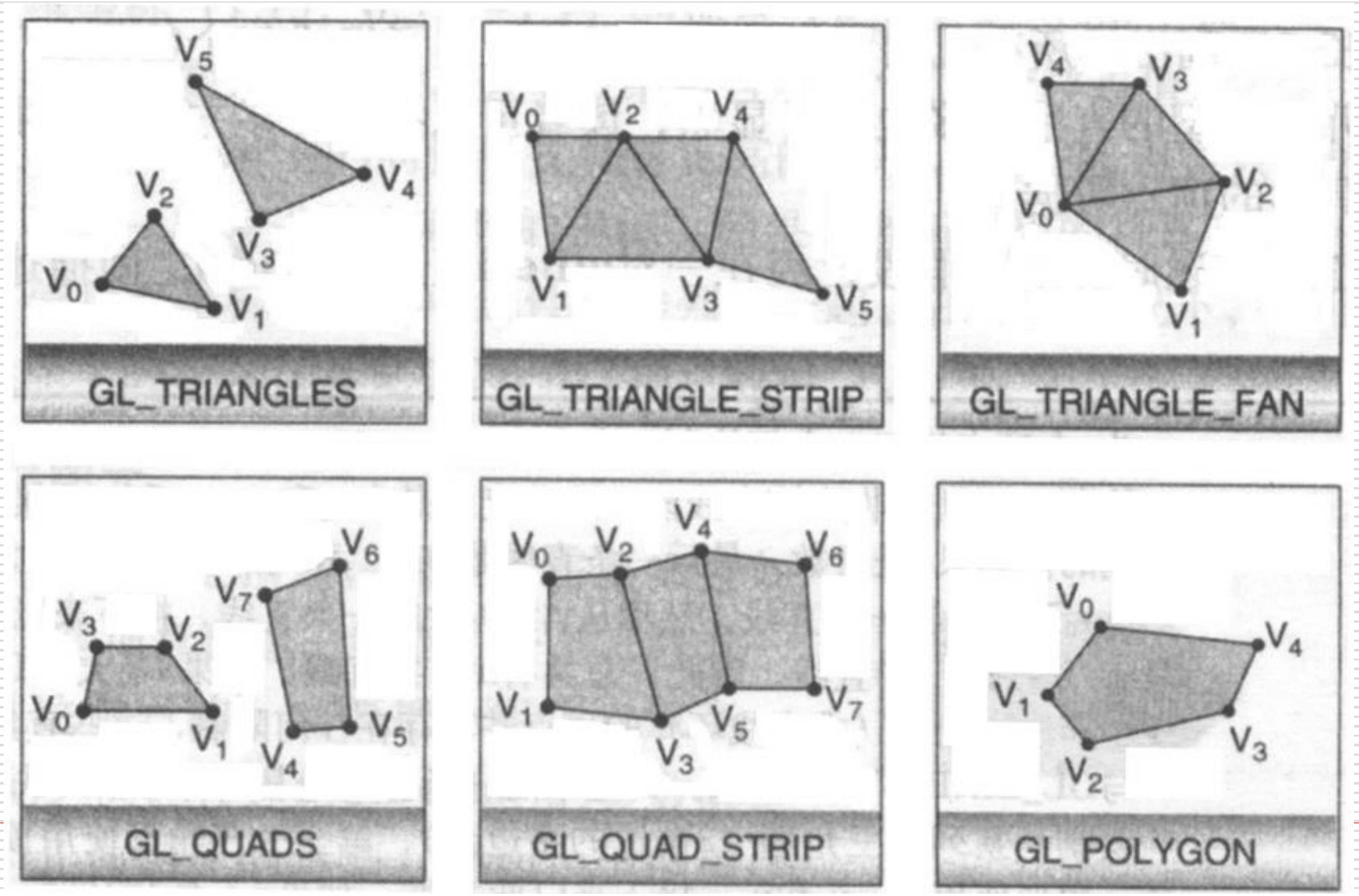
```
glPolygonMode(GL_FRONT, GL_FILL);
glBegin(GL_POLYGON);
  glVertex3f(coordinate of vertex1);
  glVertex3f(coordinate of vertex2);
  glVertex3f(coordinate of vertex3);
  glVertex3f(coordinate of vertex4);
  … …
glEnd();
```

Begin OpenGL

Draw triangle:

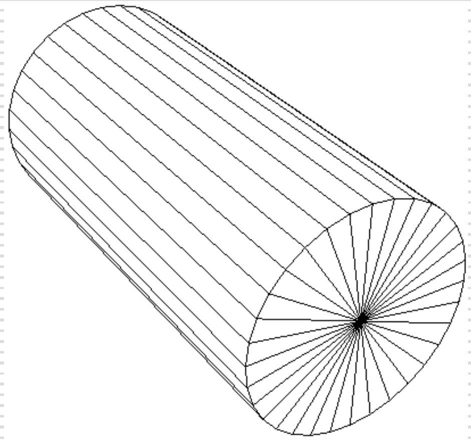
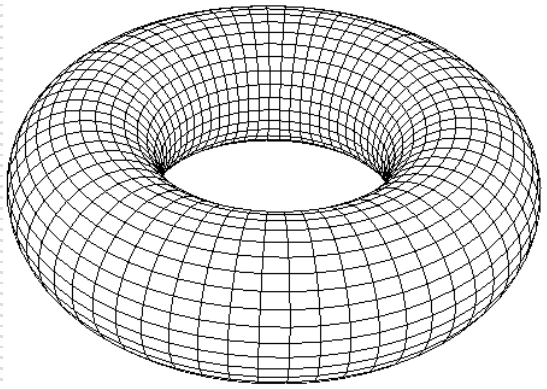
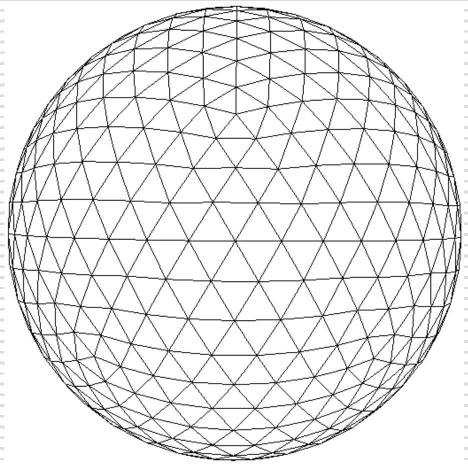
```
glPolygonMode(GL_FRONT, GL_FILL);
glBegin(GL_TRIANGLES);
  glVertex3f(coordinate of vertex1);
  glVertex3f(coordinate of vertex2);
  glVertex3f(coordinate of vertex3);
  … …
  … …
glEnd();
```

Begin OpenGL



Begin OpenGL

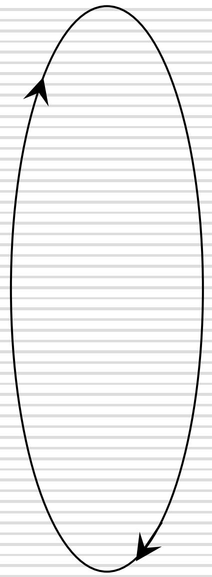
Representing 3D shape using triangle/quad meshes

The first OpenGL program

Event and message driven mechanism

- When program is running,
  Enter into a waiting status
  "Do nothing"

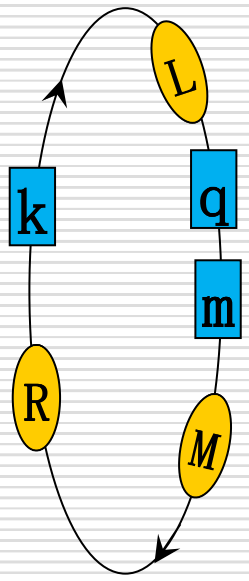


The first OpenGL program

- Event occurs

  press mouse、 press key on keyboard, resize the window size, etc.
- Generate special event message
- The System manages the event queue automatically

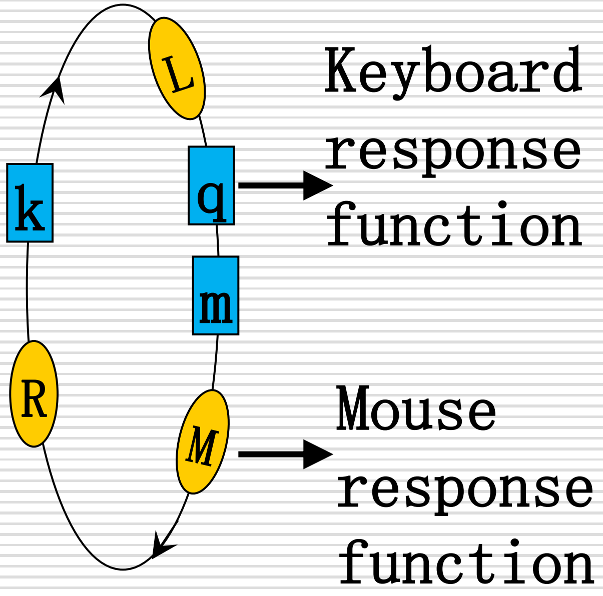
  First come first serve



The first OpenGL program

write event handler in program

Program registers events handler into operating system



Event happens → The system calls event handler → Remove the event from event queue → Restore waiting status

Use GLUT library!

Event-driven programs

- Respond to events, such as mouse click or move, key press, or window reshape or resize. System manages event queue
- Programmer provides "call-back" functions to handle each event
- Call-back functions must be registered with OpenGL to let it know which function handles which event
- Registering function does *not* call it!

OpenGL is a state-based API

- Most OpenGL functions manipulate global state
- 3 types of functions
  - Those that modify global state
  - Those that query global state
  - Those that cause something to be rendered

    e.g., glEnd or glDrawElements

OpenGL pipeline

- Rendering is object-based
  - Vertices and fragments are processed in parallel, independently of each other
  - No global effect
- Basic process

OpenGL pipeline

- Basic process
  - Evaluation of uniform and attribute data
  - Processing of vertices
  - Assembly of vertices into primitives
  - Rasterization of primitives into fragments
  - Processing of fragments
  - Composition of fragments into the frame buffer

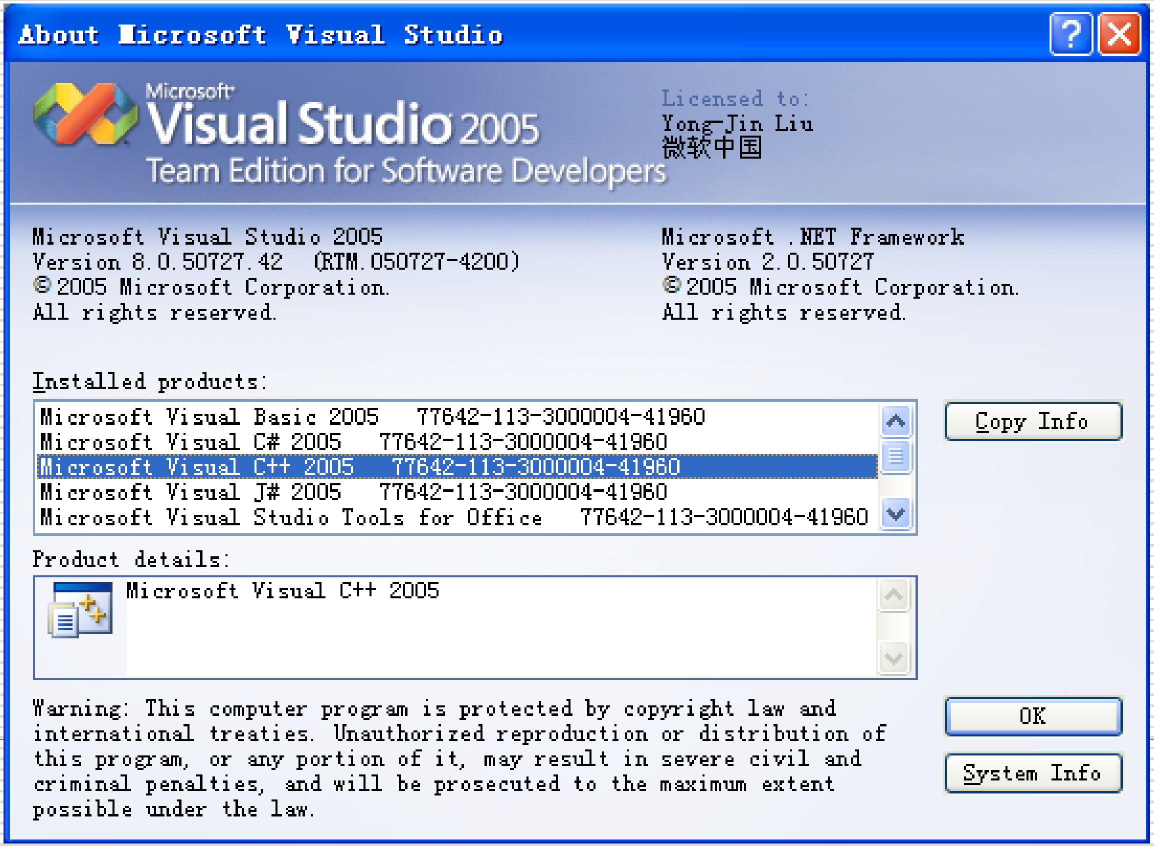
The first OpenGL program

First step：preprocessing

- opengl32.lib glut32.lib glu32.lib
- include head file <gl/glut.h> <gl/glu.h> <gl/gl.h>

The first OpenGL program

- Demo: Install and run Microsoft Visual Studio 2005 （Team Edition, Version 8.0）


About Microsoft Visual Studio
Microsoft Visual Studio 2005
Team Edition for Software Developers
Copy Info
OK
System Info

Licensed to:
Yong-Jin Liu
微软中国

Microsoft Visual Studio 2005
Version 8.0.50727.42 (RTM.050727-4200)
© 2005 Microsoft Corporation.
All rights reserved.

Microsoft .NET Framework
Version 2.0.50727
© 2005 Microsoft Corporation.
All rights reserved.

