Data Structures
and Object-oriented Programming

Programming Assignment One Instruction

To Teaching Assistants: Please check the information at the top bar is correct.

Table of Content

1. Instruction

Intended Learning Outcome: You develop two components in a system that draws curves.

You must use the assignment template to implement your programs. **You do not receive any score points if you do not use the template.**

In this assignment, you implement some classes which are integrated in a large system. You can see the files in the folder **00_StudentWork**. These two classes are:

- CURVE_FUNCTION
- GRAPH_MANAGER

Write your programs in Visual Studio 2019 on the .NET platform.

The compiler version must be v142. The **project file name** is SOGLFramework.sln.

We will rebuild your program in the Release mode and check your program.

How to run the program? You can find the executable file enjoy_programming.exe

in ./bin/Release

The demo program may have bugs. These bugs are useful for you to understand that if we do not do a good job to check our programs thoroughly, all these bugs cannot be fixed. However, as you can see, you can still run the program without a severe runtime error. **Thus, please follow the instructions. If the instructions are not clear to you, send us an email to clarify the issues. Could you find any bugs in the demo program? Do you know how to fix the bugs?**

2. Submission:

I. Change the folder name to ID_Name, where ID is your student ID and Name is your name. **Zip and upload the entire folder of the source code** to E3 platform before the deadline. If your student ID is 012345678 and your name is Mary, then the folder name must be 012345678_Mary.

II. You must demo your work to our TAs in the lab session.

III. If you cannot demo your program(s), your score is zero.

3. Penalties: No Late submission.

Cheating: you will be received a score of zero, e.g., borrowing your source code to others or/and copying others' source code.

4. Requirement Specification

Use double to define a variable which is a floating point number. All the calculations should be done in double precision. Don't use float. Show the value of a floating point number up to 8 decimal digits (if any).

A. Basic tasks.

I. Write your information (name, ID, email address) in the header file mySystemApp.h

II. Set the macro `STUDENT_INFO` in mySystem.cpp correctly. So that the top bar of the window shows the subsystem name, your name and student ID, as shown in the following figure.

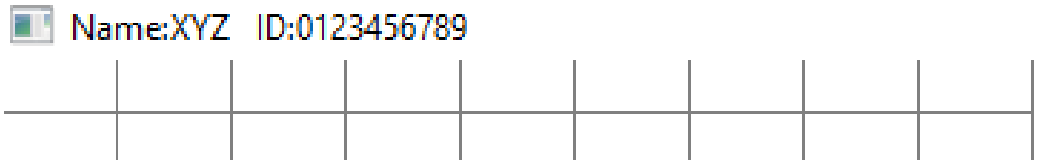


Figure 1.1: The top bar shows the student name and ID at the beginning.

III. Press 's' or 'S' to show your student information: **date**, student ID, name, and email address. `showMyStudentInfo_2024( ) in mySystemApp.cpp`

```
Step 1: initGL( argc, argv )
Step 2: Create an app
create Basic_OpenGLApp(int argc, char *argv[], std::string p_Title)
Basic_OpenGLApp::Basic_OpenGLApp
Basic_OpenGLApp::Basic_OpenGLApp:1
BGN init()
END init()
void Basic_OpenGLApp::showInfo( ) const
m_WinID:1
Step 3
Step 4: go_openGL
OpenGL_APP: WinID:1
OpenGL_APP::idle( )
*******************************
Date:2023/03/14
Student ID:
Student Name:
Student Email:
*******************************
```

Figure 1.2: Press 's' or 'S' to show student information on the console window.

Items I, II, III, and IV must be done. If not, your score is zero.

Key usages.

F1: Perform Graph Manager. The number of curves is the minimum number of curves.

F2: Perform Graph Manager. The number of curves is the maximum number of curves.

s, S: show the student information on the console window.

5. Assignment Template conversion from Visual Studio 2010 to Visual Studio 2019 (for example)

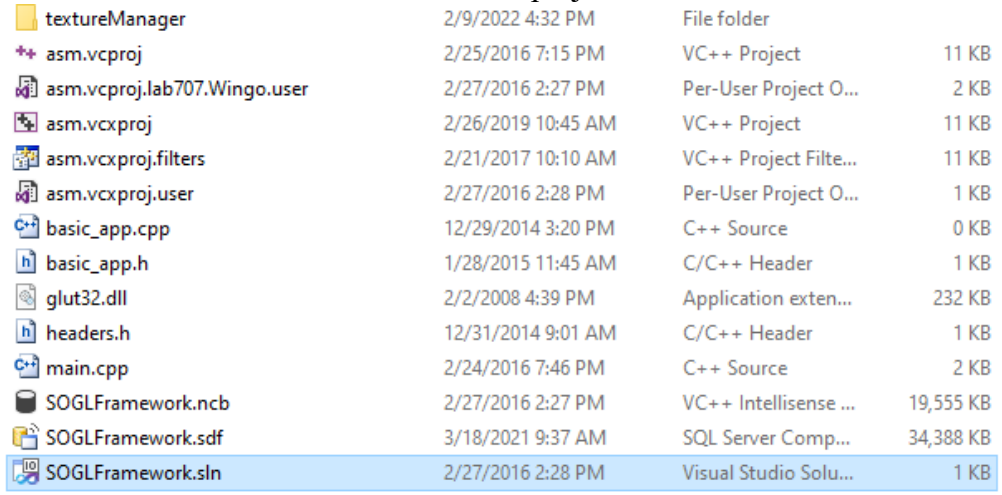
We may encounter that some programs are implemented on some old platforms. How do we use the new platforms to compile and build such programs? This exercise is important for you to understand that if the programs are well written, we can easily convert them so that we can use a new platform to handle the programs.

