電腦視覺與應用
Computer Vision and Applications

Lecture02-2

Supplementary material create an OpenCV project

Tzung-Han Lin

National Taiwan University of Science and Technology

Graduate Institute of Color and Illumination Technology

e-mail: thl@mail.ntust.edu.tw

TAIWAN TECH National Taiwan University of Science and Technology

CIT 色彩與照明科技研究所 Graduate Institute of Color and Illumination Technology

Create an OpenCV Project

version 3.2
(Microsoft vs2015, win10 64bit
for example)

Online resource:

Use google search “openCV” for further assistance.

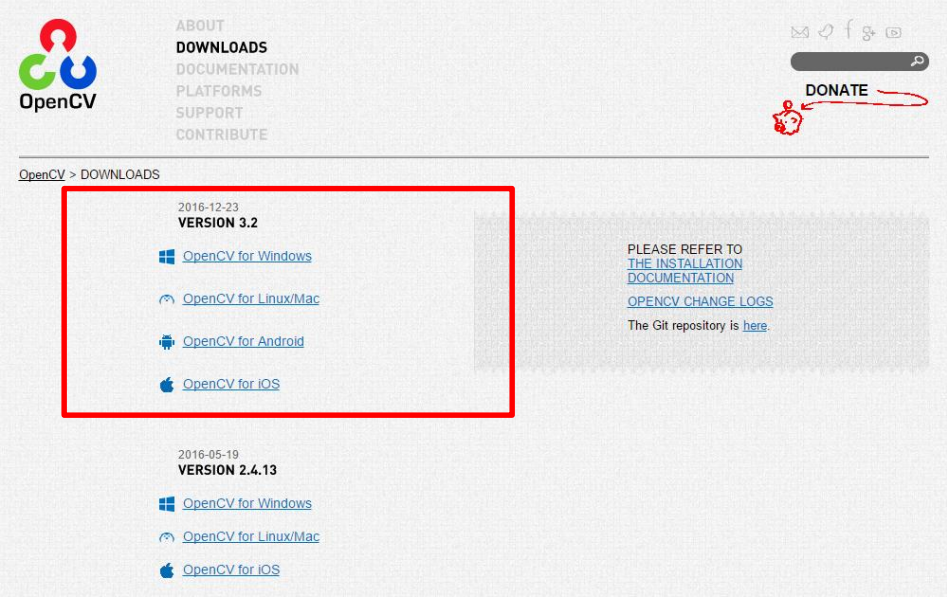
If you need to rebuild openCV for your own platform, for example:

How to build applications with OpenCV inside the *Microsoft Visual Studio*

http://opencv.itseez.com/doc/tutorials/introduction/windows_visual_studio_Opencv/windows_visual_studio_Opencv.html#windows-visual-studio-how-to

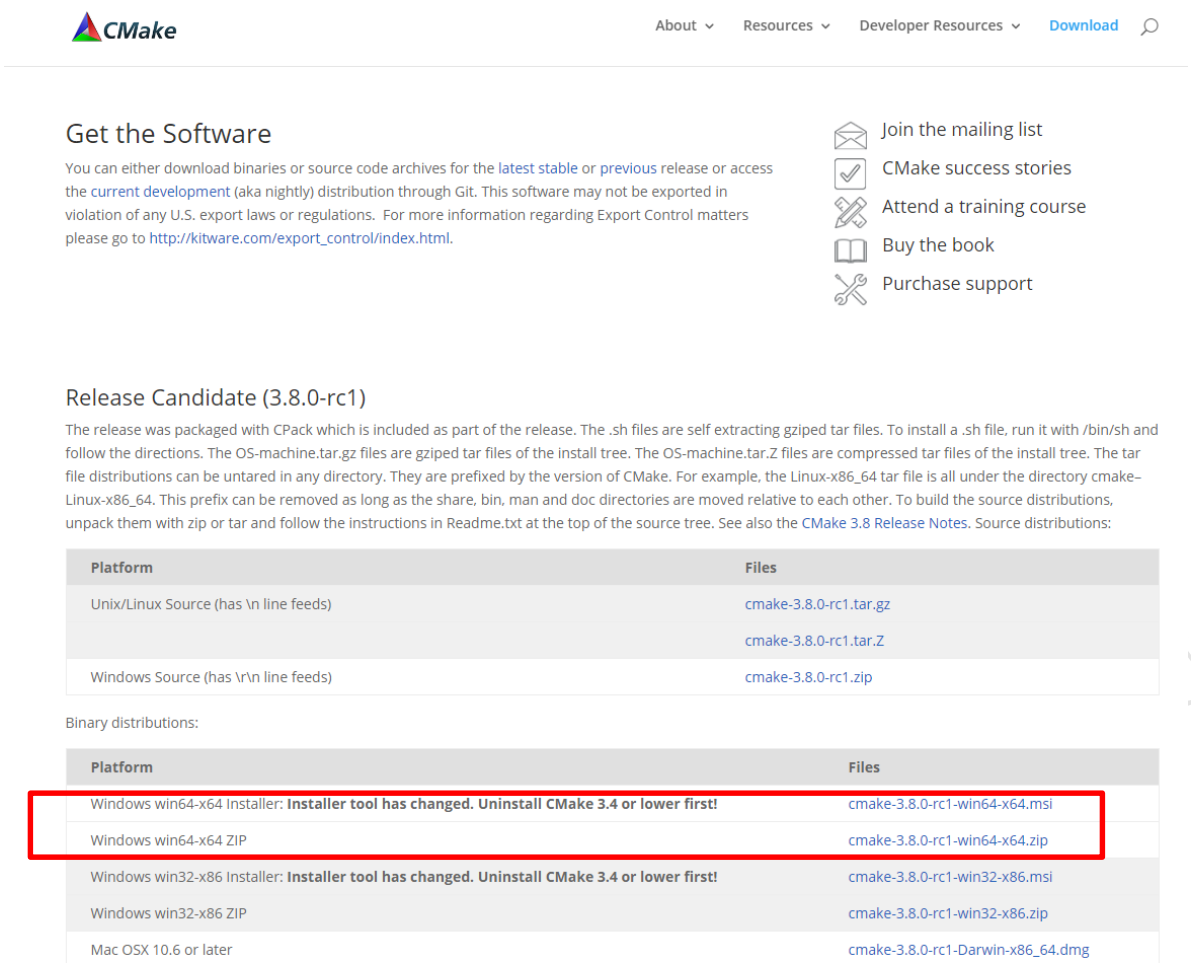
Download openCV from official website or sourceforge
V3.2 is released @2016/12/23

http://opencv.org/downloads.html



CMake → cross form maker

http://cmake.org/cmake/resources/software.html



CMake

About Resources Developer Resources Download

Get the Software

You can either download binaries or source code archives for the latest stable or previous release or access the current development (aka nightly) distribution through Git. This software may not be exported in violation of any U.S. export laws or regulations. For more information regarding Export Control matters please go to http://kitware.com/export_control/index.html.

- Join the mailing list
- CMake success stories
- Attend a training course
- Buy the book
- Purchase support

Release Candidate (3.8.0-rc1)

The release was packaged with CPack which is included as part of the release. The .sh files are self extracting gziped tar files. To install a .sh file, run it with /bin/sh and follow the directions. The OS-machine.tar.gz files are gziped tar files of the install tree. The OS-machine.tar.Z files are compressed tar files of the install tree. The tar file distributions can be untared in any directory. They are prefixed by the version of CMake. For example, the Linux-x86_64 tar file is all under the directory cmake-Linux-x86_64. This prefix can be removed as long as the share, bin, man and doc directories are moved relative to each other. To build the source distributions, unpack them with zip or tar and follow the instructions in Readme.txt at the top of the source tree. See also the CMake 3.8 Release Notes. Source distributions:

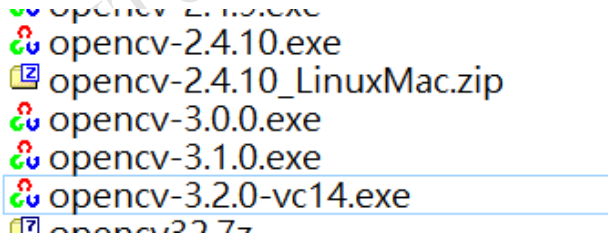
Platform	Files
Unix/Linux Source (has \n line feeds)	cmake-3.8.0-rc1.tar.gz
	cmake-3.8.0-rc1.tar.Z
Windows Source (has \r\n line feeds)	cmake-3.8.0-rc1.zip

Binary distributions:

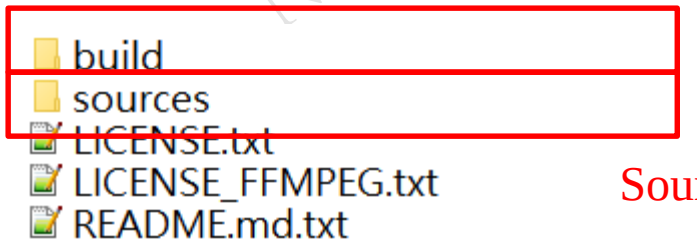
Platform	Files
Windows win64-x64 Installer: **Installer tool has changed. Uninstall CMake 3.4 or lower first!**	cmake-3.8.0-rc1-win64-x64.msi
Windows win64-x64 ZIP	cmake-3.8.0-rc1-win64-x64.zip
Windows win32-x86 Installer: **Installer tool has changed. Uninstall CMake 3.4 or lower first!**	cmake-3.8.0-rc1-win32-x86.msi
Windows win32-x86 ZIP	cmake-3.8.0-rc1-win32-x86.zip
Mac OSX 10.6 or later	cmake-3.8.0-rc1-Darwin-x86_64.dmg