Installed products:

```
Microsoft Visual Basic 2005   77642-113-3000004-41960
Microsoft Visual C# 2005   77642-113-3000004-41960
Microsoft Visual C++ 2005   77642-113-3000004-41960
Microsoft Visual J# 2005   77642-113-3000004-41960
Microsoft Visual Studio Tools for Office   77642-113-3000004-41960
```

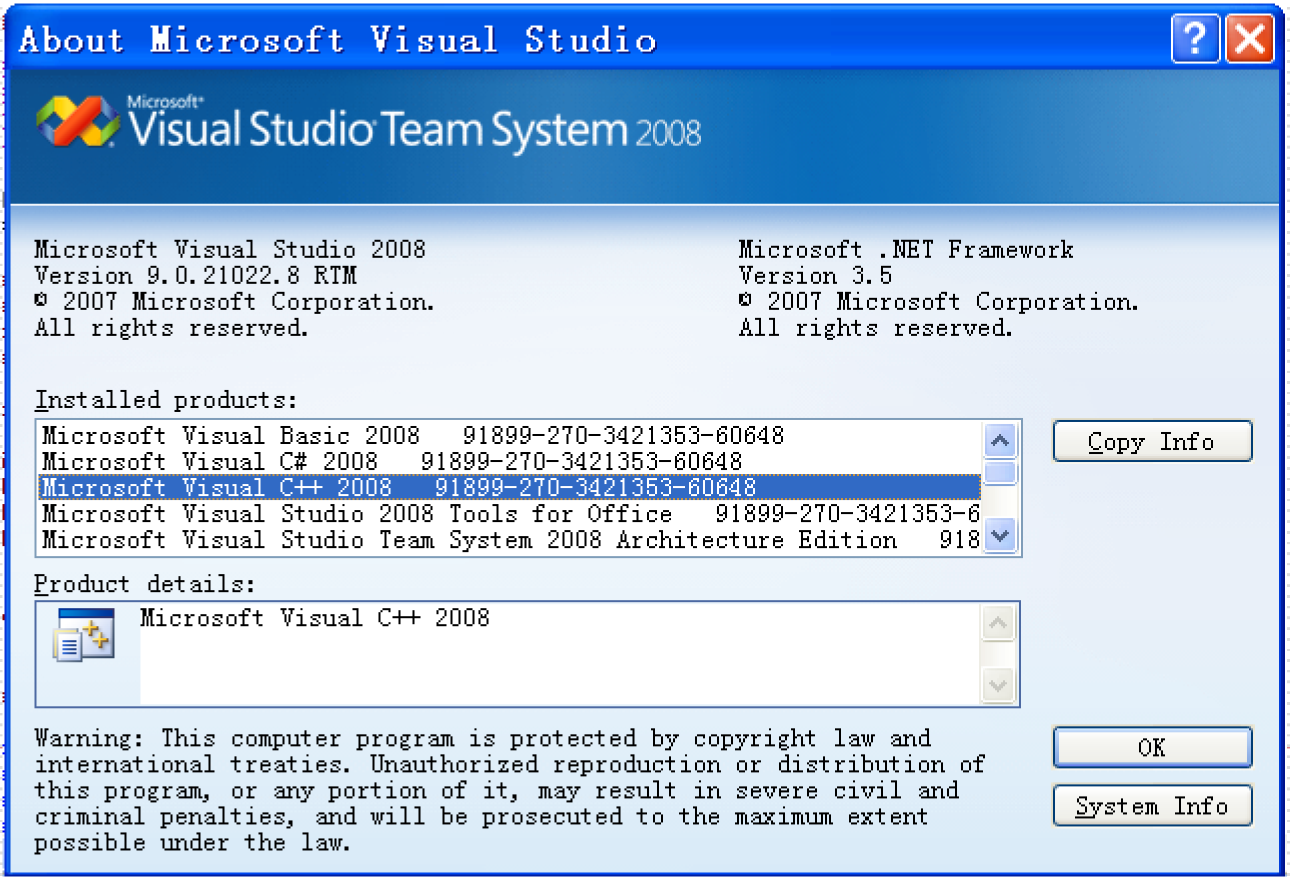
Product details:

Microsoft Visual C++ 2005

Warning: This computer program is protected by copyright law and international treaties. Unauthorized reproduction or distribution of this program, or any portion of it, may result in severe civil and criminal penalties, and will be prosecuted to the maximum extent possible under the law.

The first OpenGL program

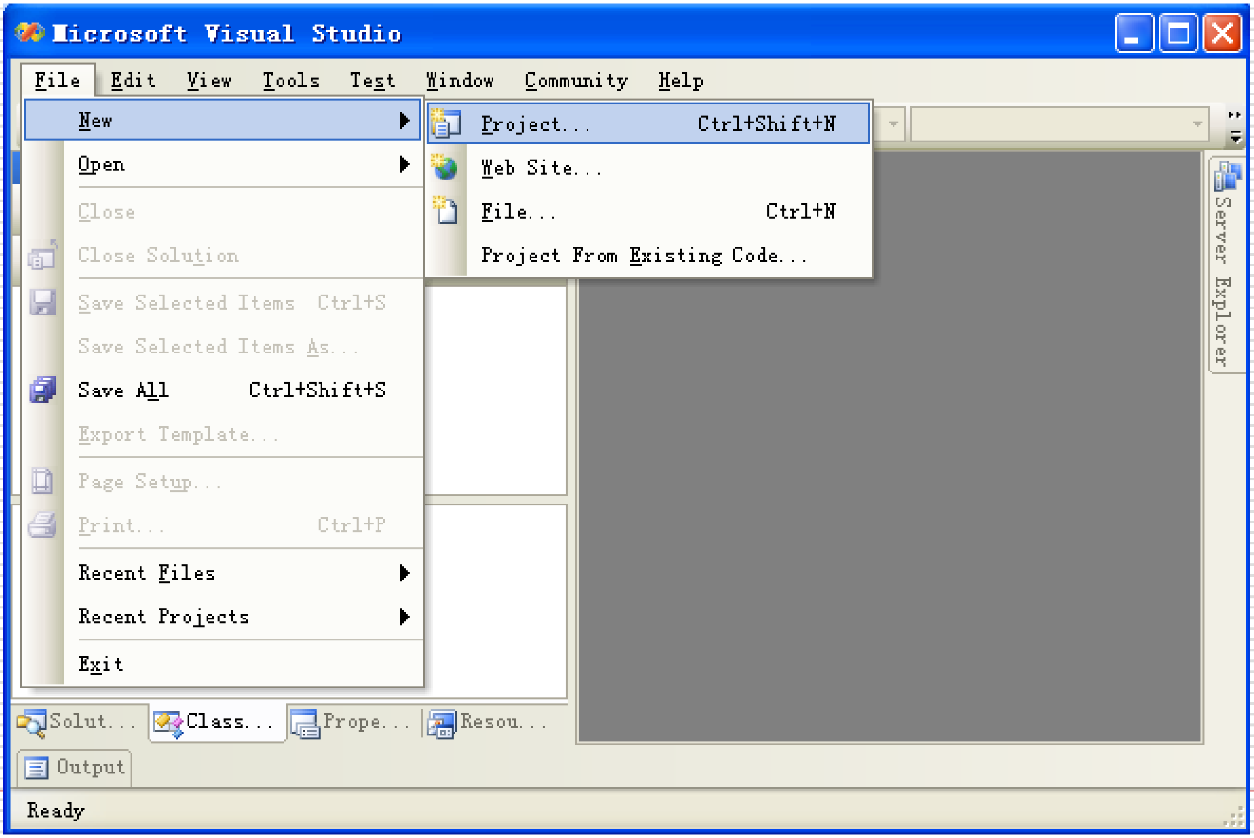
- Demo: Install and run Microsoft Visual Studio 2008 （Team Edition, Version 9.0）



```
Microsoft Visual Studio 2008                    Microsoft .NET Framework
Version 9.0.21022.8 RTM                         Version 3.5
© 2007 Microsoft Corporation.                   © 2007 Microsoft Corporation.
All rights reserved.                            All rights reserved.

Installed products:
Microsoft Visual Basic 2008    91899-270-3421353-60648
Microsoft Visual C# 2008    91899-270-3421353-60648
Microsoft Visual C++ 2008    91899-270-3421353-60648
Microsoft Visual Studio 2008 Tools for Office    91899-270-3421353-6
Microsoft Visual Studio Team System 2008 Architecture Edition    918

Product details:
Microsoft Visual C++ 2008

Warning: This computer program is protected by copyright law and
international treaties. Unauthorized reproduction or distribution of
this program, or any portion of it, may result in severe civil and
criminal penalties, and will be prosecuted to the maximum extent
possible under the law.
```

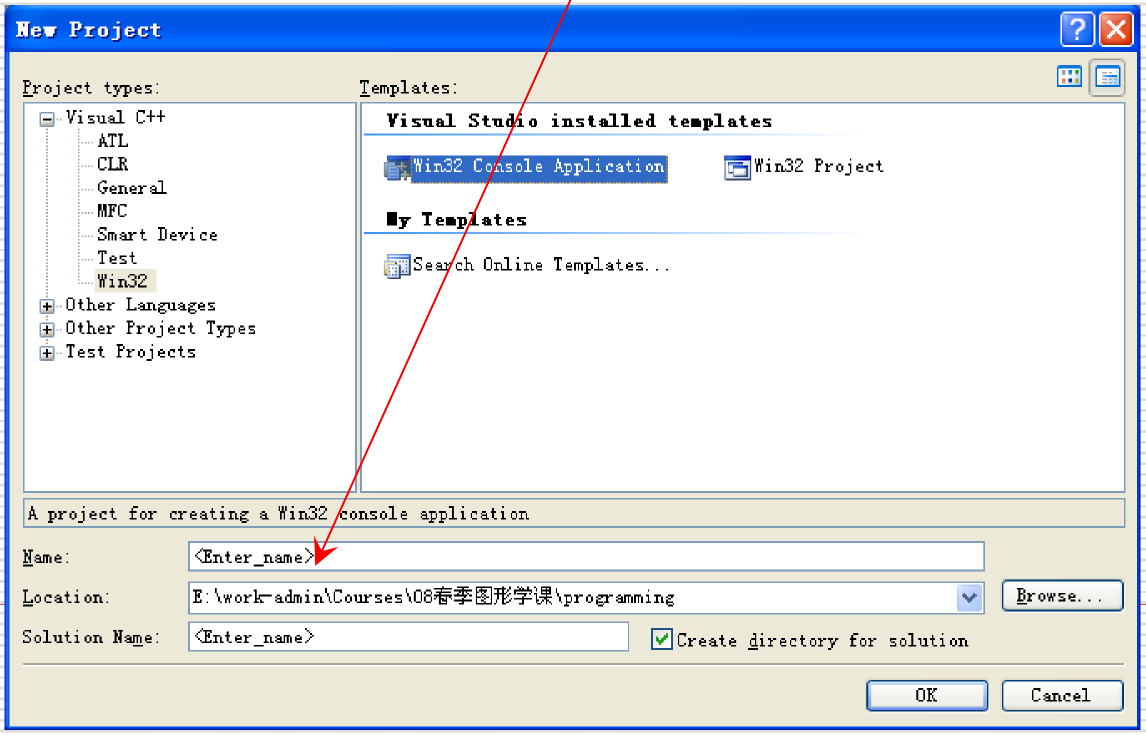
The first OpenGL program

- Demo: Select File → New → Project



The first OpenGL program

- Select Win32 and corresponding Win32 Console Application
- Input program name xxx, press OK, then a program with nothing is generated