https://cmake.org/download/

Unzip file



unzip

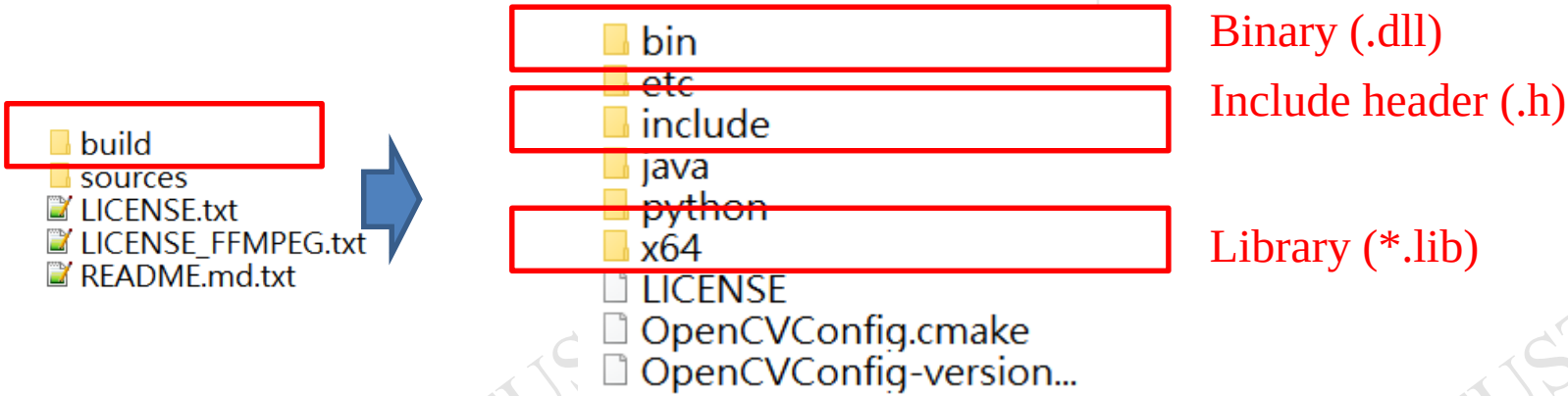


Pre-build for vs2015

Source file

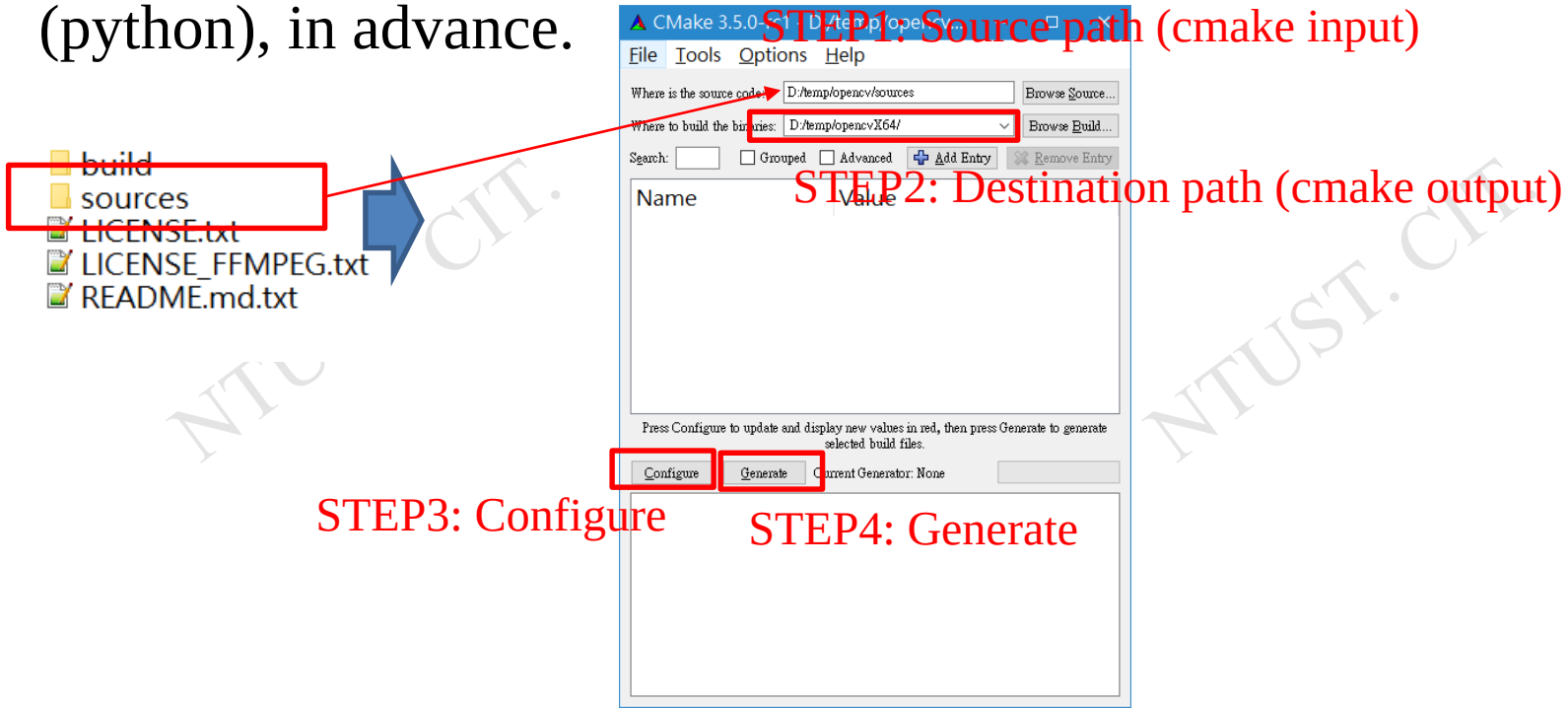
Case 1: Use lib from pre-build

- For C/C++ user
  - All you need ….



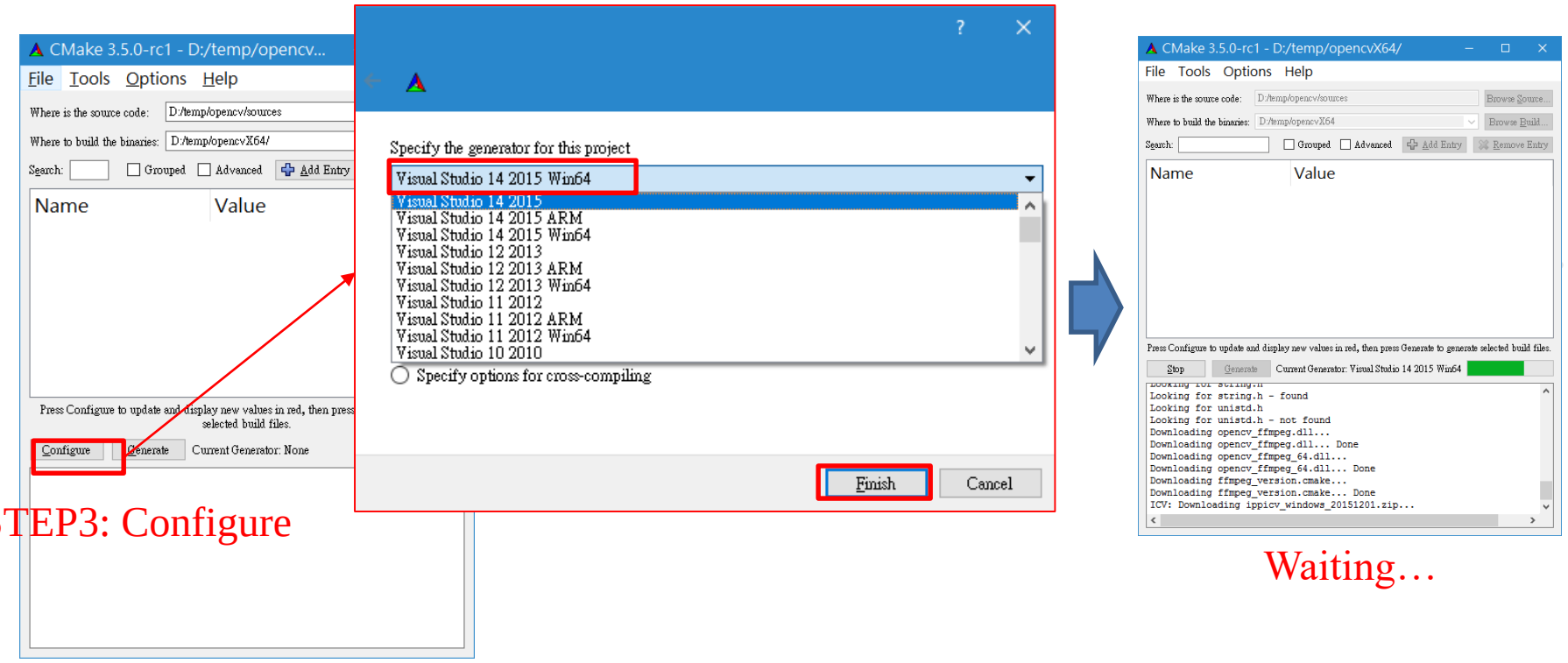
Case 2: rebuild into several platforms

- Use “Cmake” to generate “makefile” for cross platforms
  - For example, to build 64bit lib
  - You need to install visual studio 2015 and Cmake (python), in advance.



Case 2: rebuild into several platforms-cont.

- STEP3:Configure
  - For 64bit

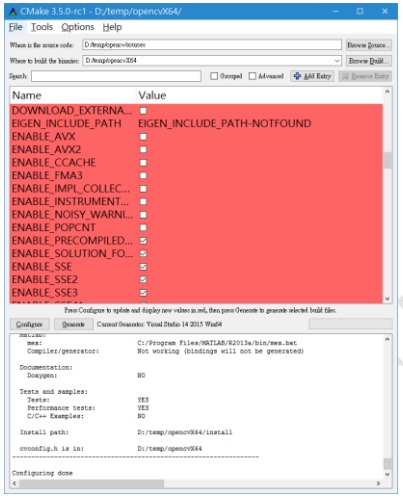


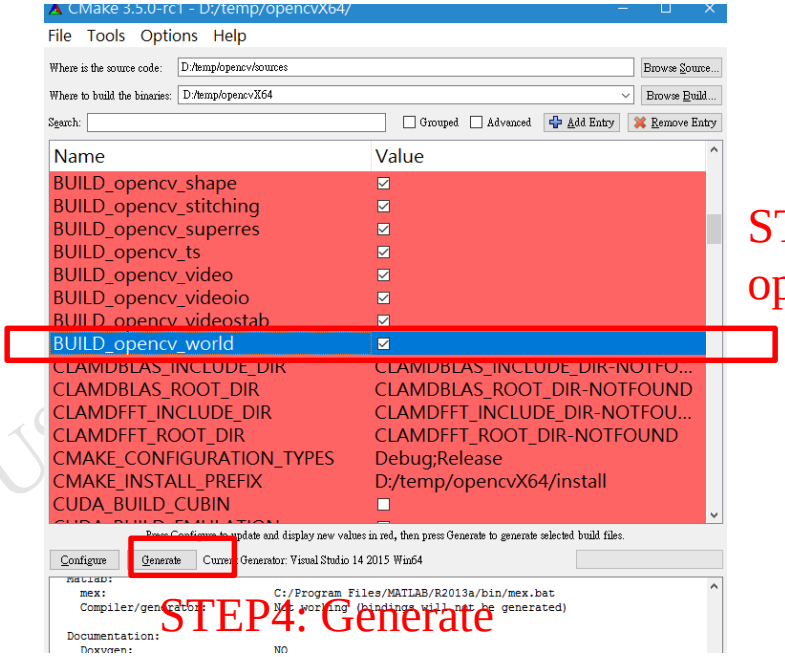
STEP3: Configure

Waiting…

Case 2: rebuild into several platforms-cont.

- STEP3:Configure
  - For 64bit



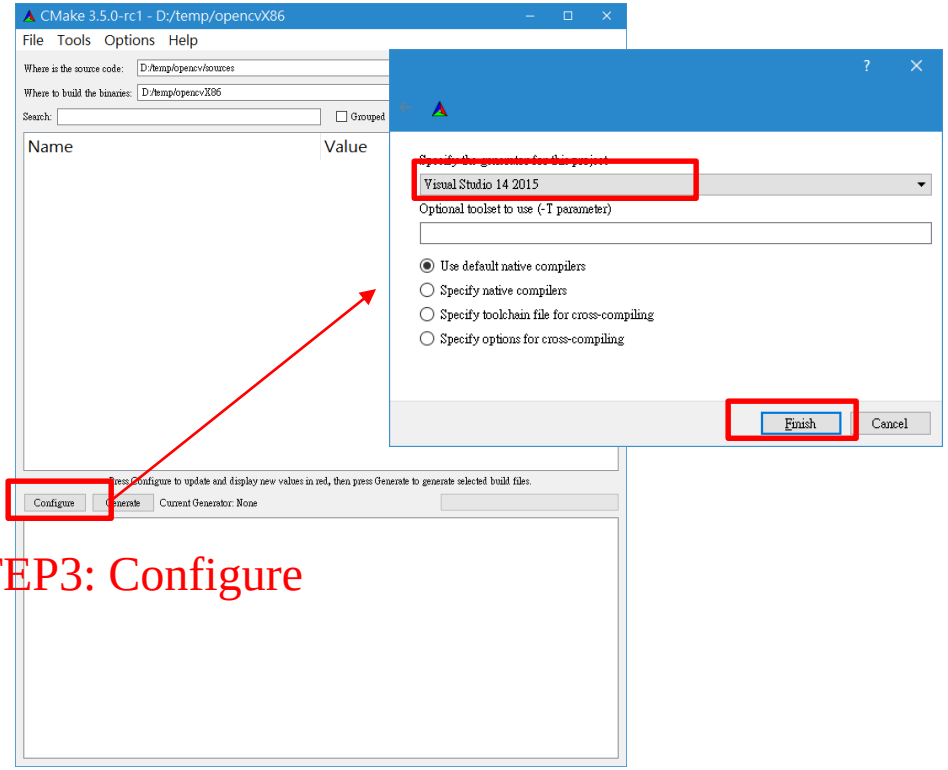


STEP3-1: Checked this option (recommend)

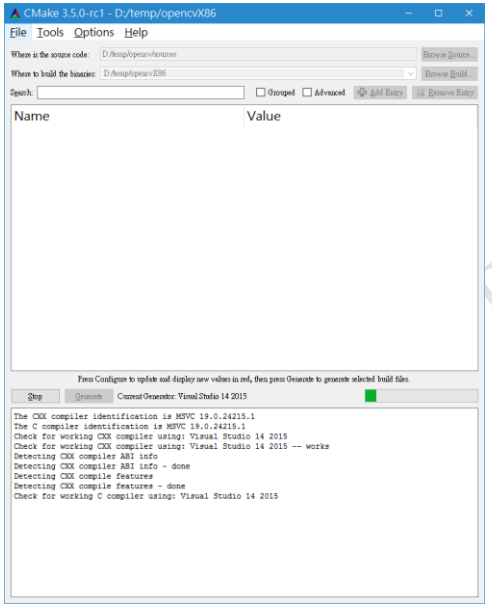
STEP4: Generate

Case 2: rebuild into several platforms-cont.