A project for creating a Win32 console application

Name: <Enter_name>

Location: E:\work-admin\Courses\08春季图形学课\programming

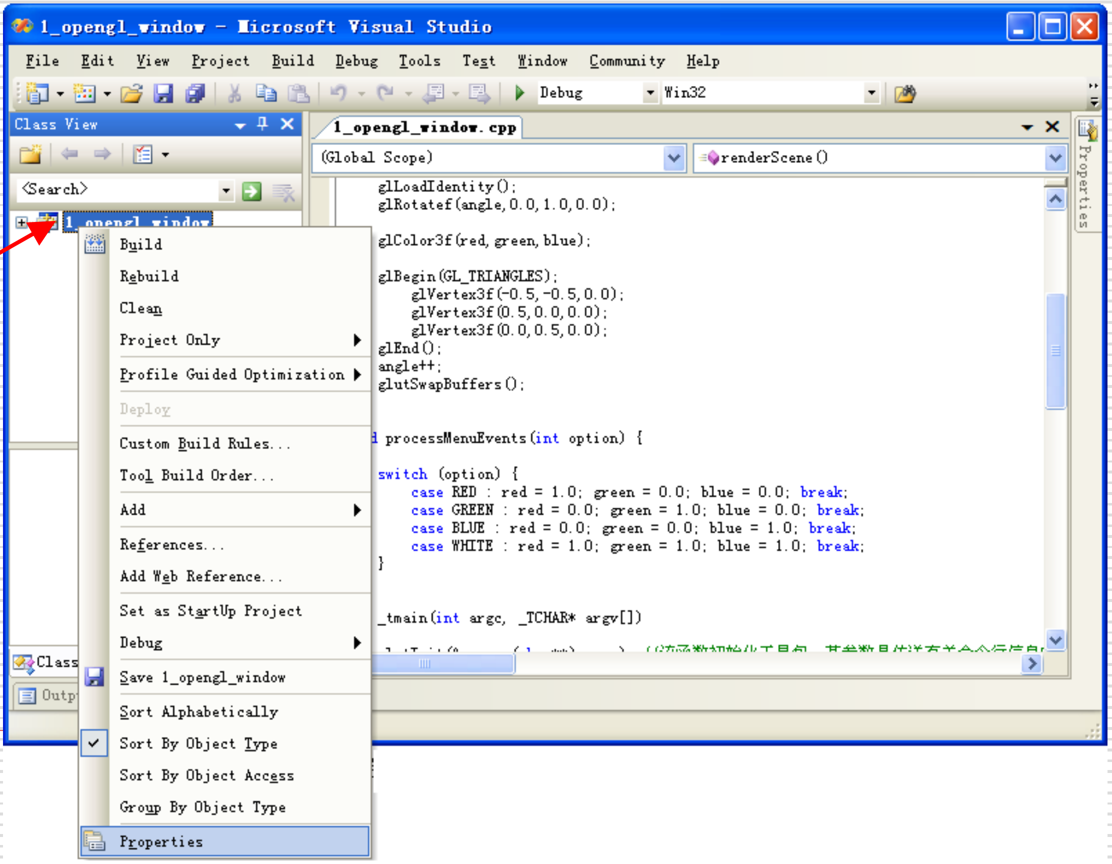
Solution Name: <Enter_name>

☑ Create directory for solution

The first OpenGL program

- Add lib in the project
  opengl32.lib glut32.lib glu32.lib

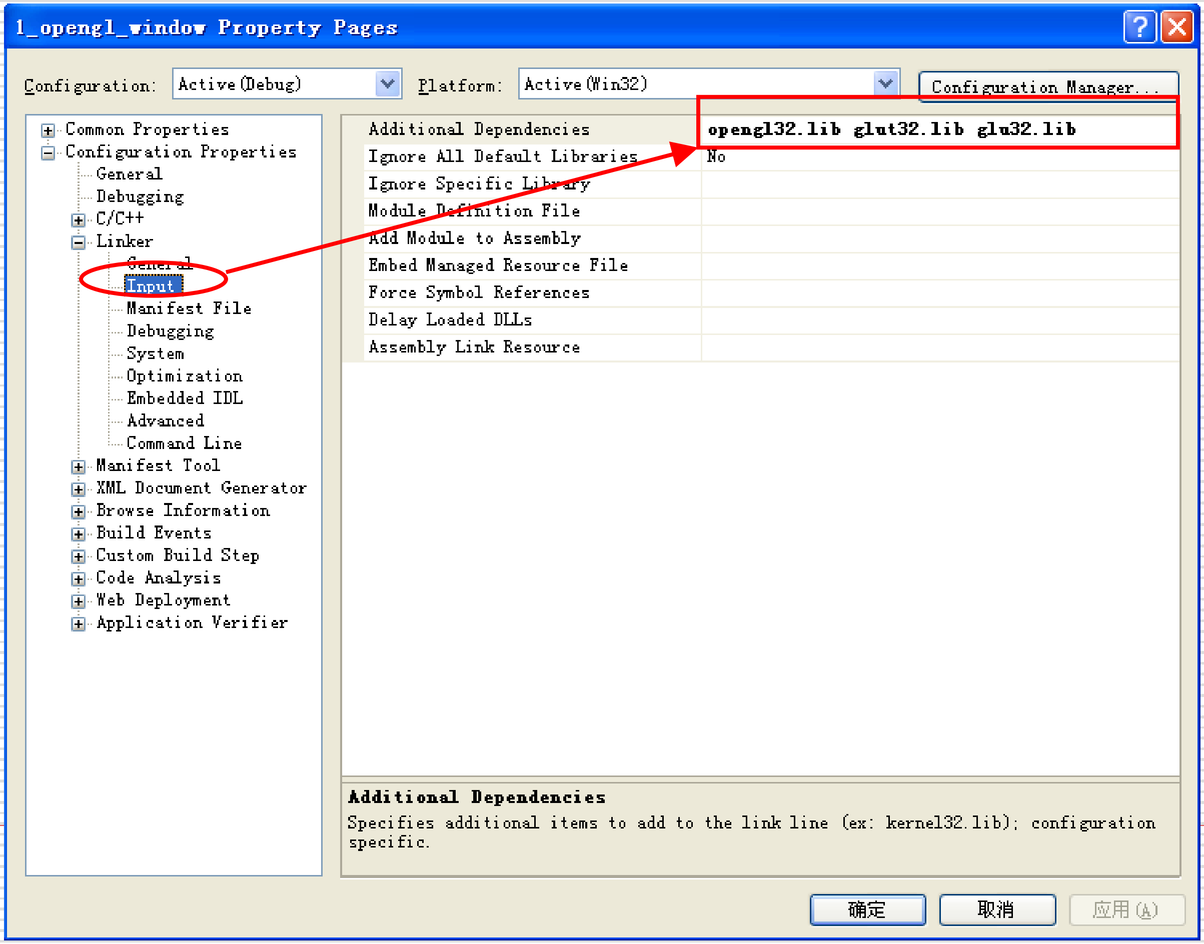
Click the right mouse button, select "Properties"



```
    glLoadIdentity();
    glRotatef(angle,0.0,1.0,0.0);

    glColor3f(red,green,blue);

    glBegin(GL_TRIANGLES);
        glVertex3f(-0.5,-0.5,0.0);
        glVertex3f(0.5,0.0,0.0);
        glVertex3f(0.0,0.5,0.0);
    glEnd();
    angle++;
    glutSwapBuffers();


d processMenuEvents(int option) {

    switch (option) {
        case RED : red = 1.0; green = 0.0; blue = 0.0; break;
        case GREEN : red = 0.0; green = 1.0; blue = 0.0; break;
        case BLUE : red = 0.0; green = 0.0; blue = 1.0; break;
        case WHITE : red = 1.0; green = 1.0; blue = 1.0; break;
    }


_tmain(int argc, _TCHAR* argv[])
```

The first OpenGL program

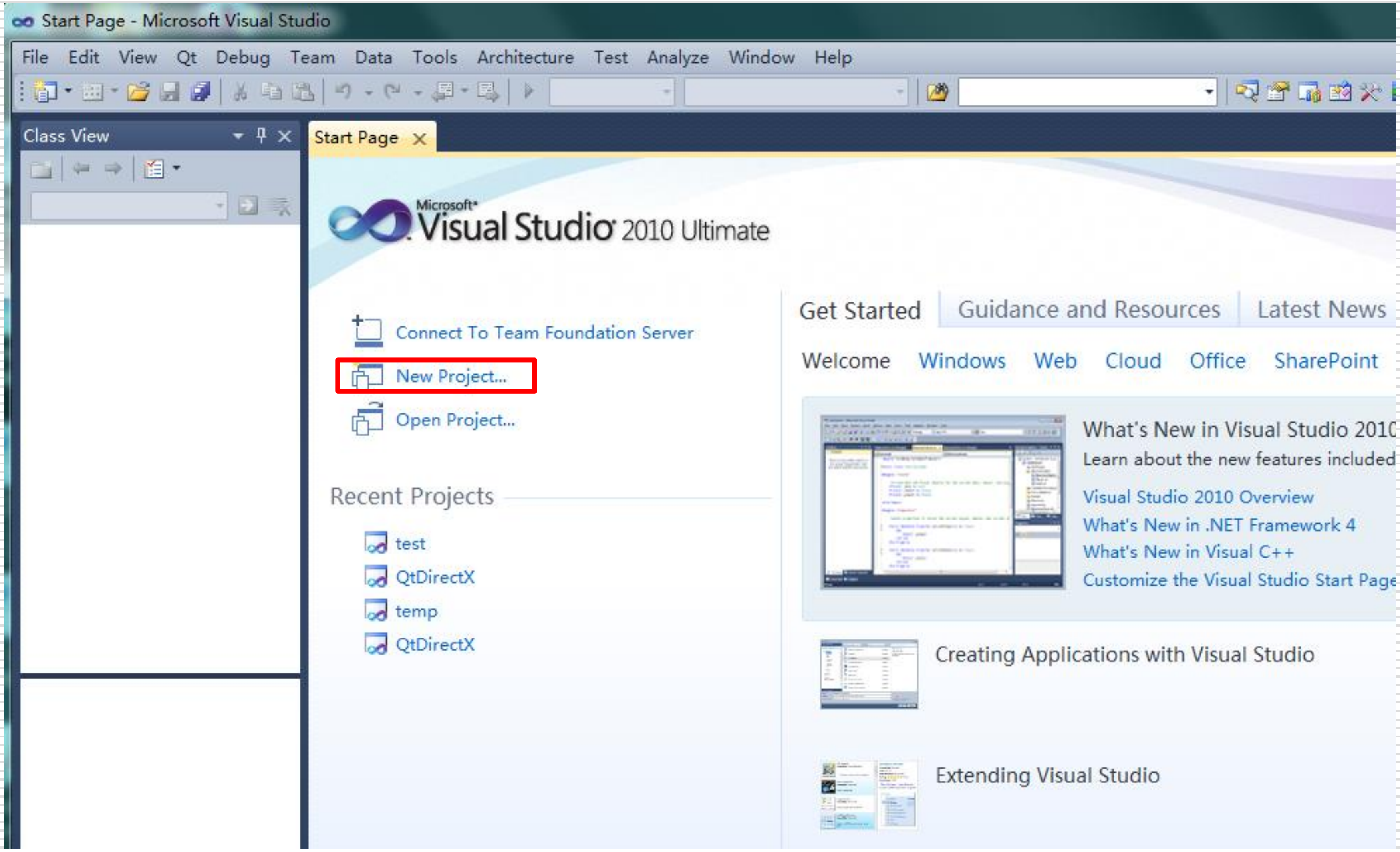
- In Configuration -> Linker -> Input

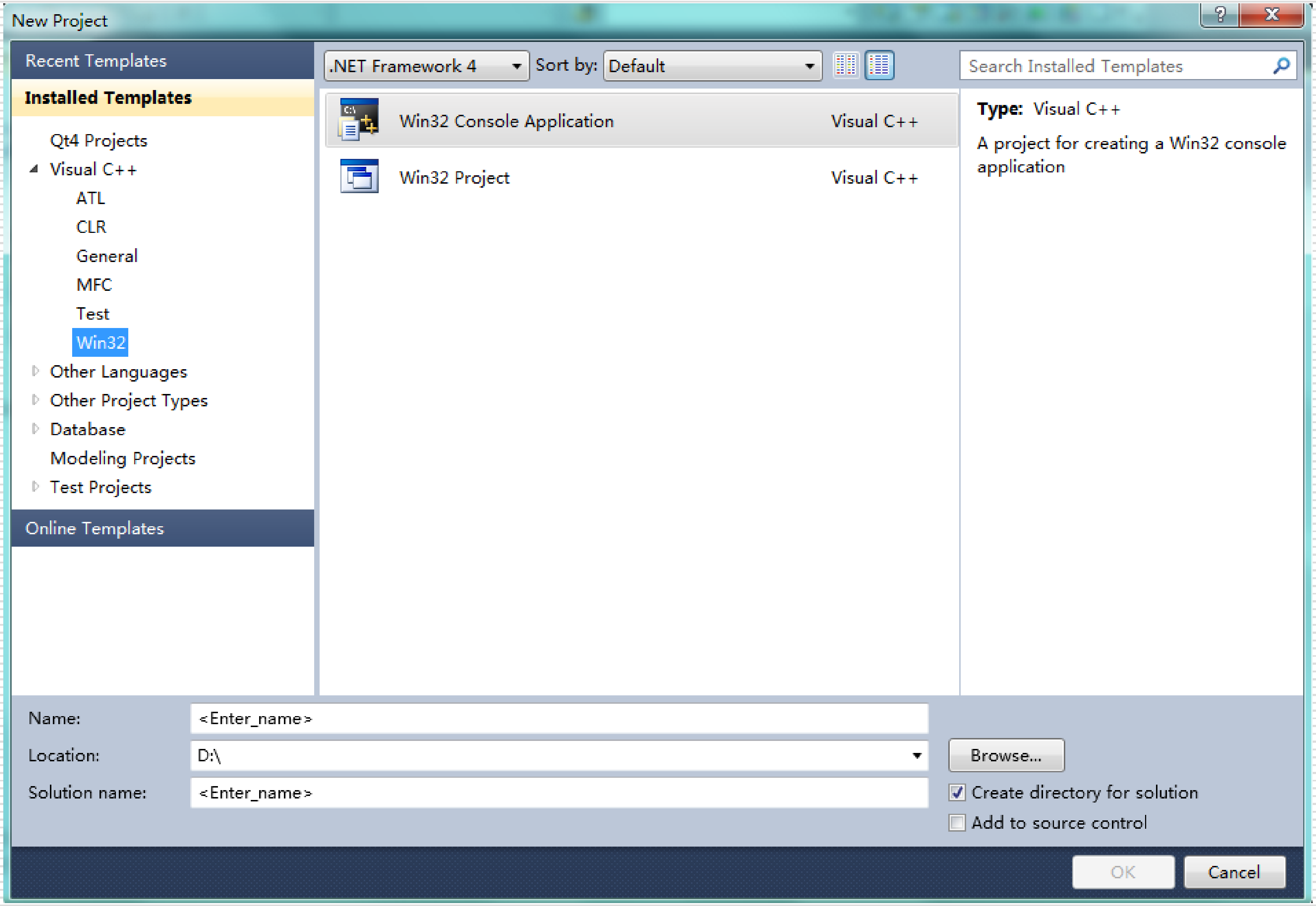


Additional Dependencies	opengl32.lib glut32.lib glu32.lib
Ignore All Default Libraries	No
Ignore Specific Library	
Module Definition File	
Add Module to Assembly	
Embed Managed Resource File	
Force Symbol References	
Delay Loaded DLLs	
Assembly Link Resource	

Additional Dependencies
Specifies additional items to add to the link line (ex: kernel32.lib); configuration specific.