- STEP3:Configure
  - For 32bit



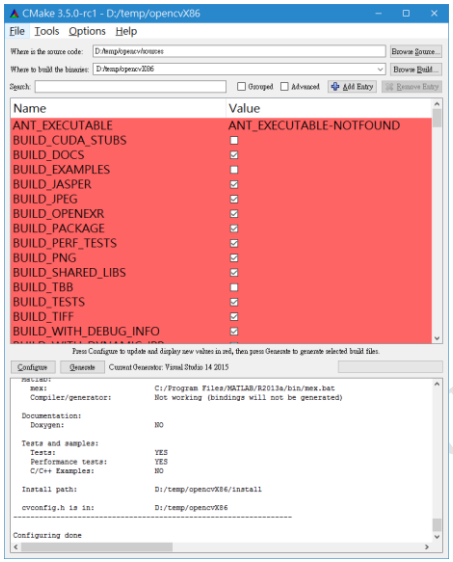
STEP3: Configure

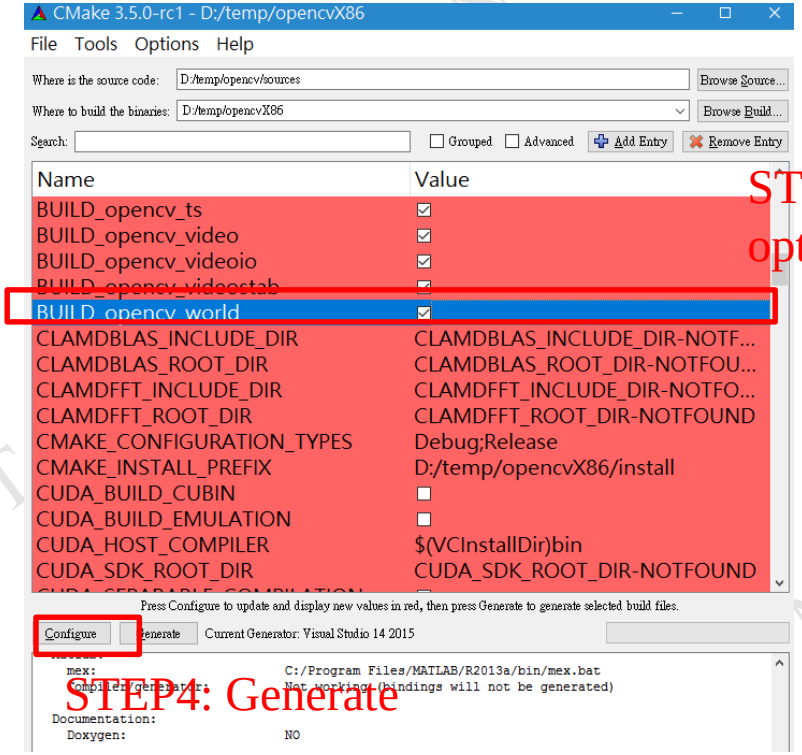


Waiting…

Case 2: rebuild into several platforms-cont.

- STEP3:Configure
  - For 32bit



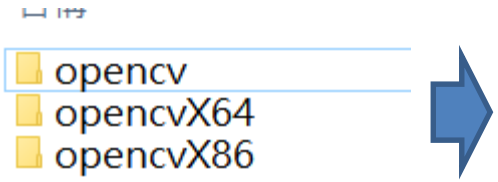


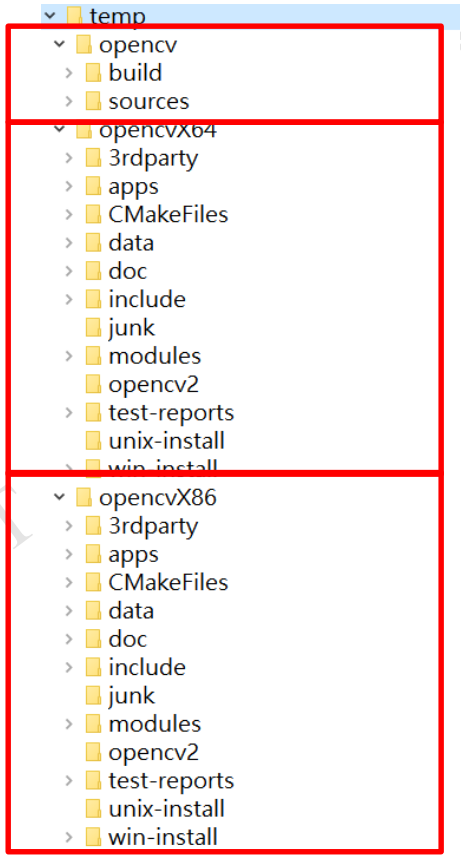
STEP3-1: Checked this option (recommend)

STEP4: Generate

Case 2: rebuild into several platforms-cont.

- After CMake:





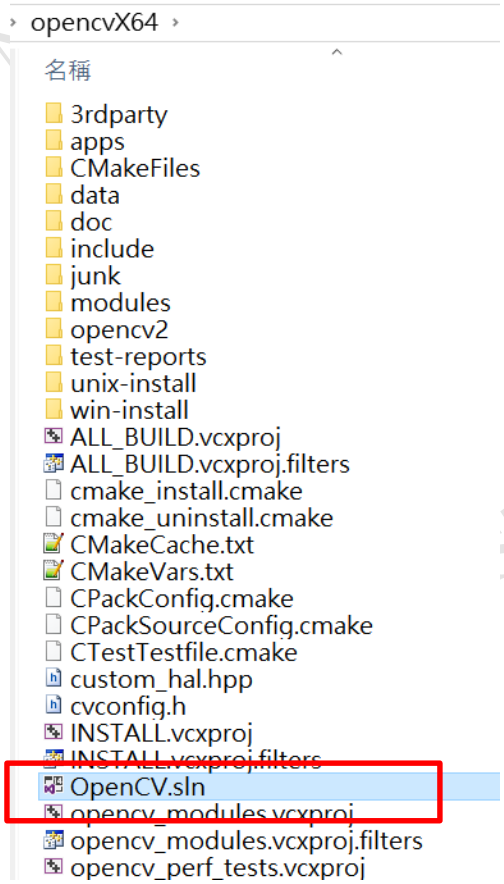
Downloaded source

Cmake generated data (for 64bit)

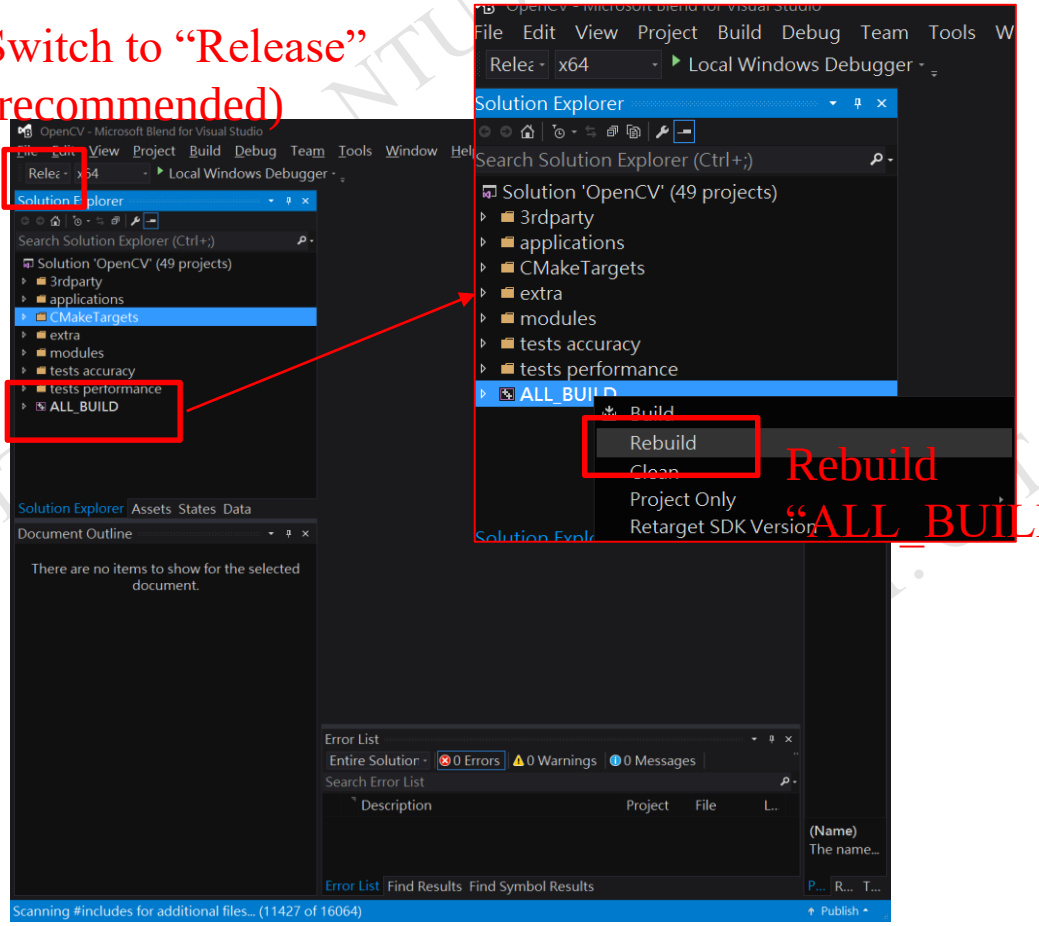
Cmake generated data (for 32bit)

Case 2: rebuild into several platforms-cont.

- Complier your project 64bit (for example) / 32bit

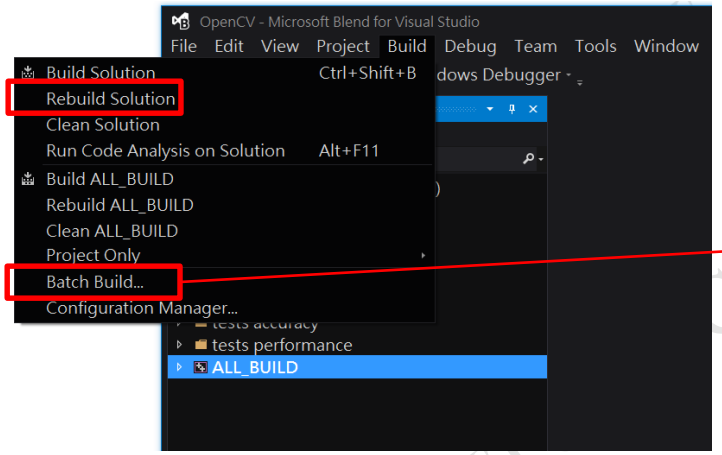


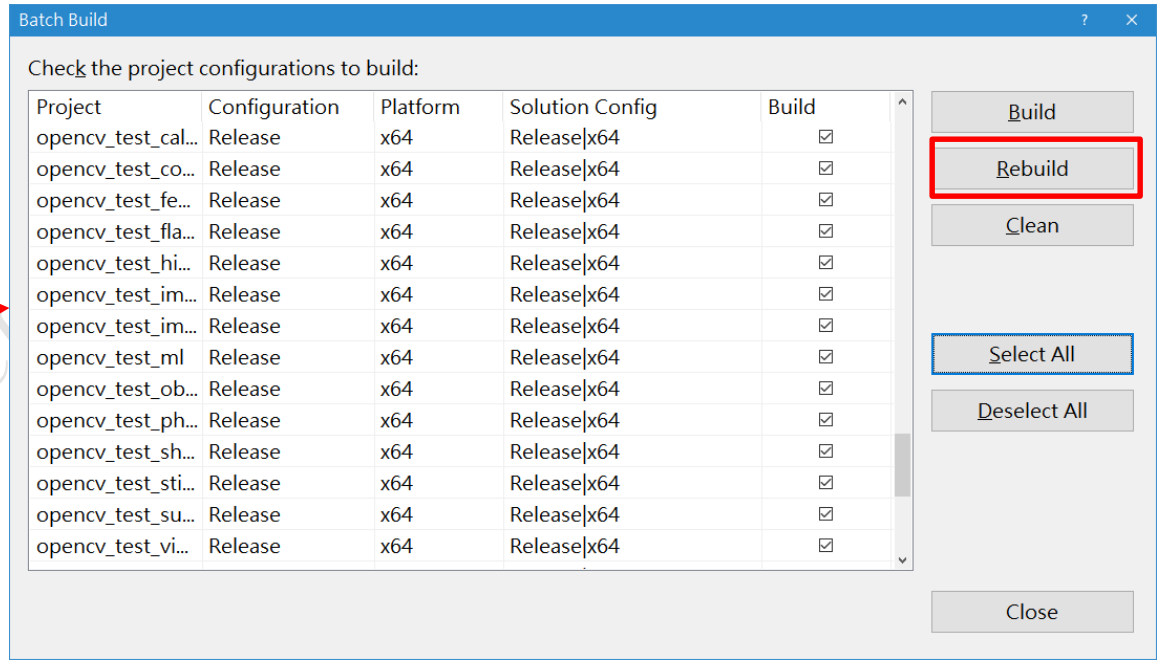
Switch to “Release” (recommended)



Case 2: rebuild into several platforms-cont.