Starting in VC++ 2010


Start Page - Microsoft Visual Studio
File Edit View Qt Debug Team Data Tools Architecture Test Analyze Window Help
Class View
Start Page
Microsoft Visual Studio 2010 Ultimate
Connect To Team Foundation Server
New Project...
Open Project...
Recent Projects
test
QtDirectX
temp
QtDirectX
Get Started
Guidance and Resources
Latest News
Welcome
Windows
Web
Cloud
Office
SharePoint
What's New in Visual Studio 2010
Learn about the new features included
Visual Studio 2010 Overview
What's New in .NET Framework 4
What's New in Visual C++
Customize the Visual Studio Start Page
Creating Applications with Visual Studio
Extending Visual Studio



Win32 Console Application	Visual C++
Win32 Project	Visual C++

Type: Visual C++

A project for creating a Win32 console application

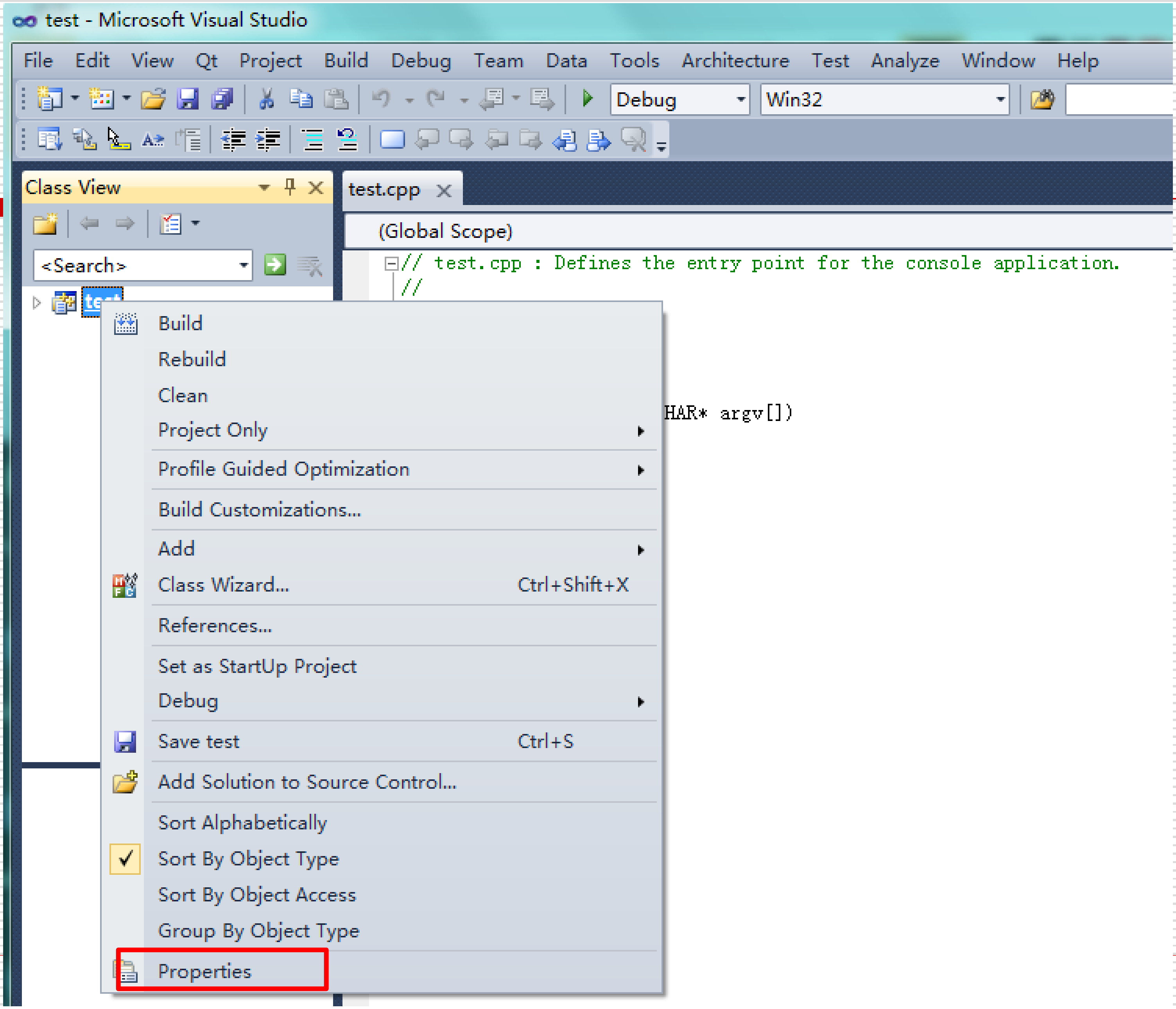
Name: <Enter_name>

Location: D:\

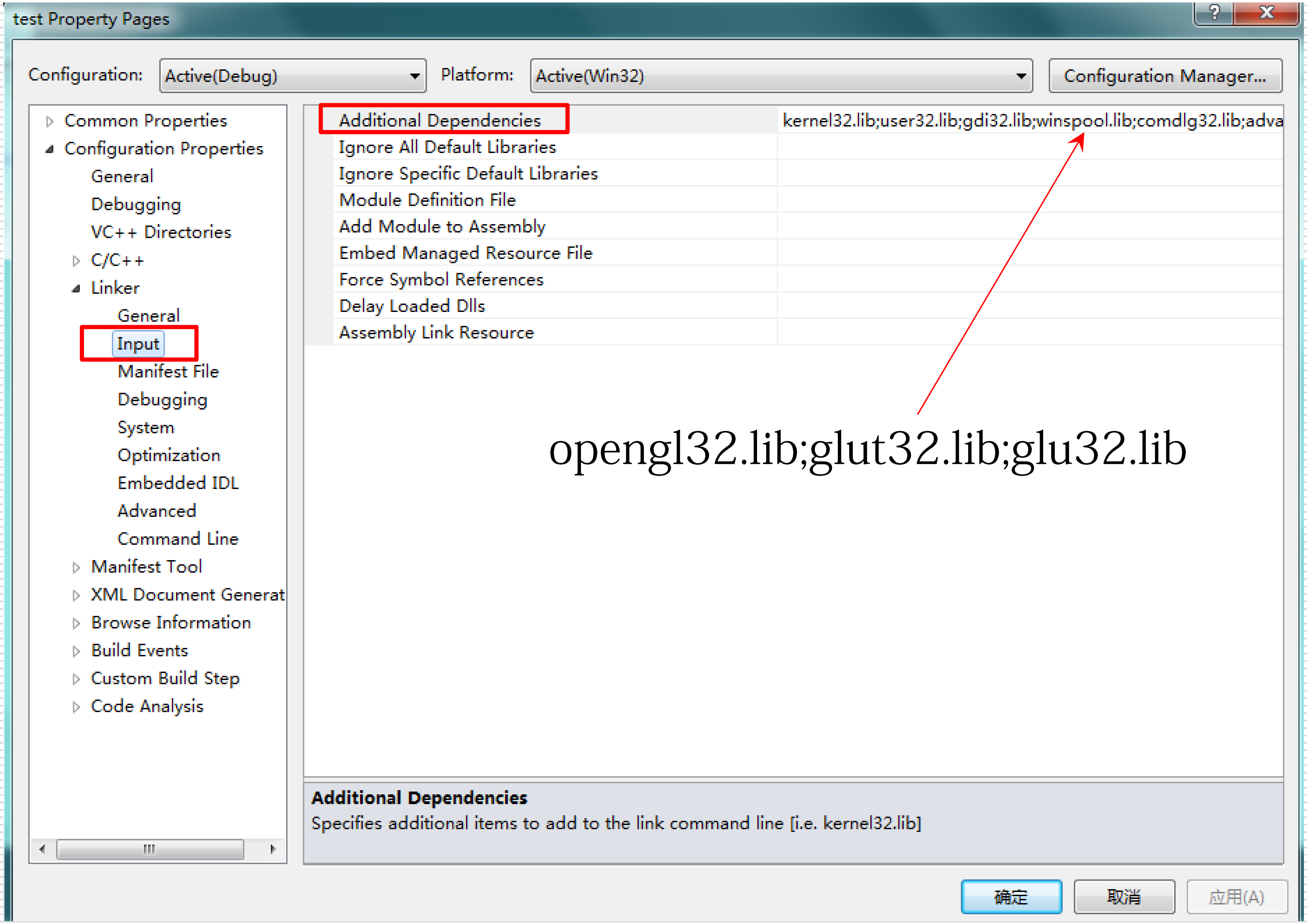
Solution name: <Enter_name>

[✓] Create directory for solution

[ ] Add to source control



```
// test.cpp : Defines the entry point for the console application.
//
HAR* argv[])
```



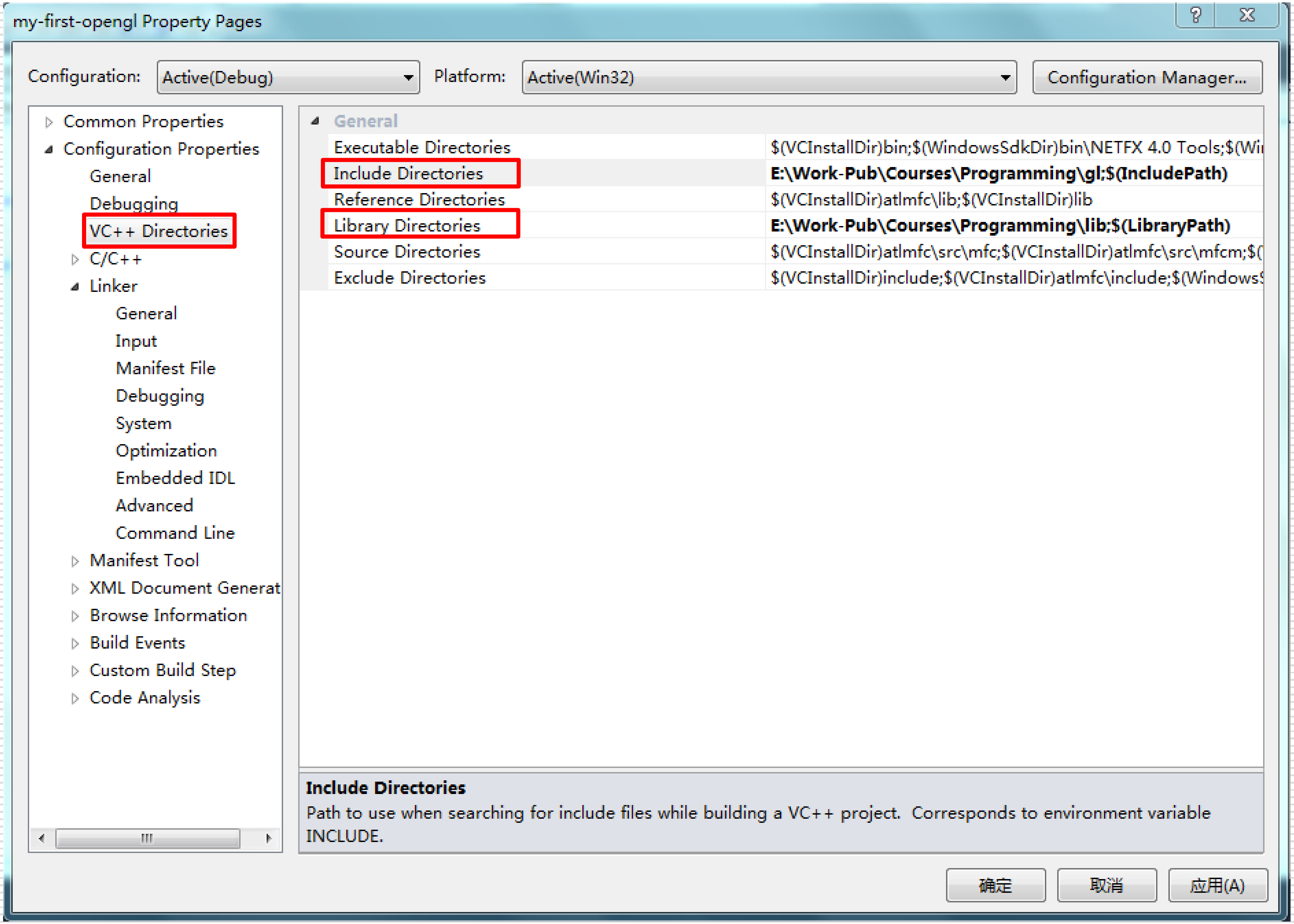
test Property Pages

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

Additional Dependencies	kernel32.lib;user32.lib;gdi32.lib;winspool.lib;comdlg32.lib;adva
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	

opengl32.lib;glut32.lib;glu32.lib

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



my-first-opengl Property Pages

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

General

Executable Directories	$(VCInstallDir)bin;$(WindowsSdkDir)bin\NETFX 4.0 Tools;$(Wi
Include Directories	**E:\Work-Pub\Courses\Programming\gl;$(IncludePath)**
Reference Directories	$(VCInstallDir)atlmfc\lib;$(VCInstallDir)lib
Library Directories	**E:\Work-Pub\Courses\Programming\lib;$(LibraryPath)**
Source Directories	$(VCInstallDir)atlmfc\src\mfc;$(VCInstallDir)atlmfc\src\mfcm;$(
Exclude Directories	$(VCInstallDir)include;$(VCInstallDir)atlmfc\include;$(Windows

Include Directories
Path to use when searching for include files while building a VC++ project. Corresponds to environment variable INCLUDE.

Libraries to Include

- GL for which the commands begin with GL
- GLUT, the GL Utility Toolkit, opens windows, develops menus, and manages events.
- GLU, the GL Utility Library, which provides high level routines to handle complex mathematical and drawing operations
- GLUI, the User Interface Library, which is completely integrated with the GLUT library

The first OpenGL program

- Add at the beginning of the program

```
#include<glut.h>
#include<glu.h>
#include<gl.h>
```

Then you can start:
A executable, simple OpenGL program！

The first OpenGL program

- main(): main function

```
int _tmain(int argc, _TCHAR* argv[])
{
        return 0;
 }
```

```
glutInit(&argc, (char**) argv);
//This funciton initializes toolkit,
//The parameters are about command line input,
//useless here
```

The first OpenGL program

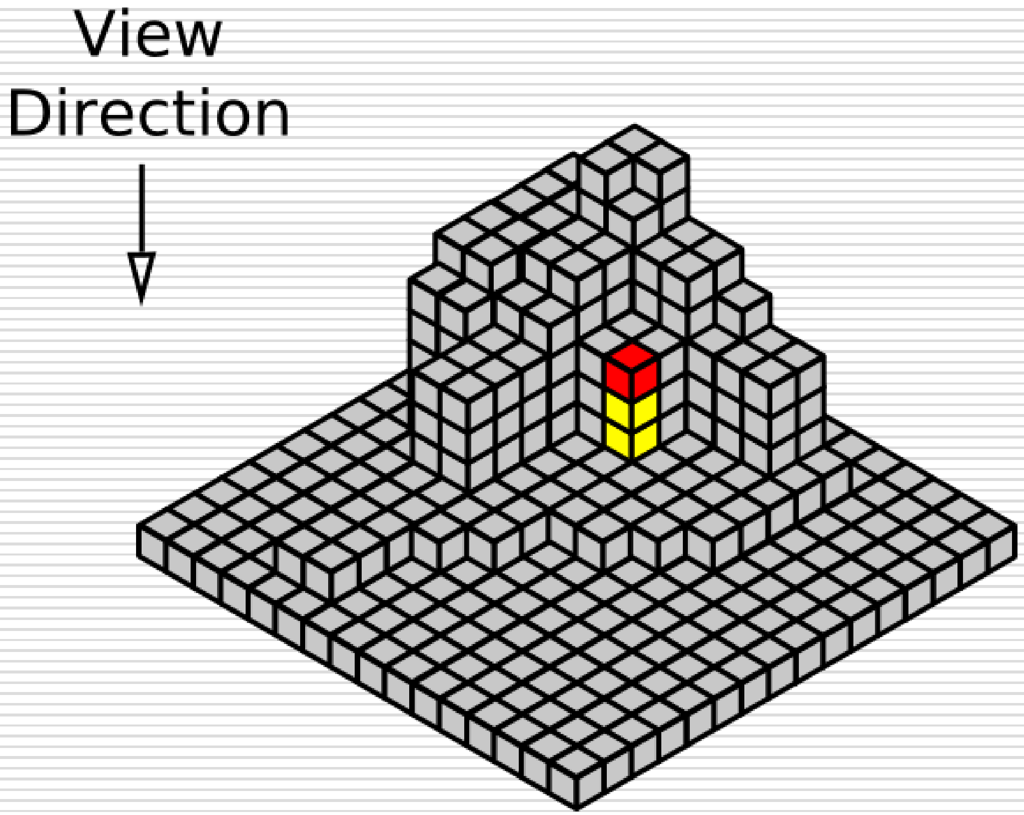
```
int _tmain(int argc, _TCHAR* argv[])
{
  glutInit(&argc, (char**) argv);
  glutInitDisplayMode(GLUT_DEPTH |
                    GLUT_DOUBLE | GLUT_RGBA);
  glutInitWindowPosition(100,100);
  glutInitWindowSize(320,320);
  glutCreateWindow ("2015 Spring Course");
 }
```

Buffers

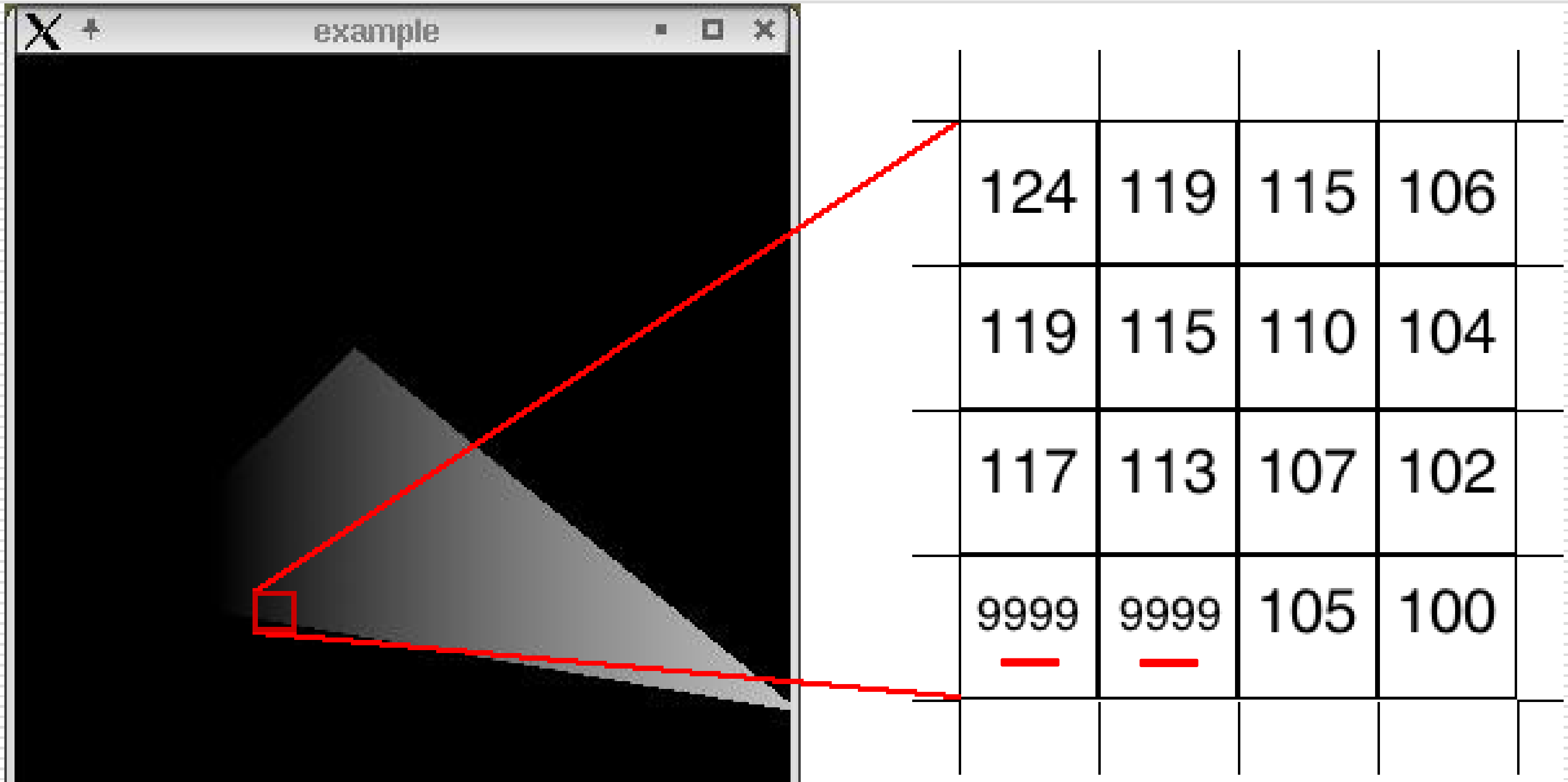
- OpenGL composites the fragments and store its final output in buffers
- Buffers are 2D arrays of data, generally correlating to per-pixel information
- Use buffers to store the results of intermediate stages for later use
- The final frame buffer is output to the screen
  - Note that it does not automatically clear each Buffer; should be manually cleared with glClear

The first OpenGL program

Depth buffer



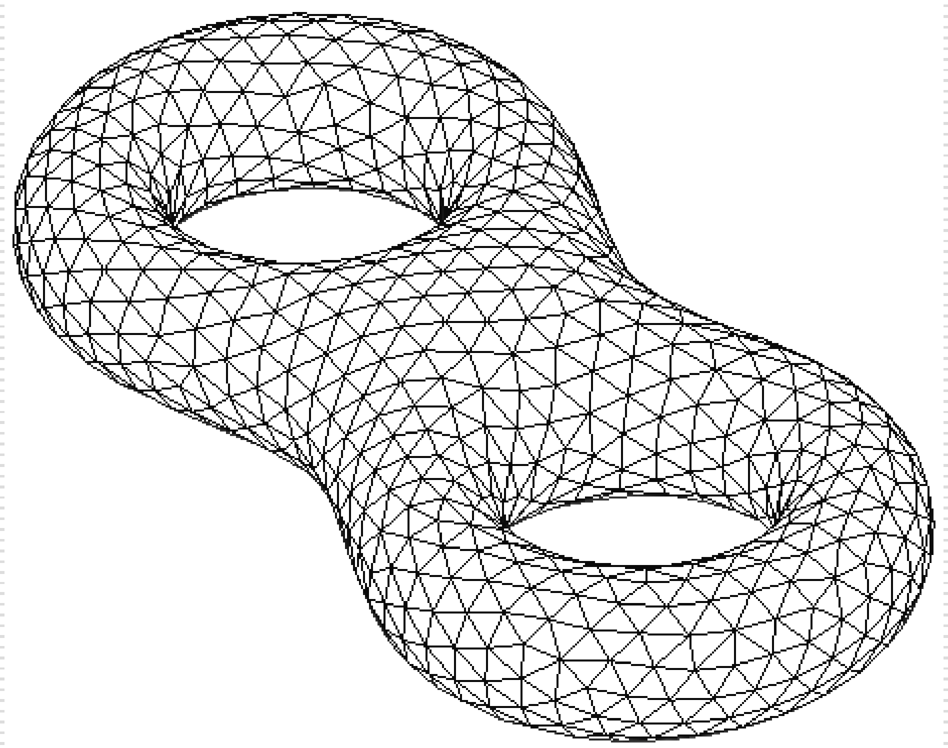
The first OpenGL program

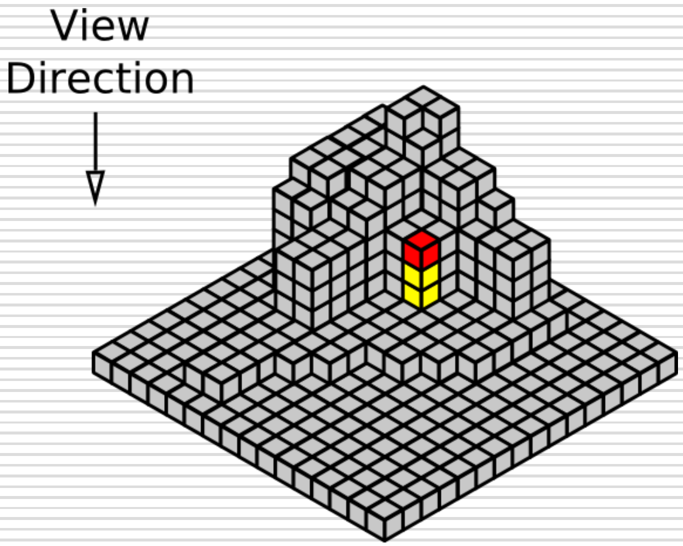


124	119	115	106
119	115	110	104
117	113	107	102
9999	9999	105	100

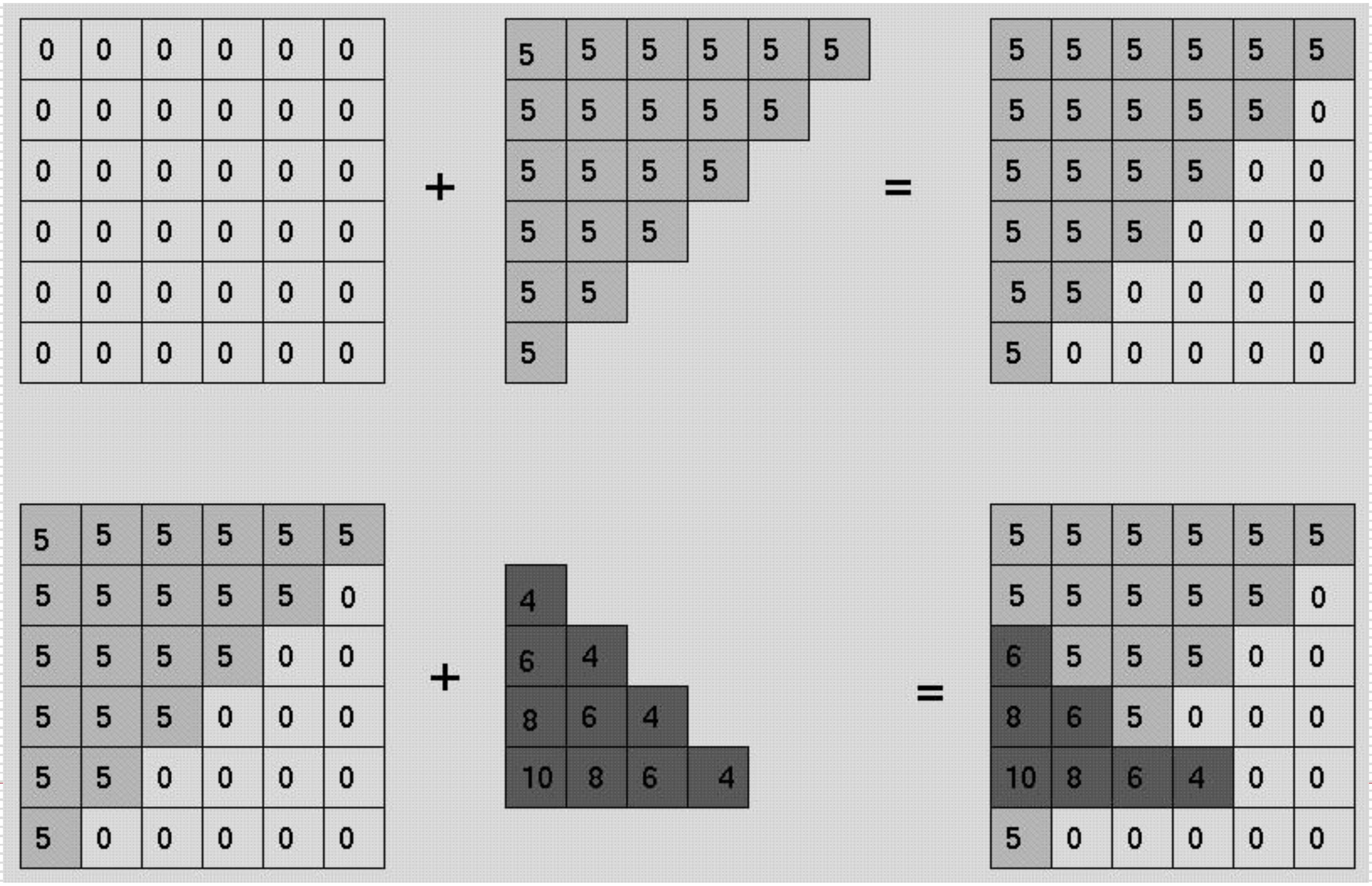
— means far away

The first OpenGL program





The first OpenGL program



$$\begin{array}{|c|c|c|c|c|c|}\hline 0&0&0&0&0&0\\\hline 0&0&0&0&0&0\\\hline 0&0&0&0&0&0\\\hline 0&0&0&0&0&0\\\hline 0&0&0&0&0&0\\\hline 0&0&0&0&0&0\\\hline\end{array} + \begin{array}{cccccc} 5&5&5&5&5&5\\ 5&5&5&5&5&\\ 5&5&5&5&&\\ 5&5&5&&&\\ 5&5&&&&\\ 5&&&&&\end{array} = \begin{array}{|c|c|c|c|c|c|}\hline 5&5&5&5&5&5\\\hline 5&5&5&5&5&0\\\hline 5&5&5&5&0&0\\\hline 5&5&5&0&0&0\\\hline 5&5&0&0&0&0\\\hline 5&0&0&0&0&0\\\hline\end{array}$$

$$\begin{array}{|c|c|c|c|c|c|}\hline 5&5&5&5&5&5\\\hline 5&5&5&5&5&0\\\hline 5&5&5&5&0&0\\\hline 5&5&5&0&0&0\\\hline 5&5&0&0&0&0\\\hline 5&0&0&0&0&0\\\hline\end{array} + \begin{array}{cccccc} &&&&&\\ 4&&&&&\\ 6&4&&&&\\ 8&6&4&&&\\ 10&8&6&4&&\\ &&&&&\end{array} = \begin{array}{|c|c|c|c|c|c|}\hline 5&5&5&5&5&5\\\hline 5&5&5&5&5&0\\\hline 6&5&5&5&0&0\\\hline 8&6&5&0&0&0\\\hline 10&8&6&4&0&0\\\hline 5&0&0&0&0&0\\\hline\end{array}$$

Examples of common used buffers

- Color buffers
  - Contain information about the color of pixel
- Depth (Z) buffers
  - Stores depth information of each pixel, which is used to correctly draw closer objects in front of farther ones
- Stencil buffers
  - Used for cropping with complicated shapes
- Accumulation buffer
  - Used to store intermediate results for later use

Buffers are simply arrays with some specified format; they can be used for any purpose.