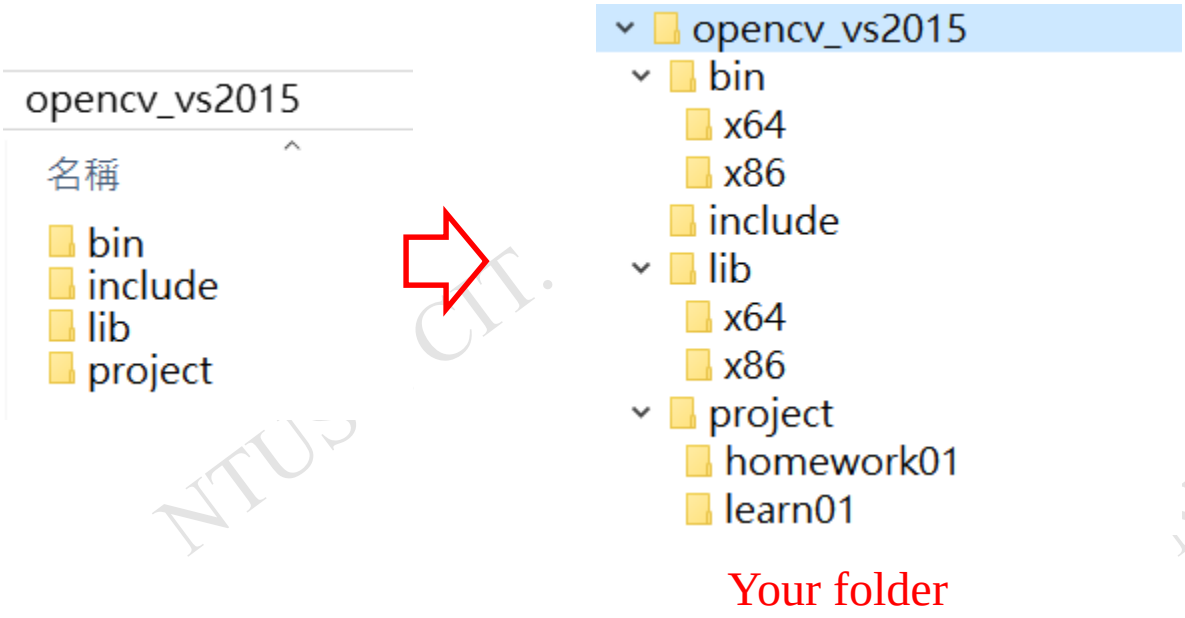
- Complier your project 64bit (for example) / 32bit





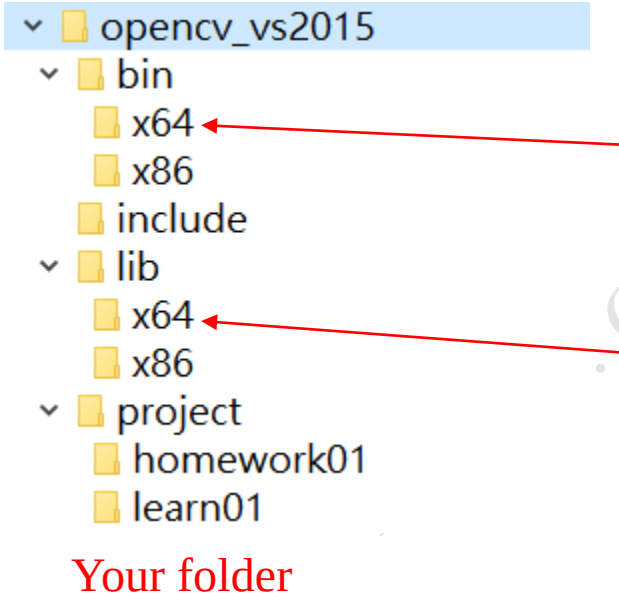
Case 2: rebuild into several platforms-cont.

- Retrieve all necessary files(My habit)
  - Create corresponding folders

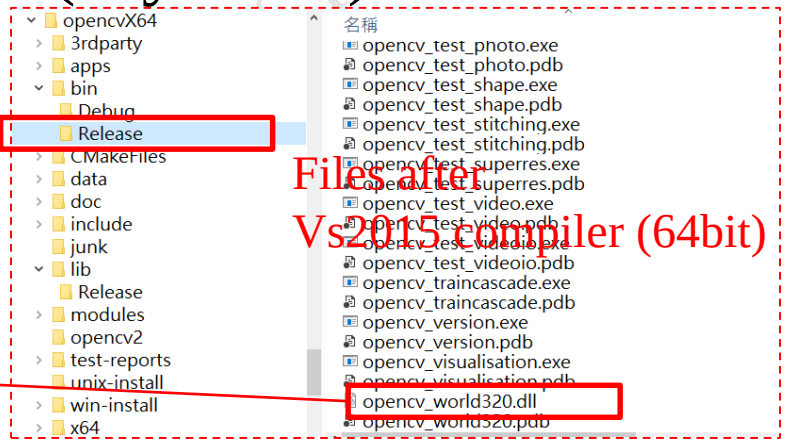


Case 2: rebuild into several platforms-cont.

- Retrieve all necessary files (My habit) x64

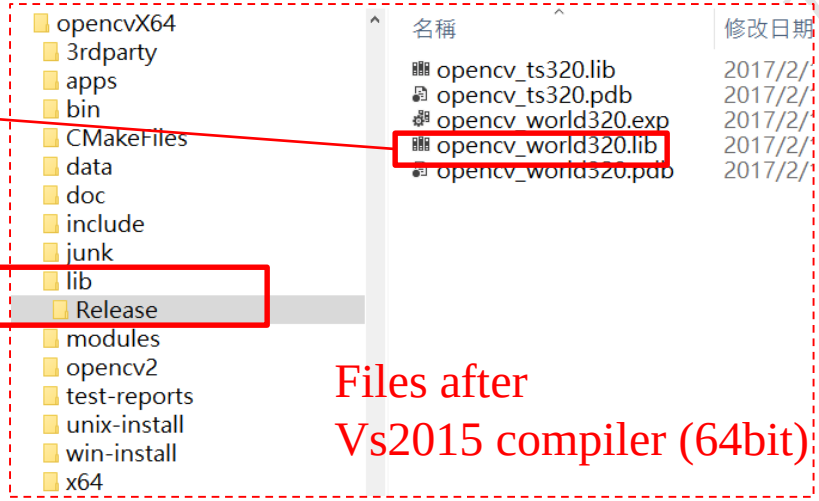


copy



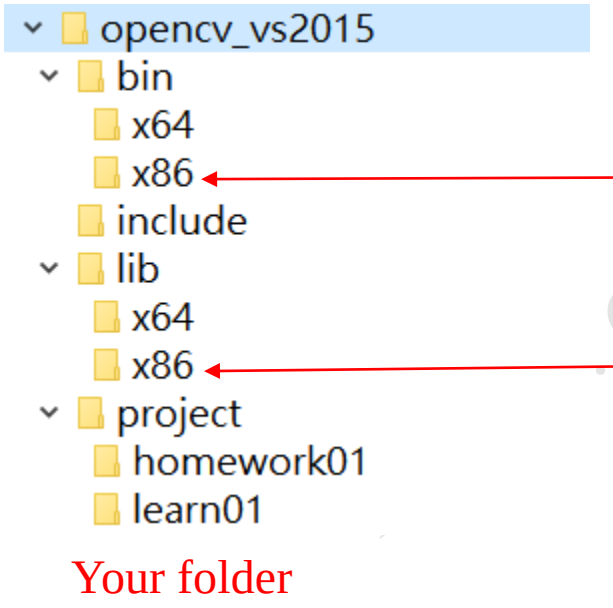
Files after
Vs2015 compiler (64bit)

copy

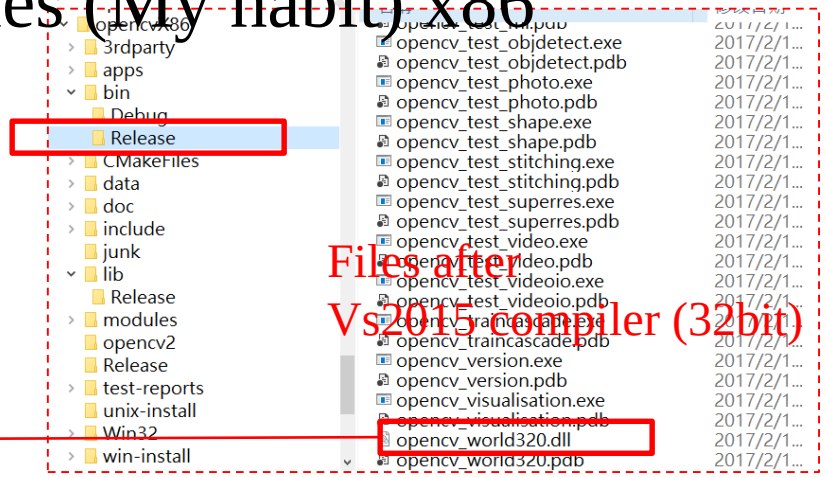


Case 2: rebuild into several platforms-cont.

- Retrieve all necessary files (My habit) x86

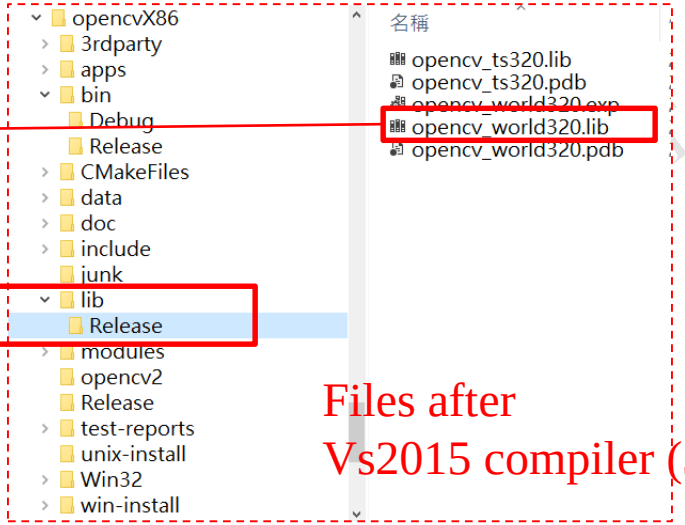


copy



Files after Vs2015 compiler (32bit)

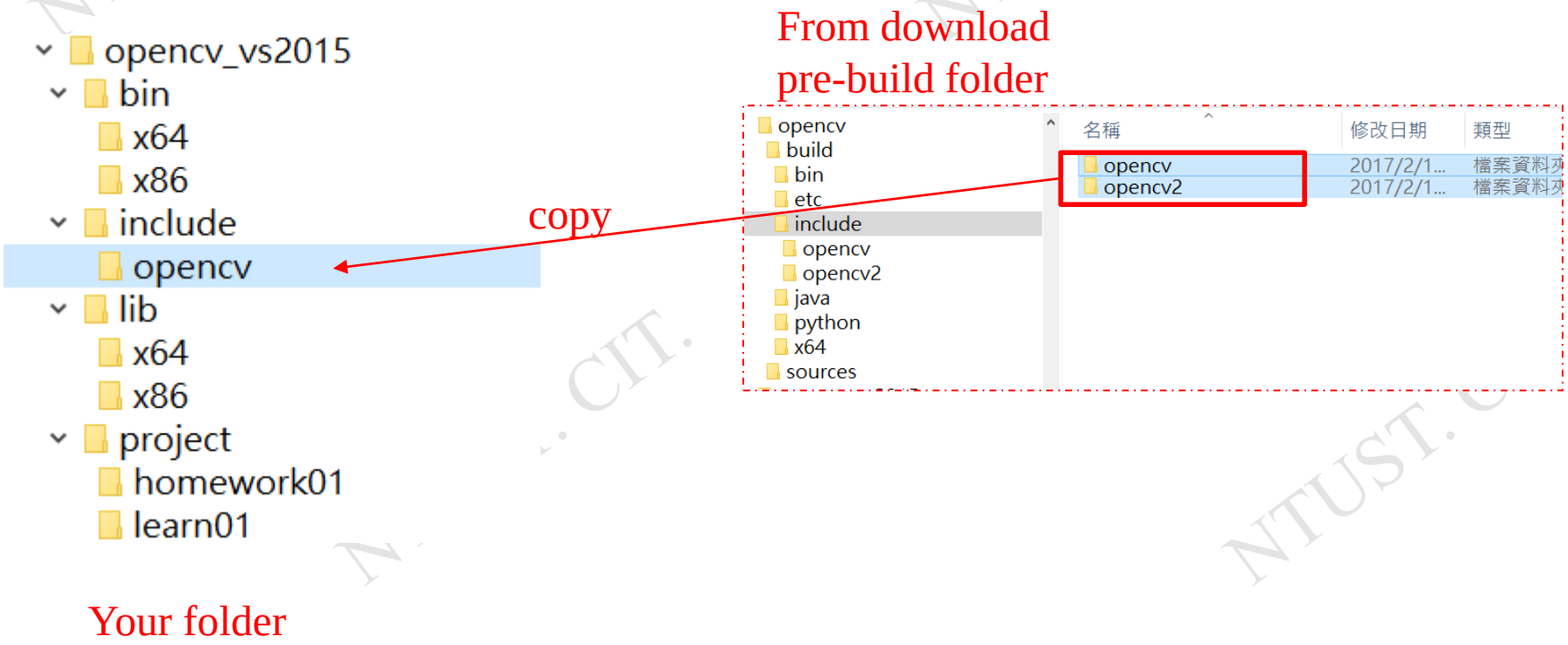
copy



Files after Vs2015 compiler (32bit)

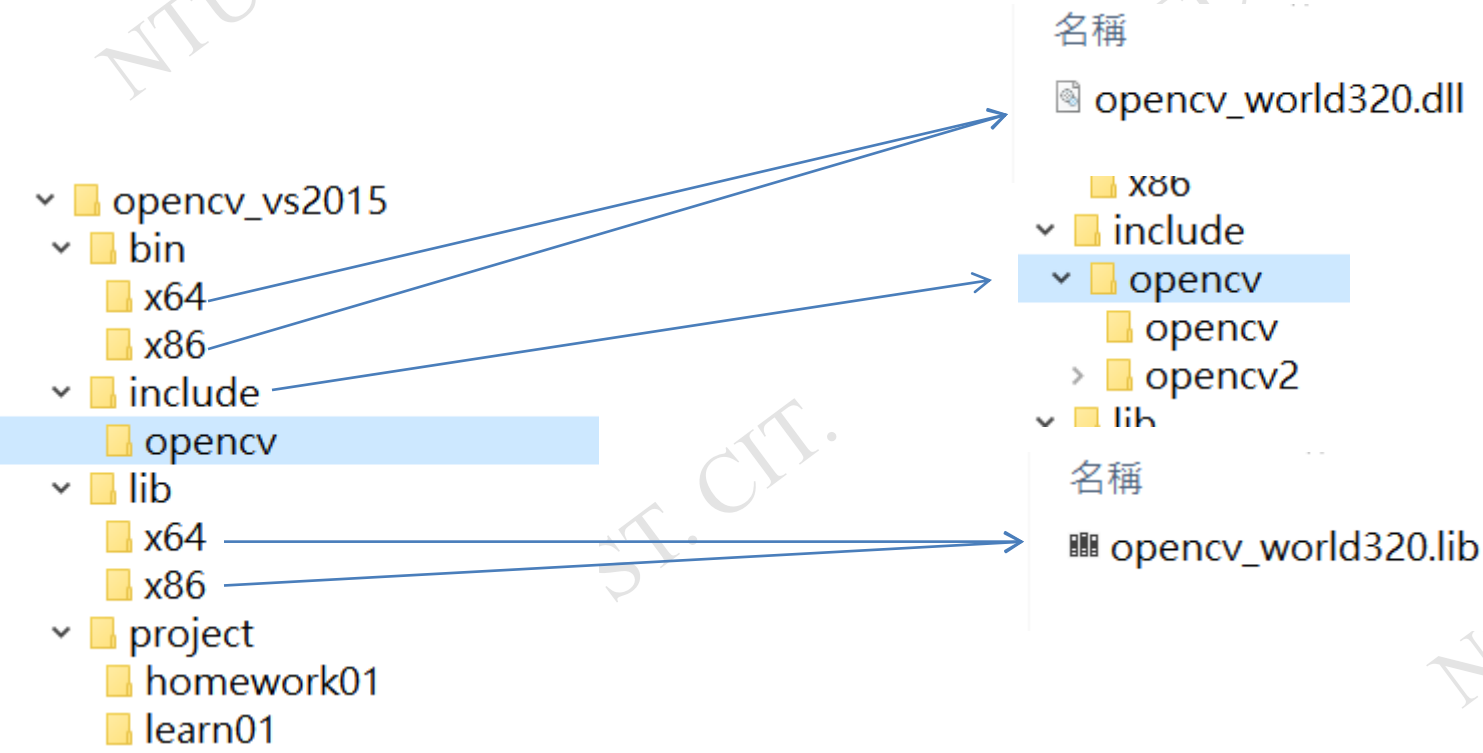
Case 2: rebuild into several platforms-cont.

- Retrieve all necessary files (My habit)
  - Include files (header)



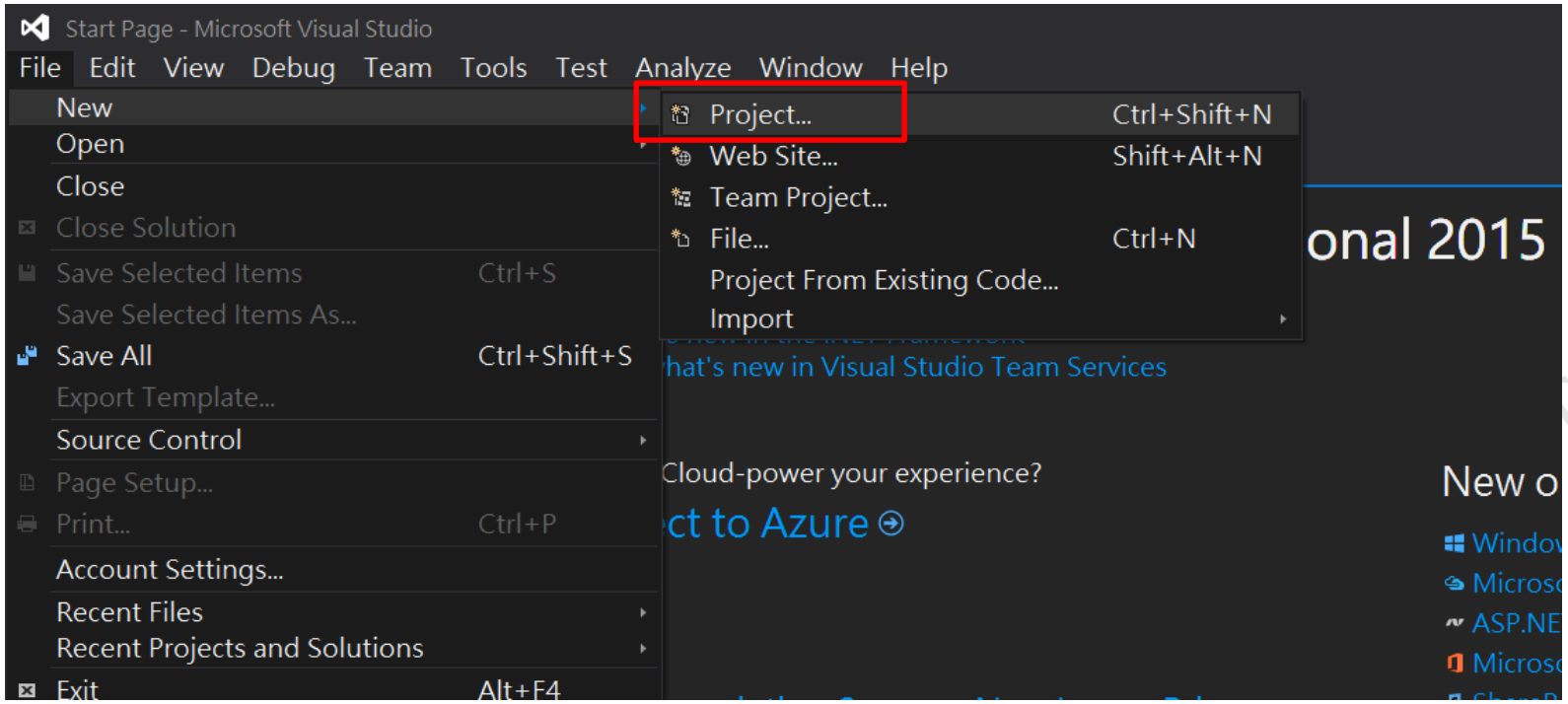
Case 2: rebuild into several platforms-cont.