The first OpenGL program

```
int _tmain(int argc, _TCHAR* argv[])
{
   … …
   glutDisplayFunc(renderScene);
   //register redraw event handler into system
 }
```

The first OpenGL program

```
void renderScene(void)
{
   glClear(GL_COLOR_BUFFER_BIT|GL_DEPTH_BUFFER_BIT);
  glLoadIdentity();
  glBegin(GL_TRIANGLES);
       glVertex3f(-0.5,-0.5,0.0);
       glVertex3f(0.5,0.0,0.0);
       glVertex3f(0.0,0.5,0.0);
  glEnd();
  glutSwapBuffers(); }
```

The first OpenGL program

```
int _tmain(int argc, _TCHAR* argv[])
{
   … … …
   glutDisplayFunc(renderScene);
   //register redraw event handler into system
   glutMainLoop();
   //enters the GLUT event processing loop
 }
```

The first OpenGL program

A executable, simple OpenGL program can run now！

The first OpenGL program

Add color control：

glColor3f(red, green, blue);

Define at the beginning of the program

float red=1.0, blue=1.0, green=1.0;

```
float red=1.0, blue=1.0, green=1.0;
void renderScene(void)
{
  glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
  glLoadIdentity();
  glColor3f(red, green, blue);
  glBegin(GL_TRIANGLES);
     glVertex3f(-0.5,-0.5,0.0);
     glVertex3f(0.5,0.0,0.0);
     glVertex3f(0.0,0.5,0.0);
  glEnd();
  glutSwapBuffers();
 }
```

The first OpenGL program

```
int _tmain(int argc, _TCHAR* argv[])
{
   … …
   glutDisplayFunc(renderScene);
   //create menu（to control color）
   glutMainLoop();
   … …
 }
```

```
int _tmain(int argc, _TCHAR* argv[]) {
   … …
   glutDisplayFunc(renderScene);
   glutCreateMenu(processMenuEvents);
   //register callback funciton processMenuEvents
   glutAddMenuEntry("Red",RED);
   //add option in menu
   glutAddMenuEntry("Blue",BLUE);
   glutAddMenuEntry("Green",GREEN);
   glutAddMenuEntry("White",WHITE);
   //create the connection to mouse button
   glutAttachMenu(GLUT_RIGHT_BUTTON);
   glutMainLoop();            }
```

```
#define RED 1
#define GREEN 2
#define BLUE 3
#define WHITE 4
```

```
void processMenuEvents(int option) {
  switch (option) {
   case RED : red = 1.0; green = 0.0; blue = 0.0; break;
   case GREEN: red = 0.0; green = 1.0; blue = 0.0; break;
   case BLUE : red = 0.0; green = 0.0; blue = 1.0; break;
   case WHITE : red = 1.0; green = 1.0; blue = 1.0; break;
   }
   glutPostRedisplay();
}
```

Add animation

We can specify a function in glutIdleFunc(function pointer parameter)，if there is no event to handle（i.e. event loop is in idle status），then execute this function.

Add animation

```
int _tmain(int argc, _TCHAR* argv[])
{
   … …
   glutDisplayFunc(renderScene);
   glutCreateMenu(processMenuEvents);
   //if no event happens, execute this function
   glutIdleFunc(renderScene);
   glutMainLoop();
}
```

```
float angle = 0.0;
void renderScene(void)
{  glClear(GL_COLOR_BUFFER_BIT|GL_DEPTH_BUFFER_BIT);
   glLoadIdentity();
   glRotatef(angle,0.0,1.0,0.0);
   glColor3f(red, green, blue);
   glBegin(GL_TRIANGLES);
        glVertex3f(-0.5,-0.5,0.0);
        glVertex3f(0.5,0.0,0.0);
        glVertex3f(0.0,0.5,0.0);
   glEnd();
   angle += 0.2;
   glutSwapBuffers(); }
```

VC++ platform setting

- Set default searching directory of the VC++ project
  - Different for VC++.net 2010 and previous version
  - Discuss with TA

Set default searching directory in VC++ 2008

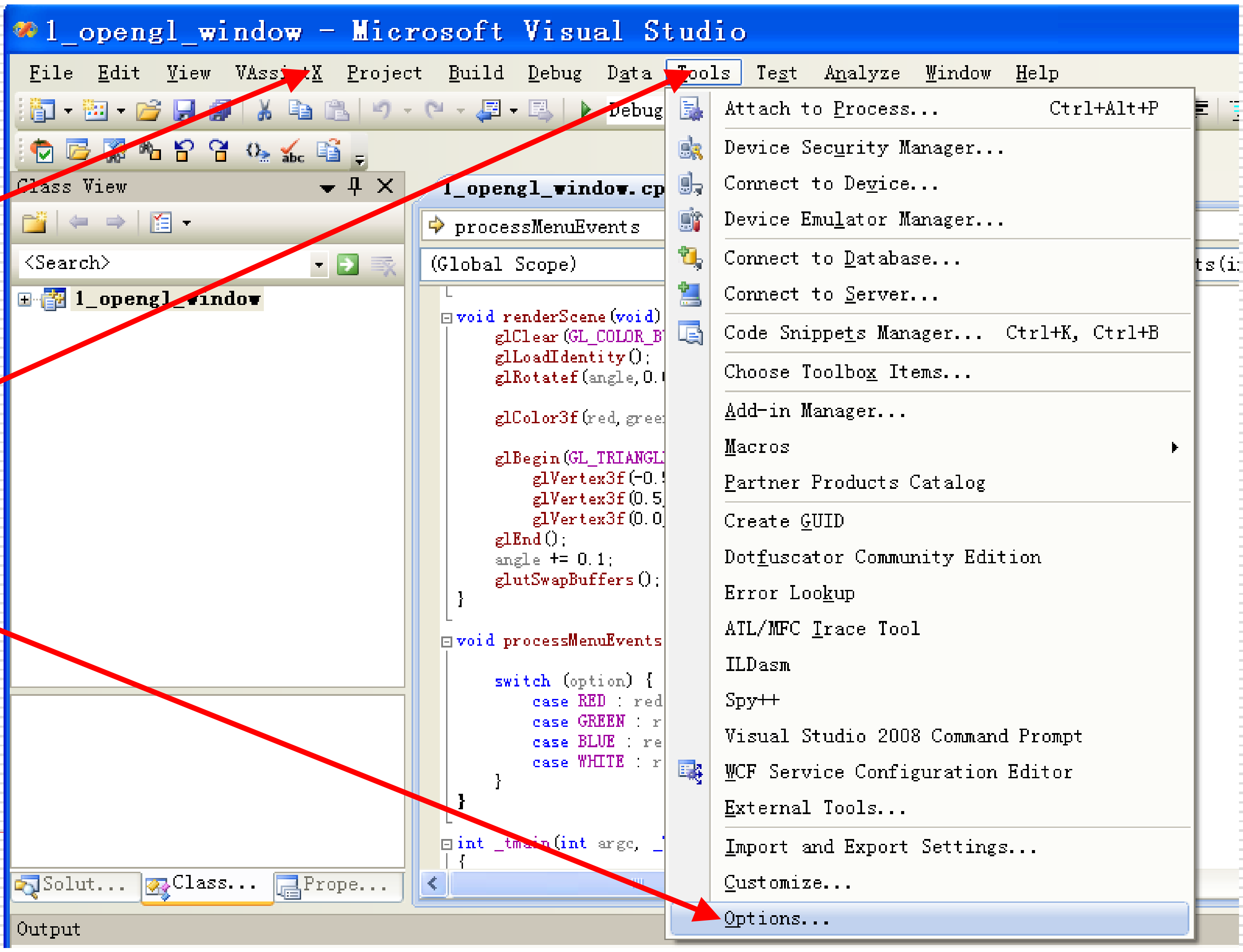
Visual studio platform menu

↓

"Tool" menu

↓

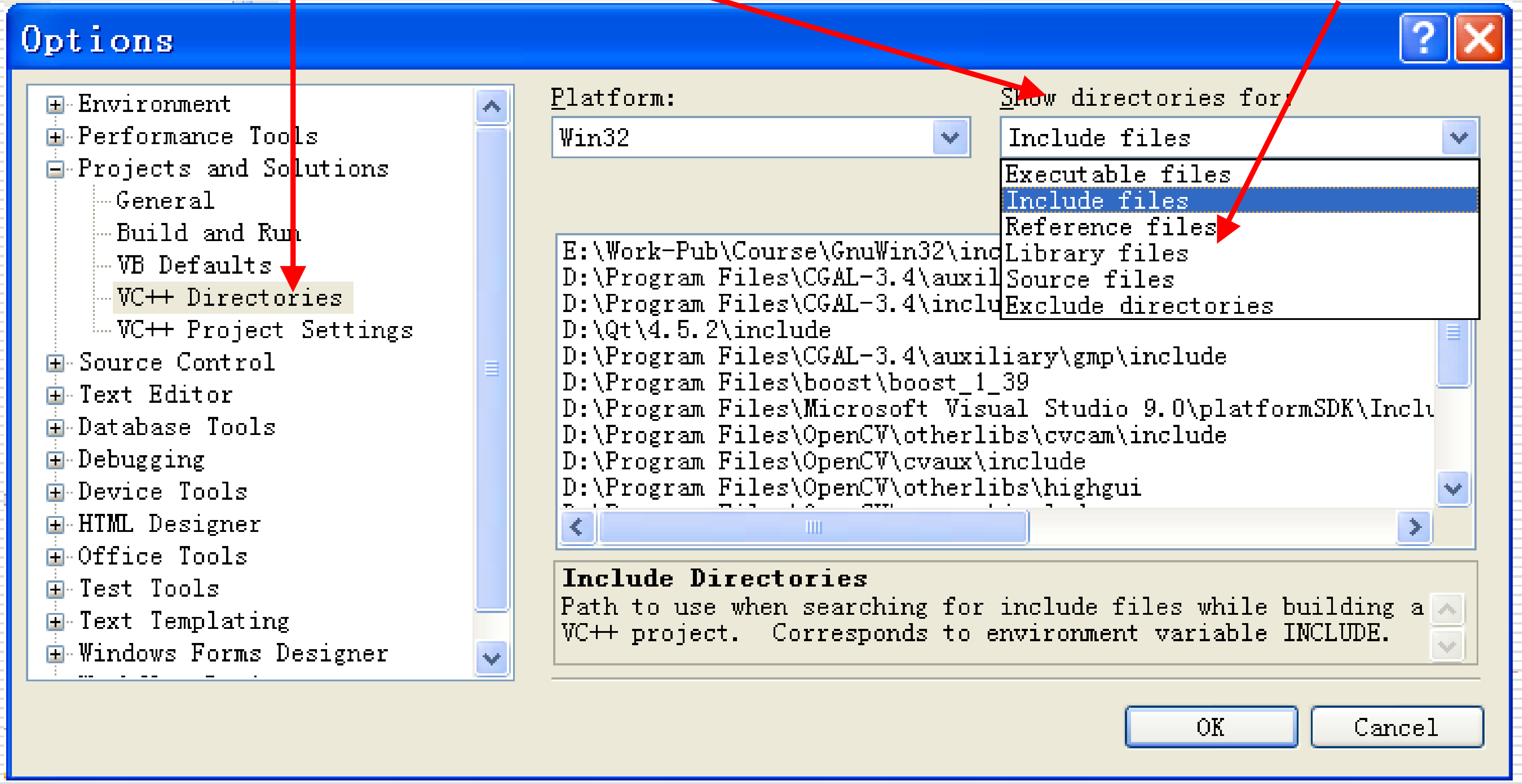
Options



```
void renderScene(void)
    glClear(GL_COLOR_B
    glLoadIdentity();
    glRotatef(angle, 0.

    glColor3f(red, gree

    glBegin(GL_TRIANGL
        glVertex3f(-0.
        glVertex3f(0.5
        glVertex3f(0.0
    glEnd();
    angle += 0.1;
    glutSwapBuffers();
}

void processMenuEvents

    switch (option) {
        case RED : red
        case GREEN : r
        case BLUE : re
        case WHITE : r
    }
}

int _tmain(int argc, _
{
```