- What your FILE looks like



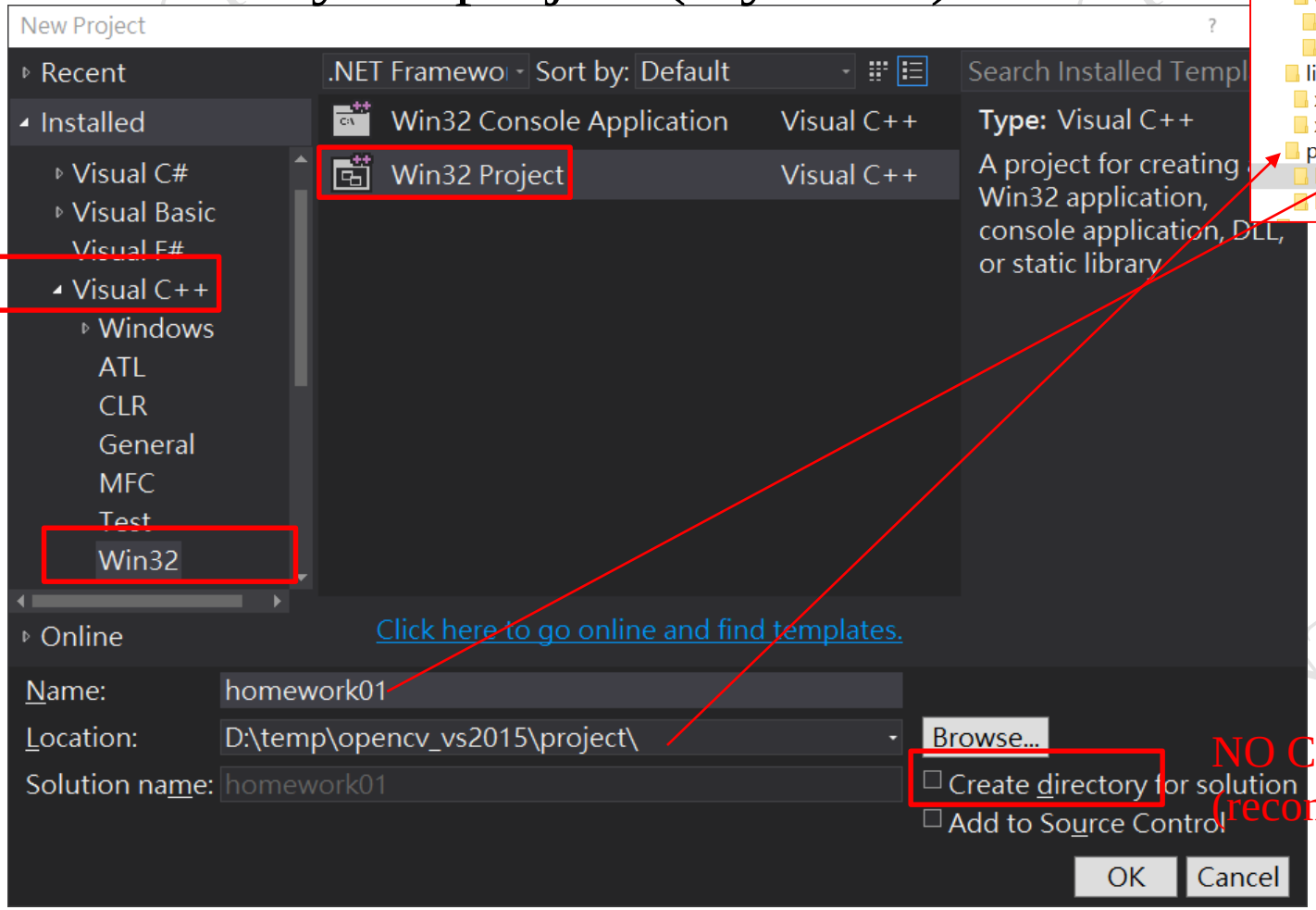
Create a simple project

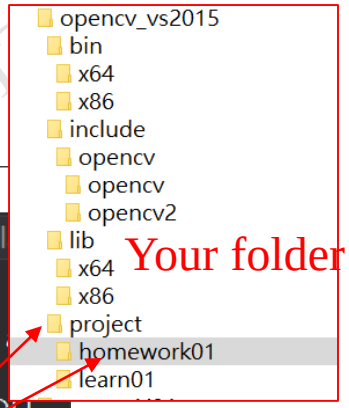
- Open Visual studio 2015
  - New→Project→



Create a simple project—cont.

- Name your project (my habit)

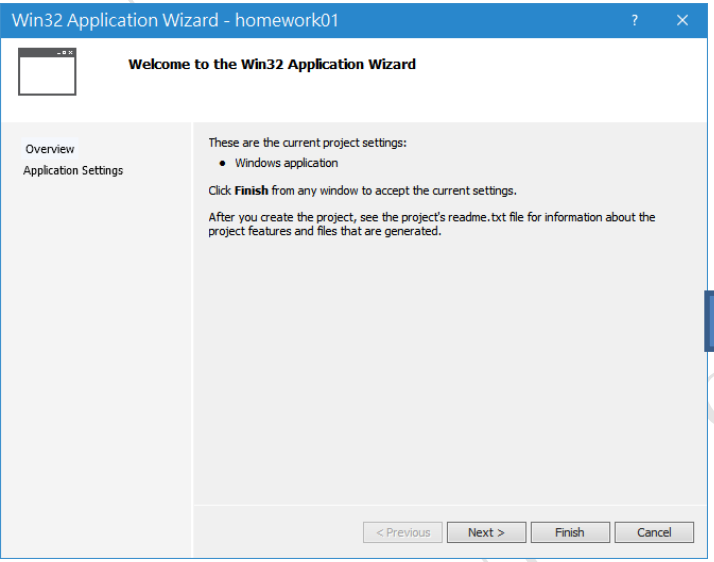


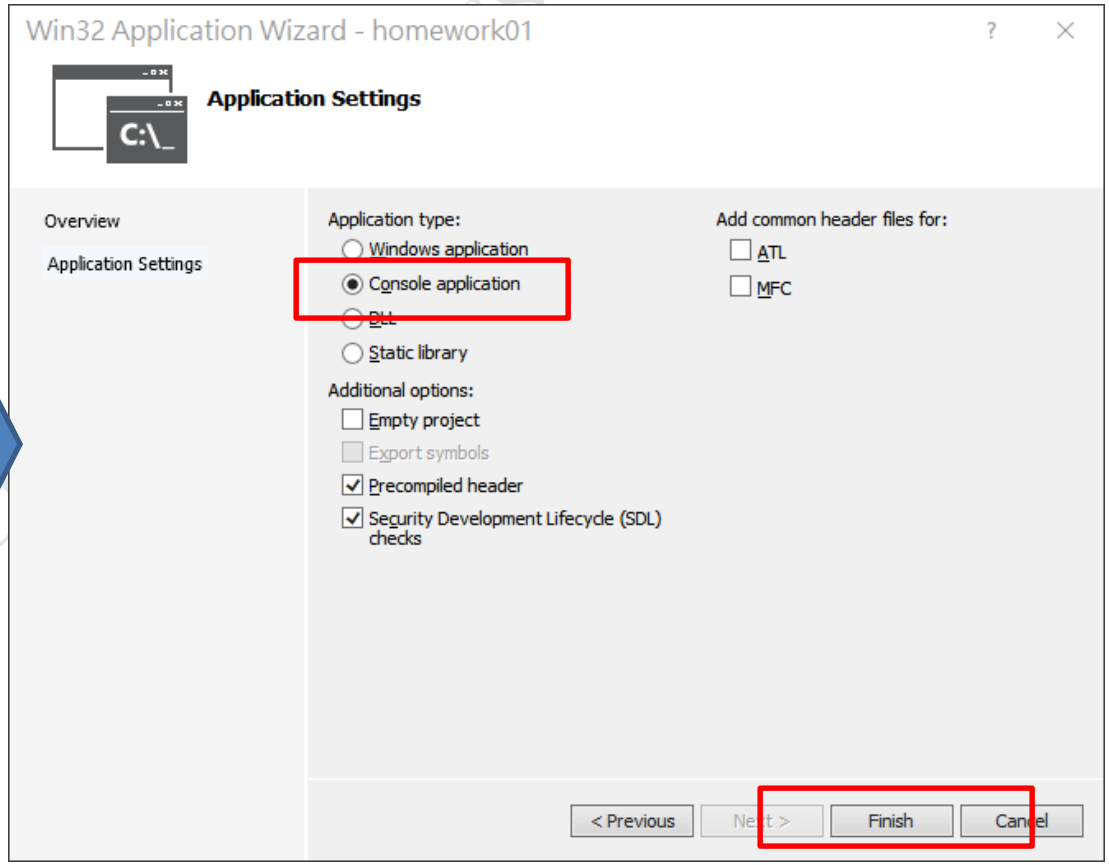


NO Check!
(recommended)

Create a simple project—cont.

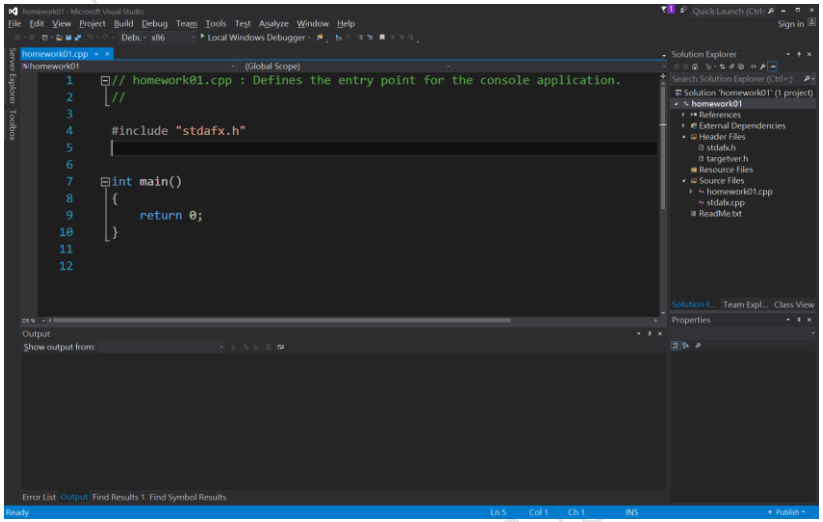
- Next …



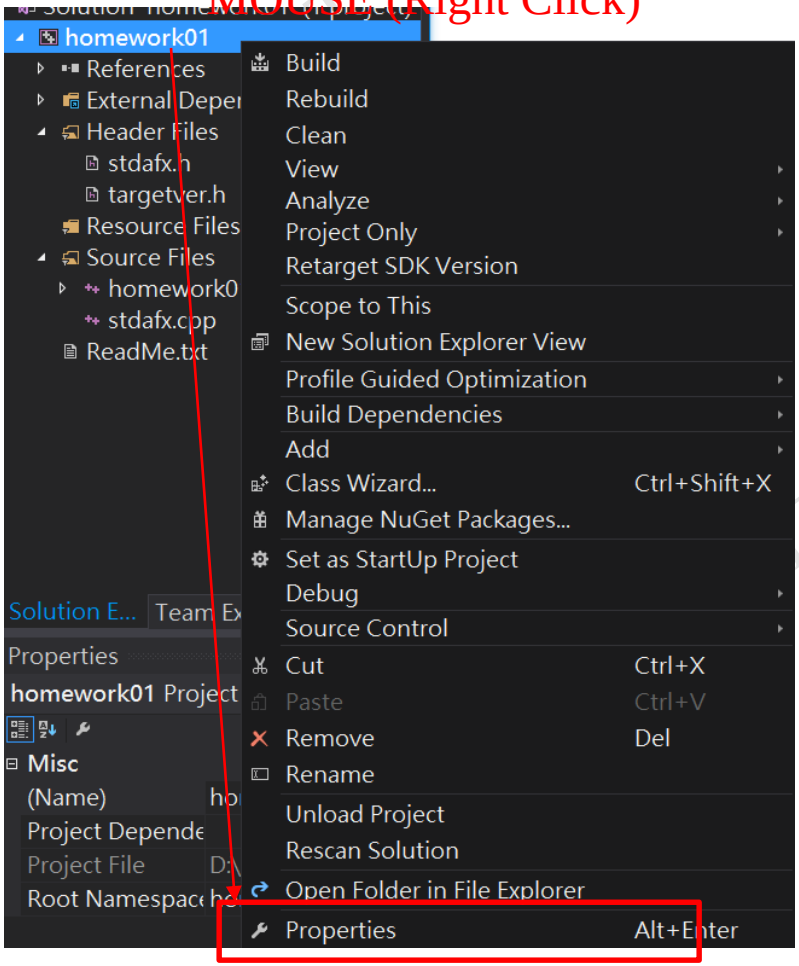


Create a simple project—cont.

- Set the include path



MOUSE (Right Click)

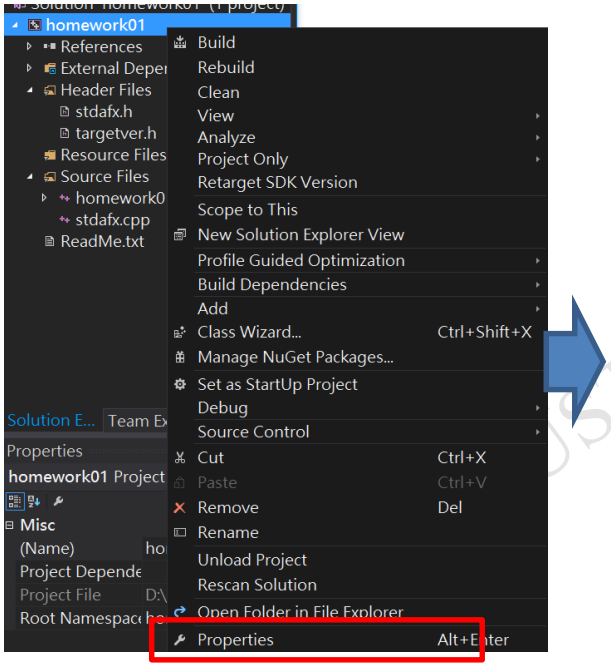


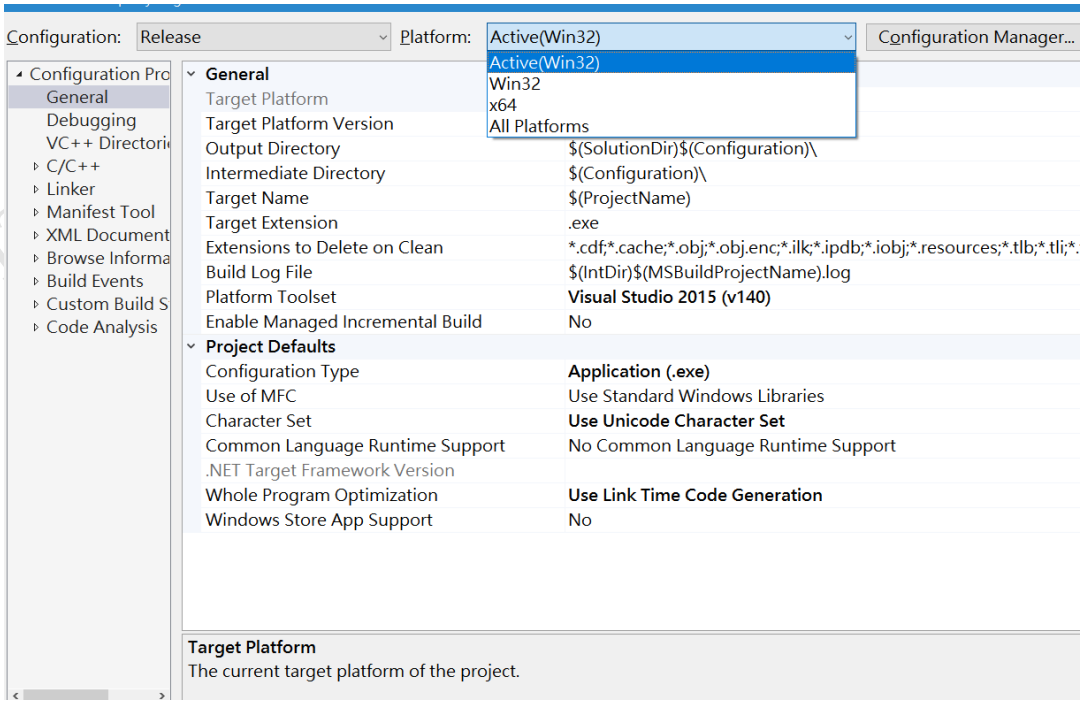
Create a simple project—cont.

- Set the include path

Project	Configuration	Platform	Solution Config
homework01	Debug	Win32	Debug|x86
homework01	Debug	x64	Debug|x64
homework01	Release	Win32	Release|x86
homework01	Release	x64	Release|x64

SET for 4 Configuration States
(Win32 + Release) (Win32 + Debug)
(x64 + Release) (x64 + Debug)

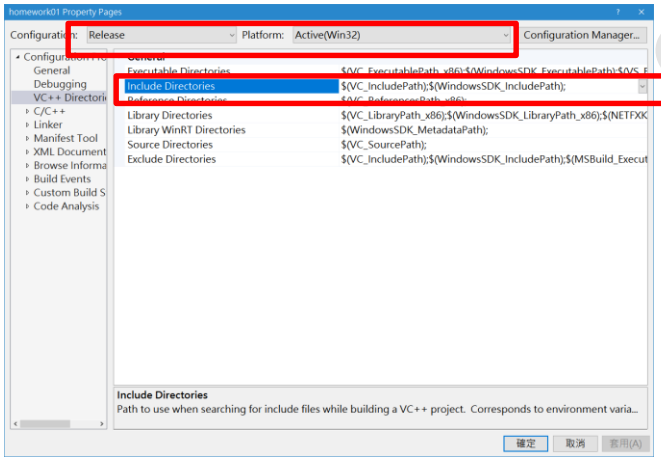


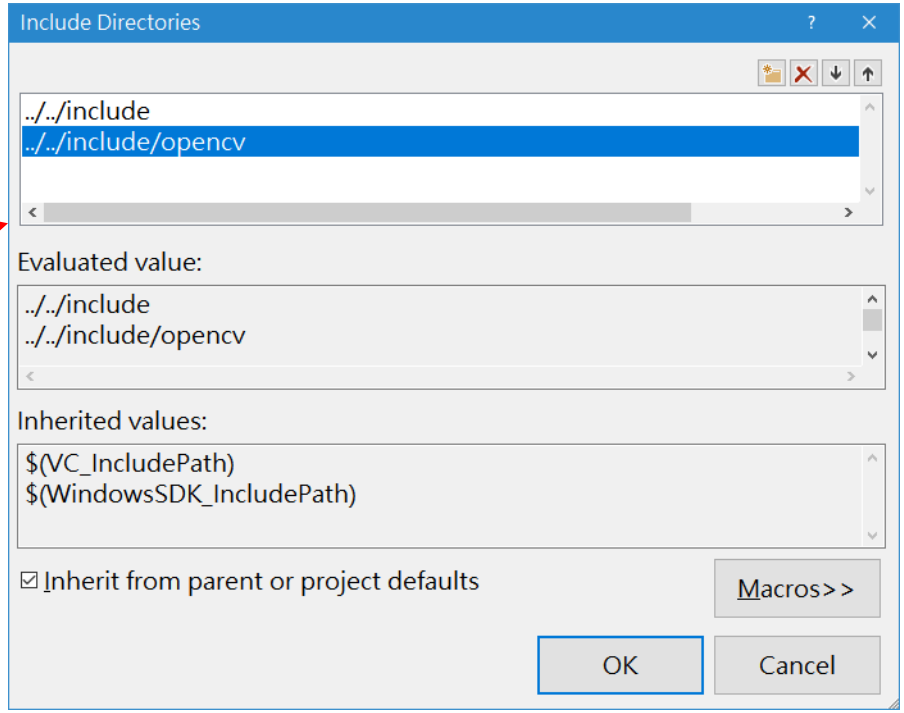


Create a simple project—cont.

- Set the include path: Add
  - ../../include
  - ../../include/opencv

SET for 4 Configuration States
(Win32 + Release) (Win32 + Debug)
(x64 + Release) (x64 + Debug)

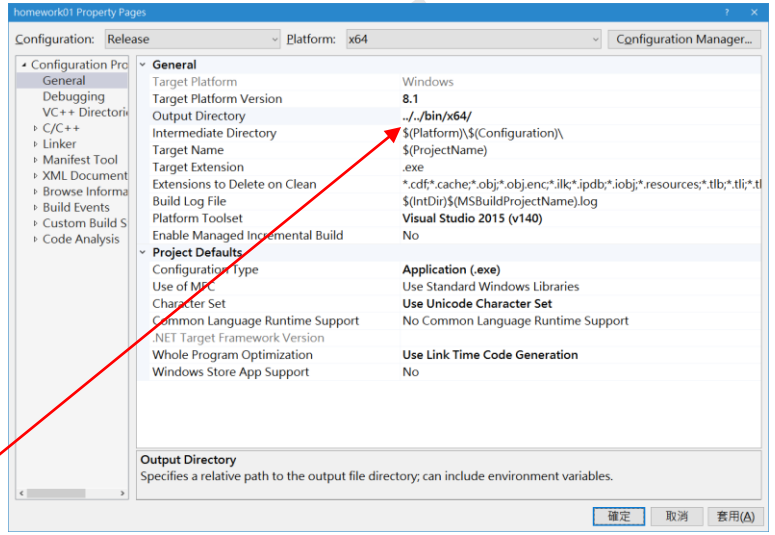


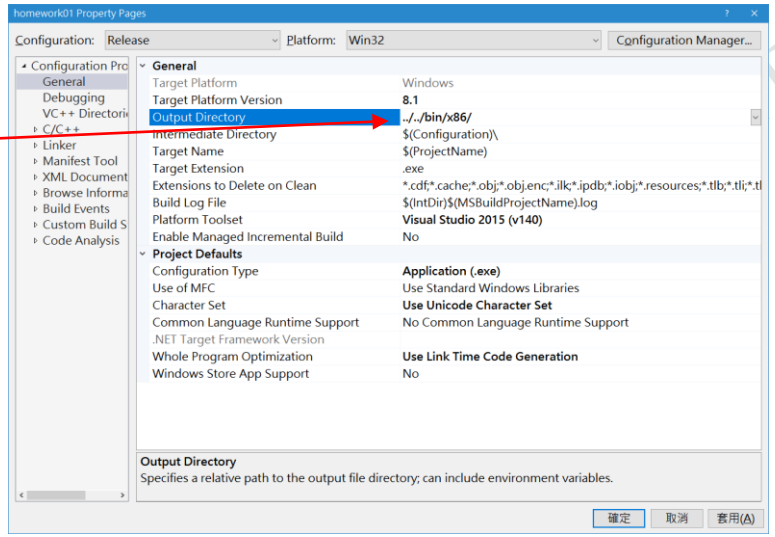


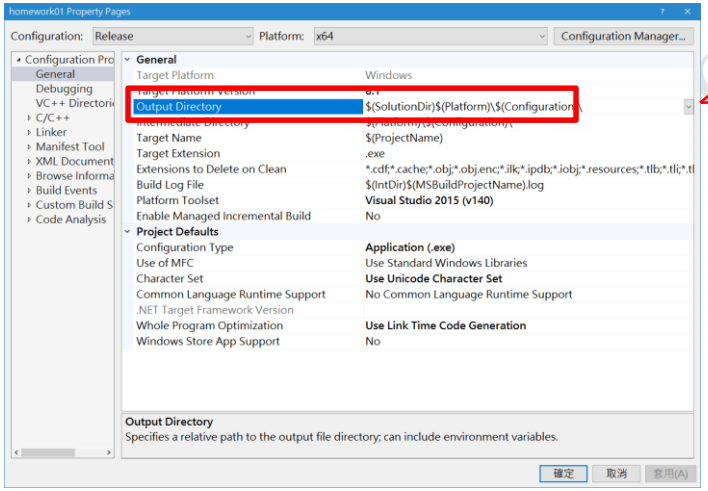
Create a simple project—cont.

- Set output directory
  - For x86 → ../../bin/x86/
  - For x64 → ../../bin/x64/

SET for 4 Configuration States
(Win32 + Release) (Win32 + Debug)
(x64 + Release) (x64 + Debug)