Set default searching directory in VC++ 2008

VC++ Directories→ Show directories for → Include for .h
Library file for .lib



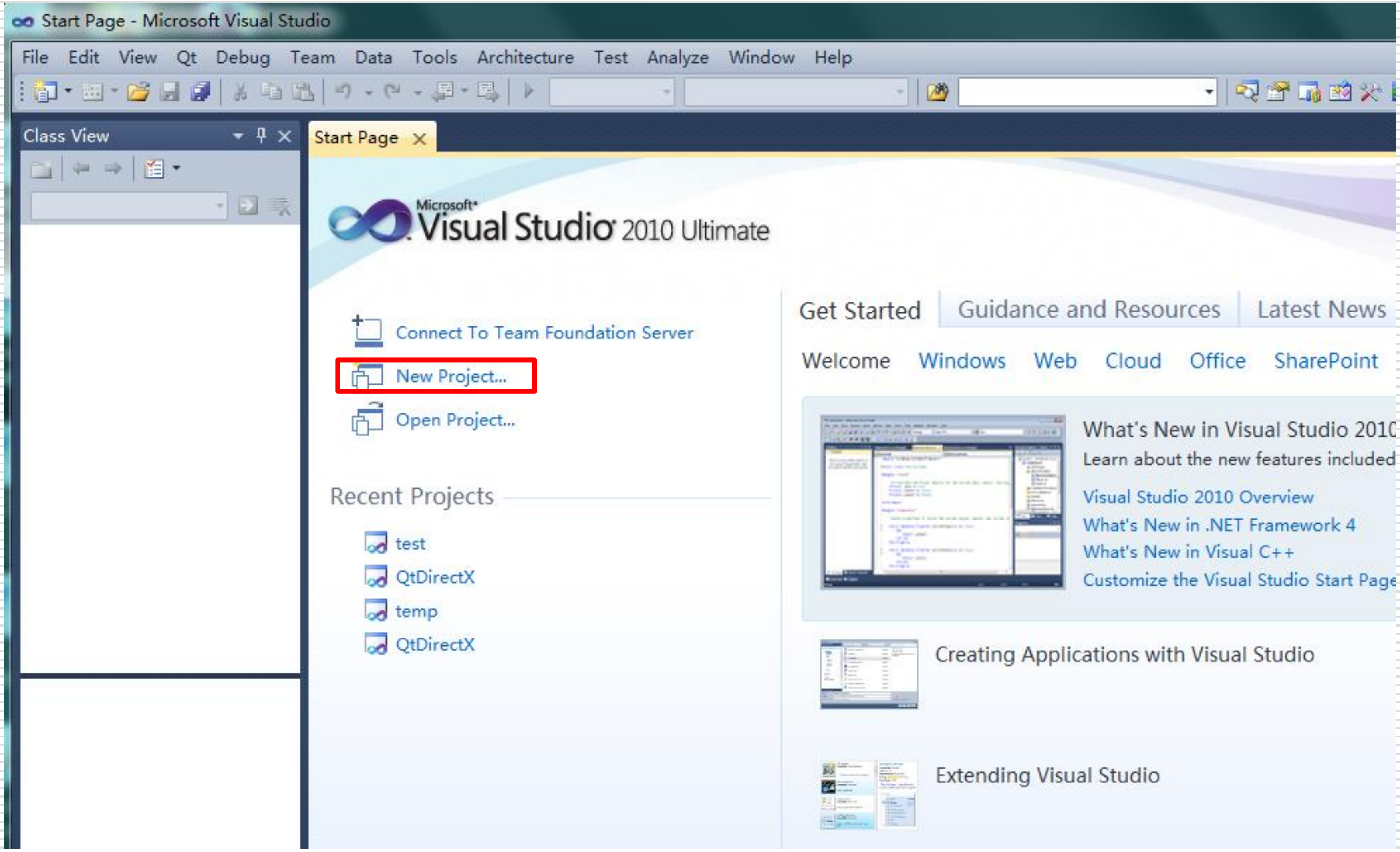
Platform: Win32

Show directories for: Include files

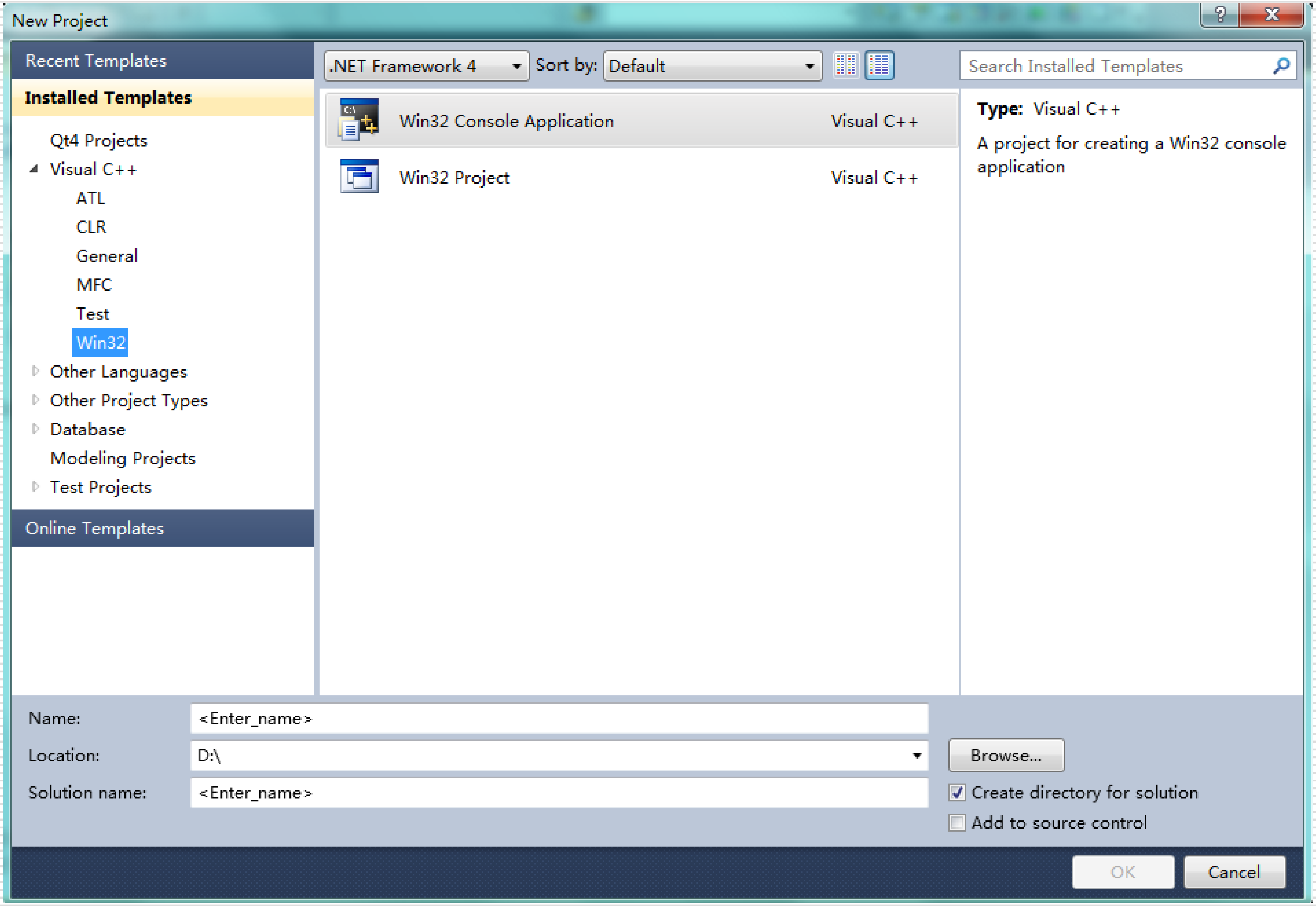
```
E:\Work-Pub\Course\GnuWin32\inc
D:\Program Files\CGAL-3.4\auxil
D:\Program Files\CGAL-3.4\inclu
D:\Qt\4.5.2\include
D:\Program Files\CGAL-3.4\auxiliary\gmp\include
D:\Program Files\boost\boost_1_39
D:\Program Files\Microsoft Visual Studio 9.0\platformSDK\Inclu
D:\Program Files\OpenCV\otherlibs\cvcam\include
D:\Program Files\OpenCV\cvaux\include
D:\Program Files\OpenCV\otherlibs\highgui
```

Include Directories
Path to use when searching for include files while building a VC++ project. Corresponds to environment variable INCLUDE.

Starting in VC++ 2010



Start Page - Microsoft Visual Studio
File Edit View Qt Debug Team Data Tools Architecture Test Analyze Window Help
Class View
Start Page
Microsoft Visual Studio 2010 Ultimate
Connect To Team Foundation Server
New Project...
Open Project...
Recent Projects
test
QtDirectX
temp
QtDirectX
Get Started
Guidance and Resources
Latest News
Welcome
Windows
Web
Cloud
Office
SharePoint
What's New in Visual Studio 201
Learn about the new features included
Visual Studio 2010 Overview
What's New in .NET Framework 4
What's New in Visual C++
Customize the Visual Studio Start Page
Creating Applications with Visual Studio
Extending Visual Studio



Recent Templates

Installed Templates

- Qt4 Projects
- Visual C++
  - ATL
  - CLR
  - General
  - MFC
  - Test
  - Win32
- Other Languages
- Other Project Types
- Database
- Modeling Projects
- Test Projects

Online Templates

.NET Framework 4 Sort by: Default

Search Installed Templates

Win32 Console Application	Visual C++
Win32 Project	Visual C++

Type: Visual C++

A project for creating a Win32 console application

Name: <Enter_name>

Location: D:\

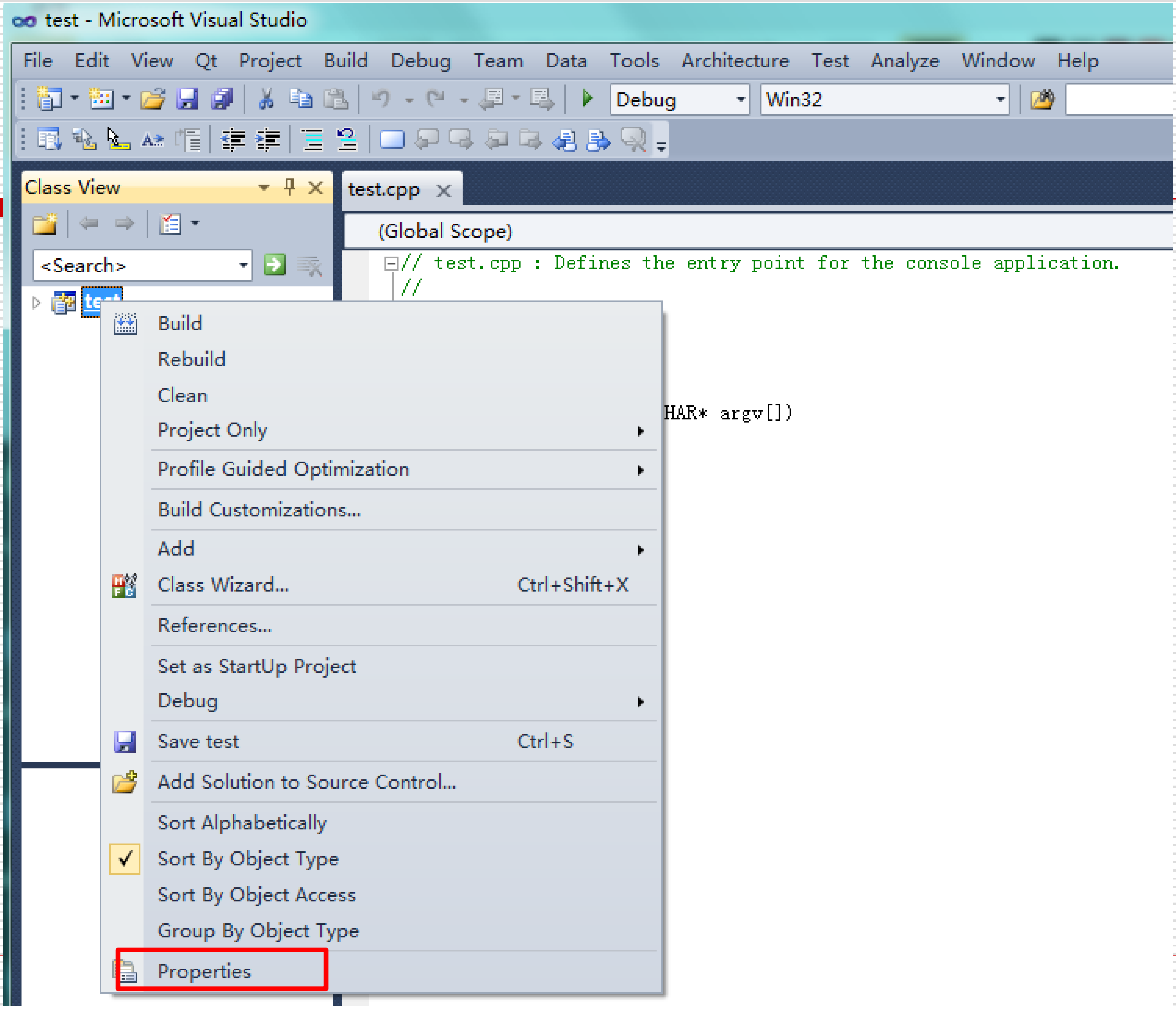
Solution name: <Enter_name>

Browse...

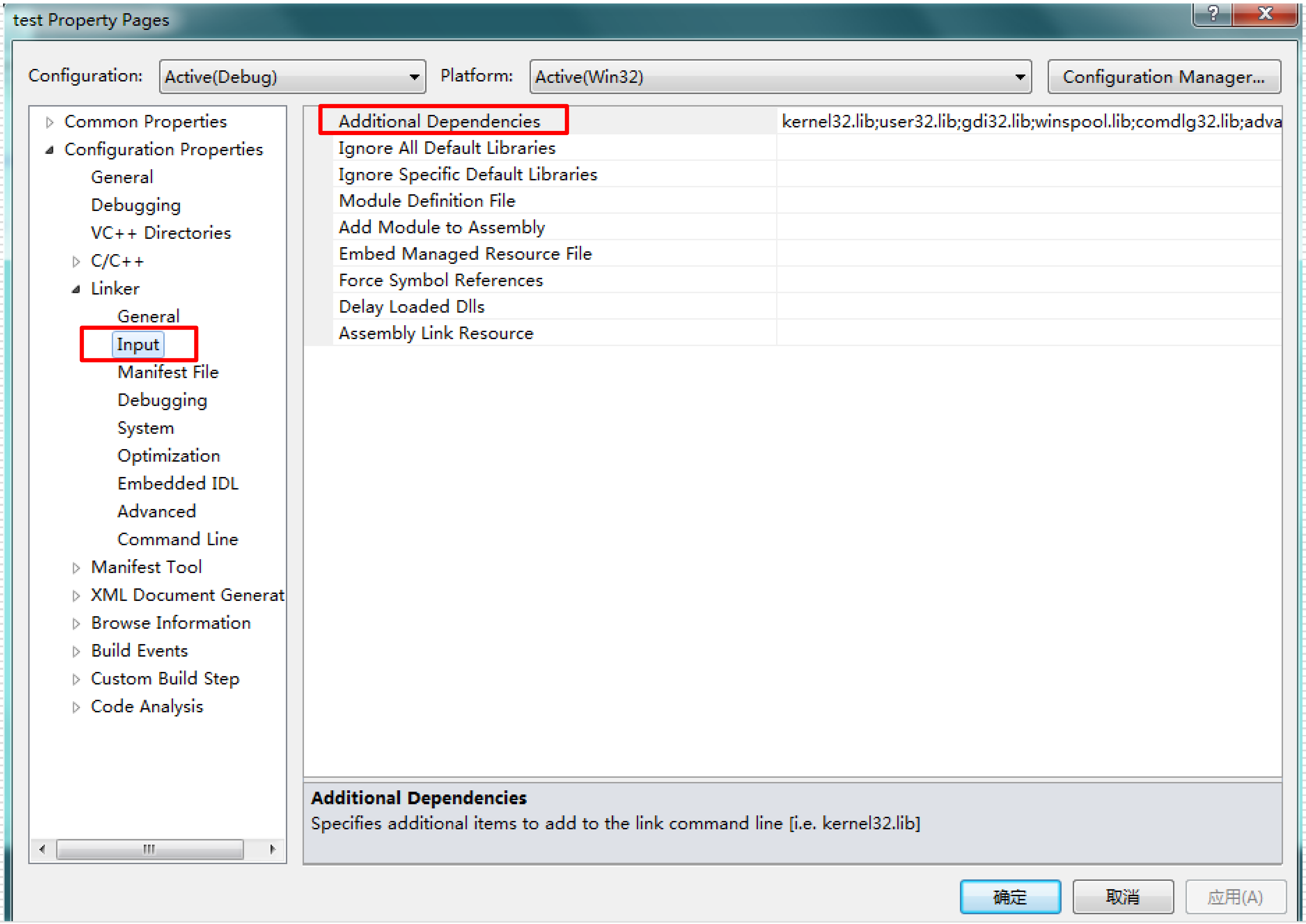
[✓] Create directory for solution

[ ] Add to source control

OK Cancel



```
// test.cpp : Defines the entry point for the console application.
//
HAR* argv[])
```