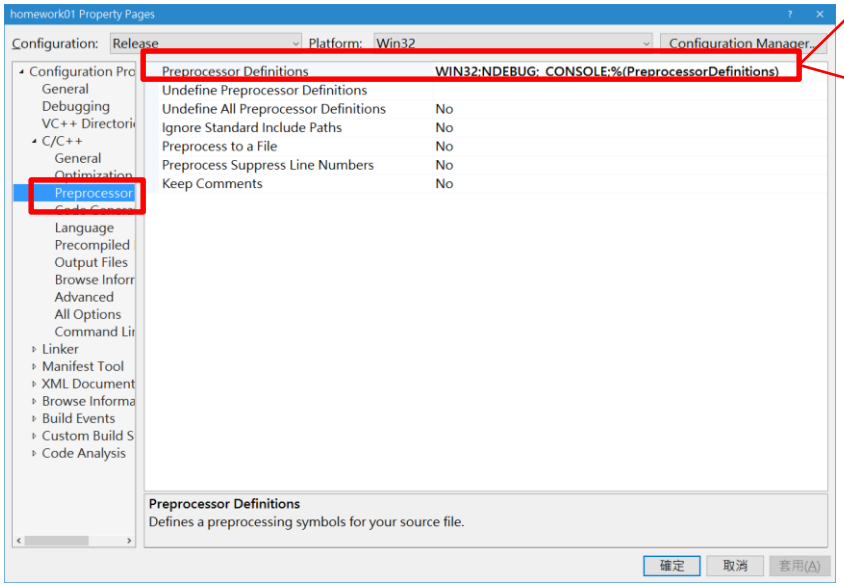


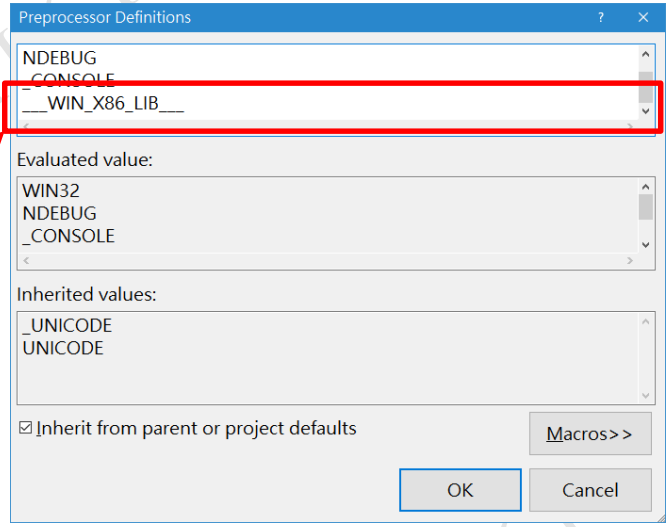


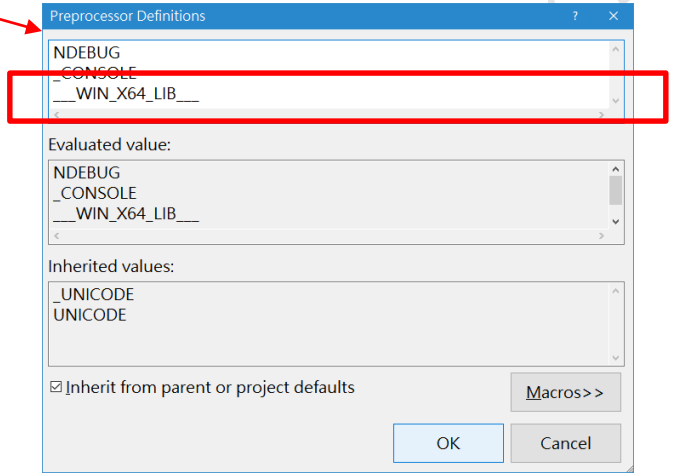
Create a simple project—cont.

- Add preprocessor (my habit)
  - For x86→___WIN_X86_LIB___
  - For x64→___WIN_X64_LIB___

SET for 4 Configuration States
(Win32 + Release) (Win32 + Debug)
(x64 + Release) (x64 + Debug)



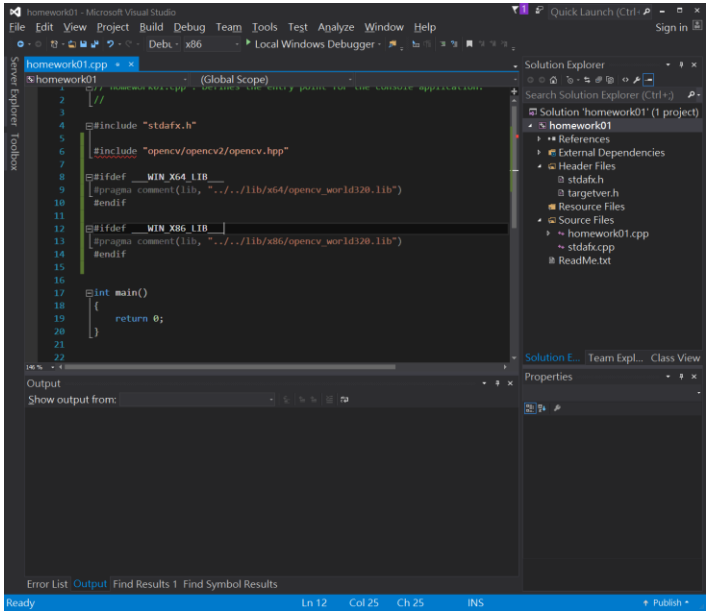




Create a simple project—cont.

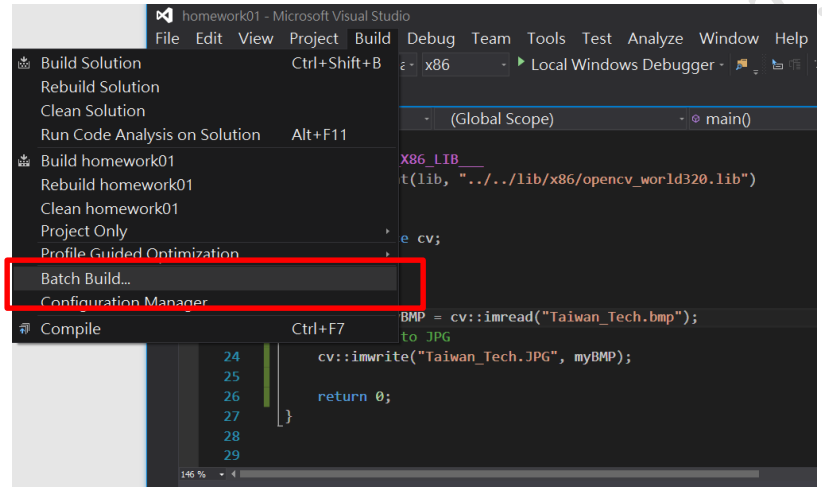
- Add following txt in *.cpp

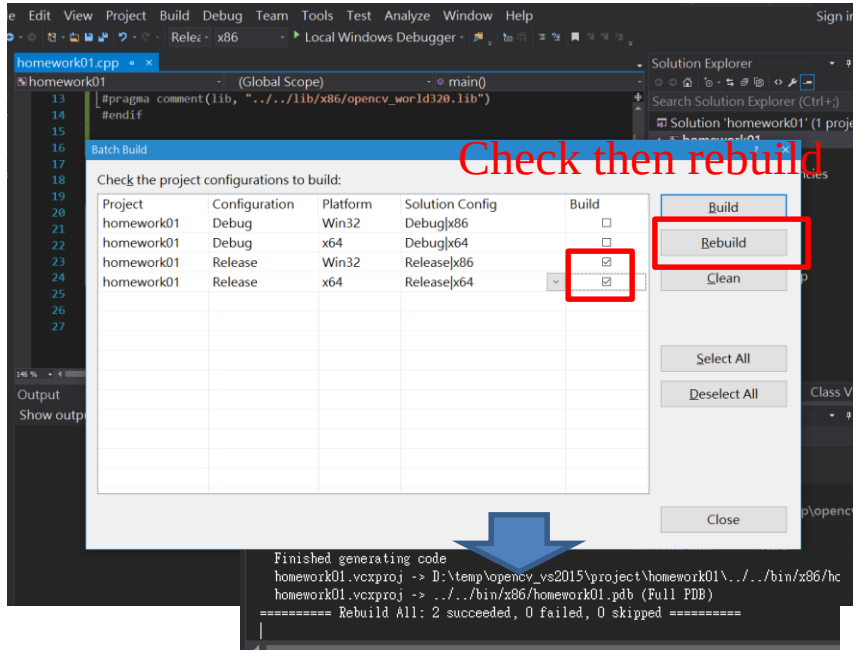
```
#include <stdlib.h>
#include <stdio.h>
#include <windows.h>
#include "opencv/opencv2/opencv.hpp"

#ifdef ___WIN_X64_LIB___
#pragma comment(lib, "../../lib/x64/opencv_world320.lib")
#endif

#ifdef ___WIN_X86_LIB___
#pragma comment(lib, "../../lib/x86/opencv_world320.lib")
#endif

using namespace cv;
```



Compiler your work

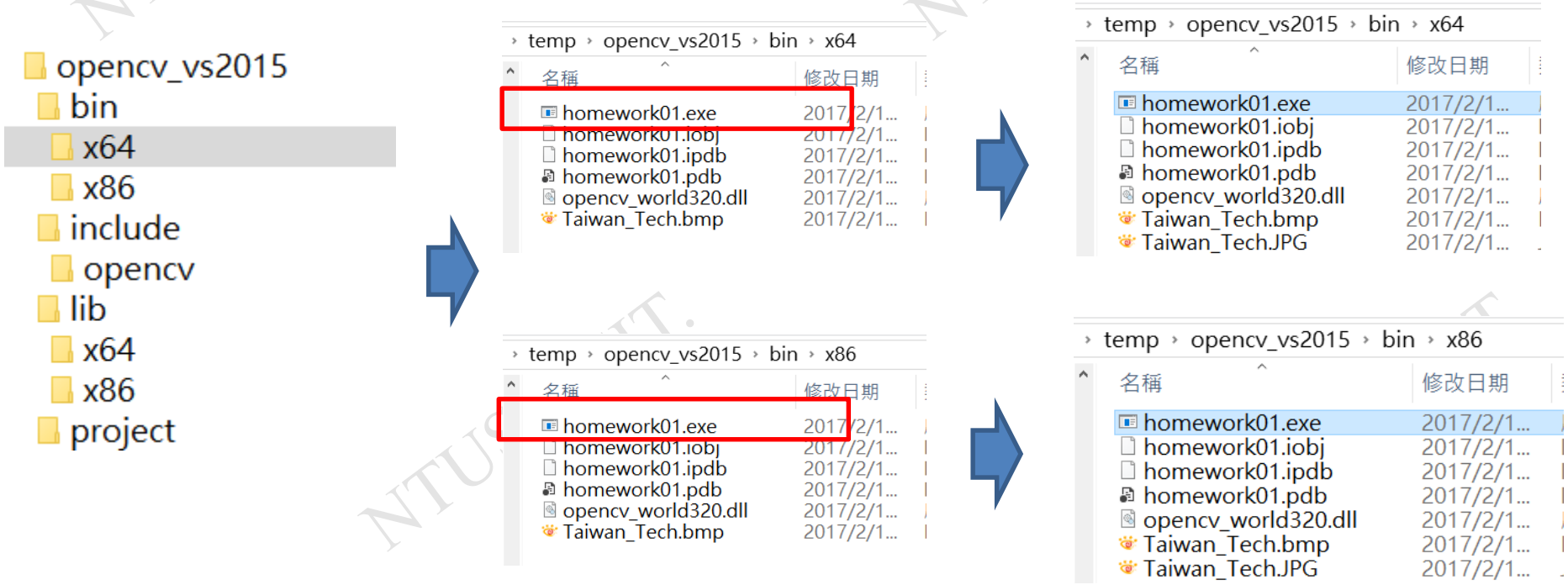
```
//Example
int main()
{
        cv::Mat myBMP = cv::imread("Taiwan_Tech.bmp");
        //convert to JPG
        cv::imwrite("Taiwan_Tech.JPG", myBMP);
        return 0;
}
```





Compiler your work—cont.

- Go to bin directory and execute “homework01.exe”



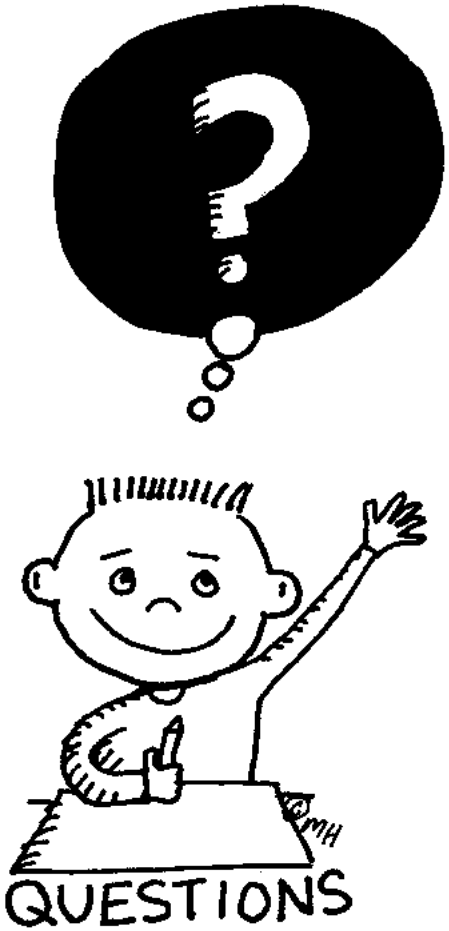


Figure from : http://www.gabankruptcylawyersnetwork.com/files/2014/08/question.gif