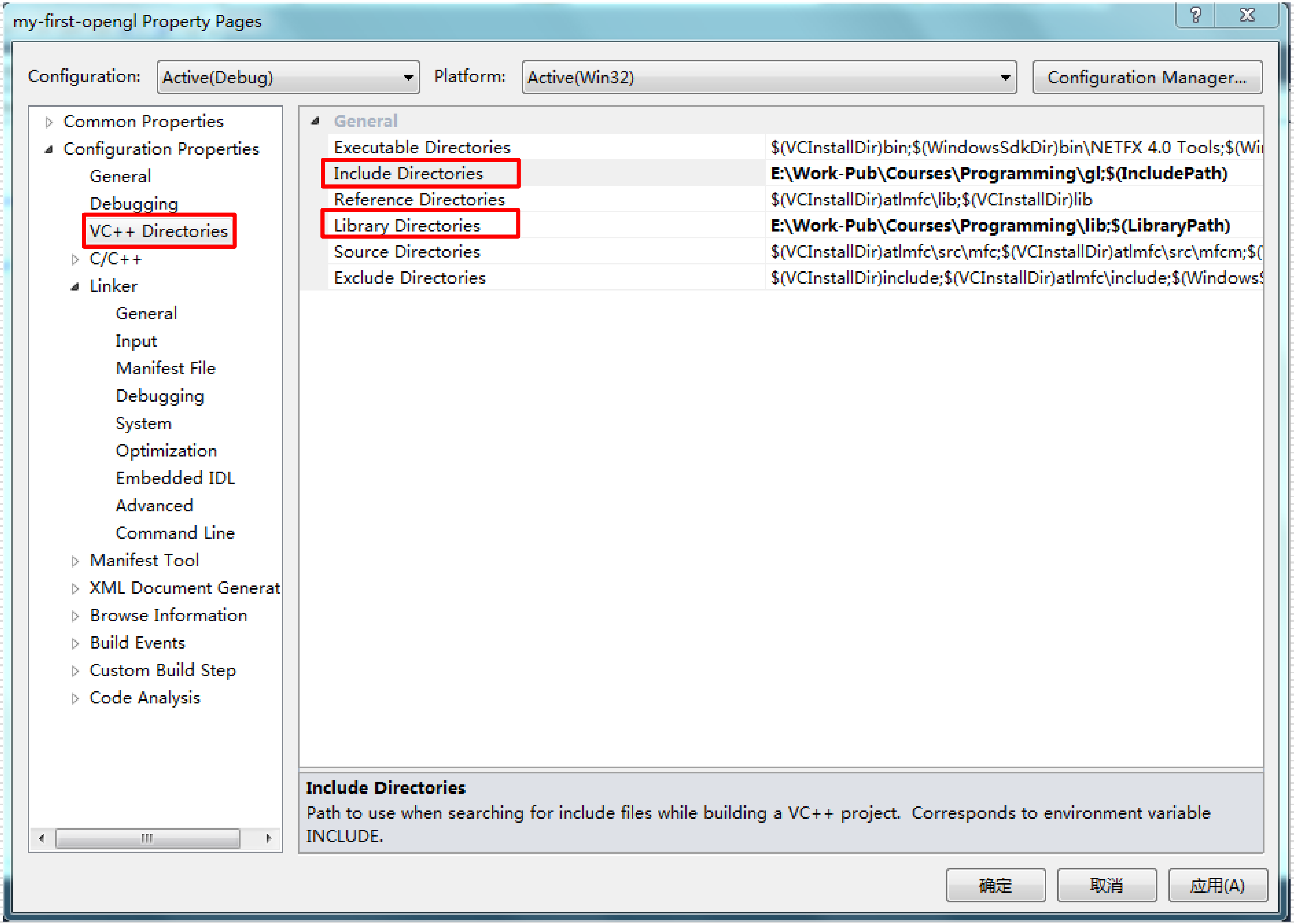

test Property Pages

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

- Common Properties
- Configuration Properties
  - General
  - Debugging
  - VC++ Directories
  - C/C++
  - Linker
    - General
    - Input
    - Manifest File
    - Debugging
    - System
    - Optimization
    - Embedded IDL
    - Advanced
    - Command Line
  - Manifest Tool
  - XML Document Generat
  - Browse Information
  - Build Events
  - Custom Build Step
  - Code Analysis

Additional Dependencies	kernel32.lib;user32.lib;gdi32.lib;winspool.lib;comdlg32.lib;adva
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]

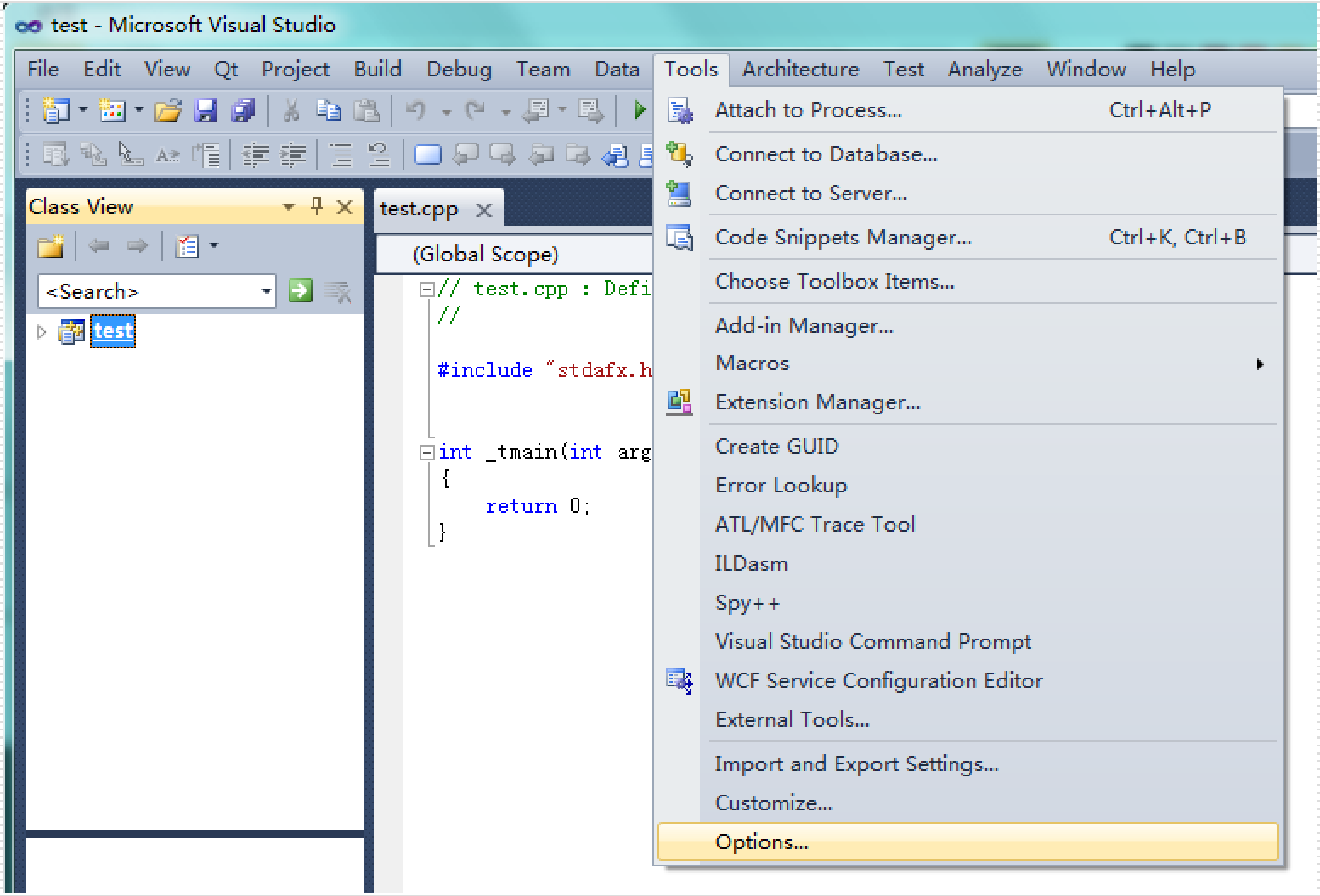


Configuration: Active(Debug) Platform: Active(Win32)

General

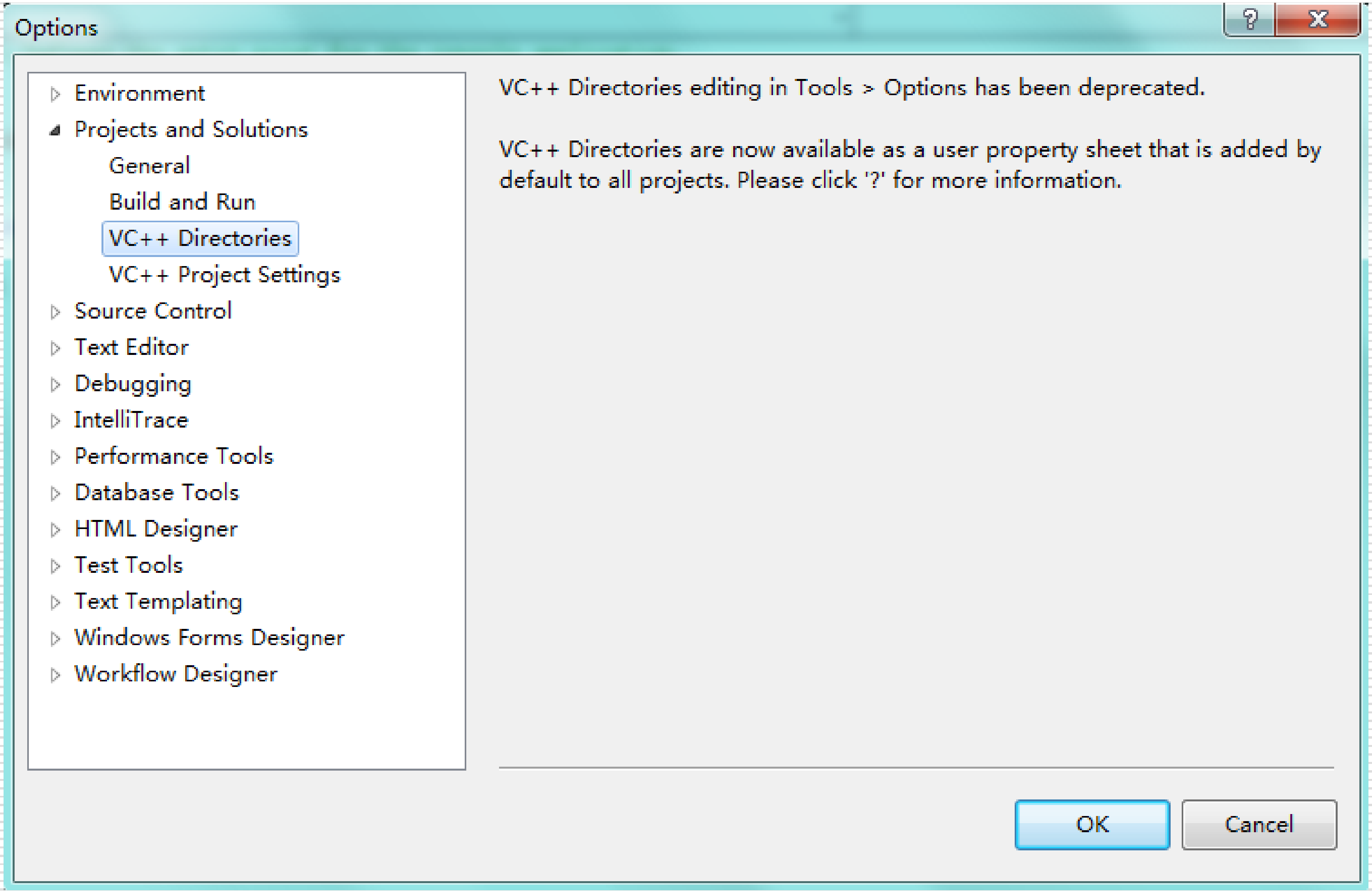
Executable Directories	$(VCInstallDir)bin;$(WindowsSdkDir)bin\NETFX 4.0 Tools;$(Wi
Include Directories	**E:\Work-Pub\Courses\Programming\gl;$(IncludePath)**
Reference Directories	$(VCInstallDir)atlmfc\lib;$(VCInstallDir)lib
Library Directories	**E:\Work-Pub\Courses\Programming\lib;$(LibraryPath)**
Source Directories	$(VCInstallDir)atlmfc\src\mfc;$(VCInstallDir)atlmfc\src\mfcm;$(
Exclude Directories	$(VCInstallDir)include;$(VCInstallDir)atlmfc\include;$(Windows

Include Directories
Path to use when searching for include files while building a VC++ project. Corresponds to environment variable INCLUDE.


test - Microsoft Visual Studio
File Edit View Qt Project Build Debug Team Data Tools Architecture Test Analyze Window Help
Class View
<Search>
test
test.cpp
(Global Scope)
Attach to Process... Ctrl+Alt+P
Connect to Database...
Connect to Server...
Code Snippets Manager... Ctrl+K, Ctrl+B
Choose Toolbox Items...
Add-in Manager...
Macros
Extension Manager...
Create GUID
Error Lookup
ATL/MFC Trace Tool
ILDasm
Spy++
Visual Studio Command Prompt
WCF Service Configuration Editor
External Tools...
Import and Export Settings...
Customize...
Options...

```
// test.cpp : Defi
//

#include "stdafx.h


int _tmain(int arg
{
    return 0;
}
```

Set directory is no longer available in VC++ 2010



VC++ Directories editing in Tools > Options has been deprecated.

VC++ Directories are now available as a user property sheet that is added by default to all projects. Please click '?' for more information.

Fundamentals of Computer Graphics

End.

Thanks