Follow the steps to convert the assignment template written on Visual Studio 2010 to Visual Studio 2019 and use **Platform Tool v142**.

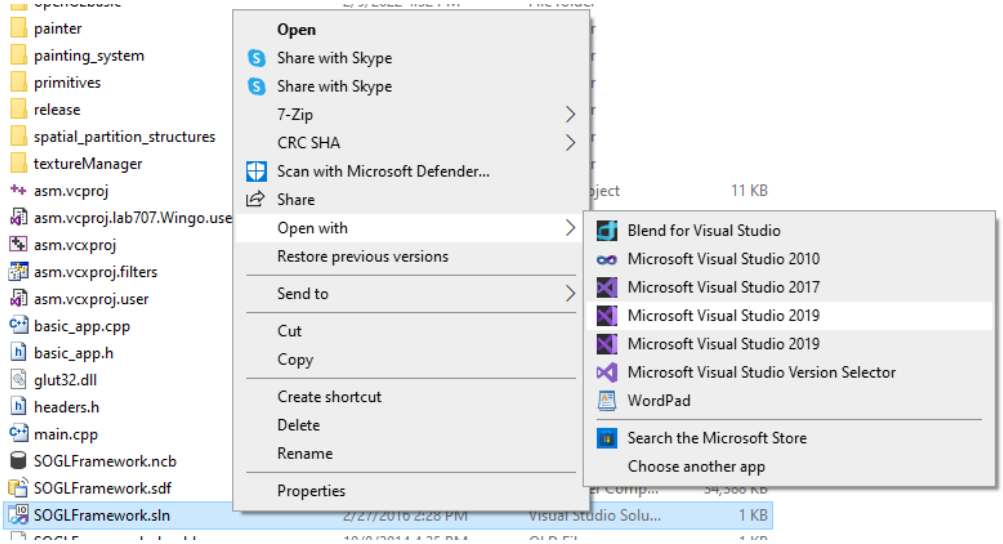
You may encounter different situations when you convert the assignment template. Here, we show two situations that you may encounter: (1) set the version to be upgraded; (2) set the compiler version manually in Visual Studio.

5.1 Situation One: In this case, Visual Studio will ask you to upgrade the platform to the newest version and compiler version to v142.

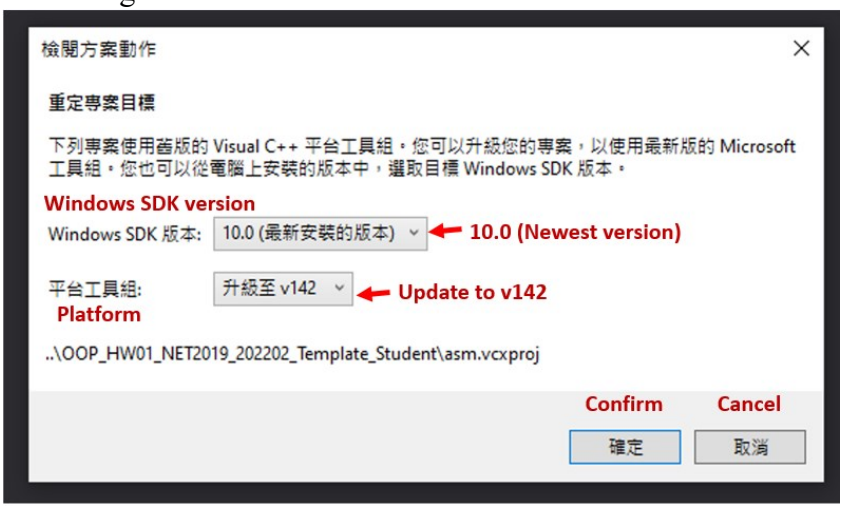
(a) In our case, we need to find out the main project file which is SOGLFramework.sln



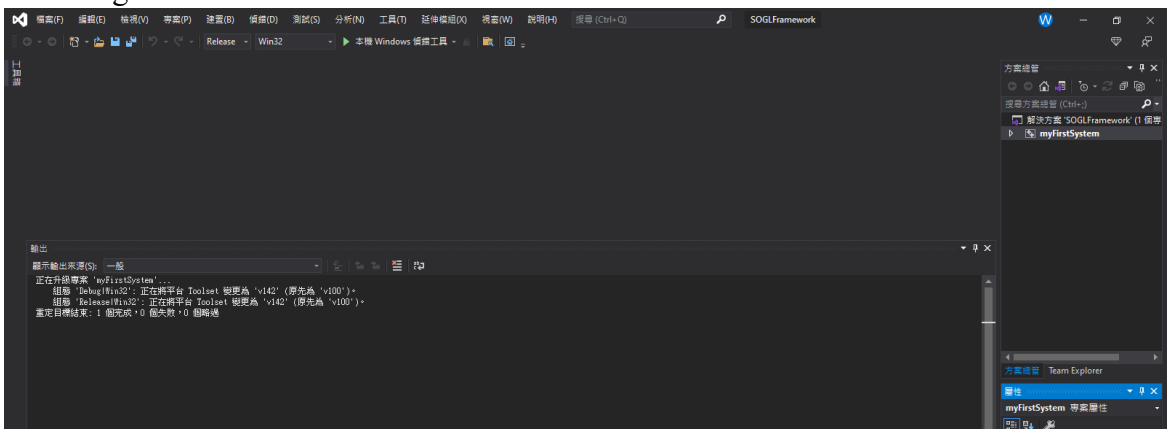
(b) Use the mouse and right click on the project file SOGLFramework.sln and then select “Open with VS 2019”



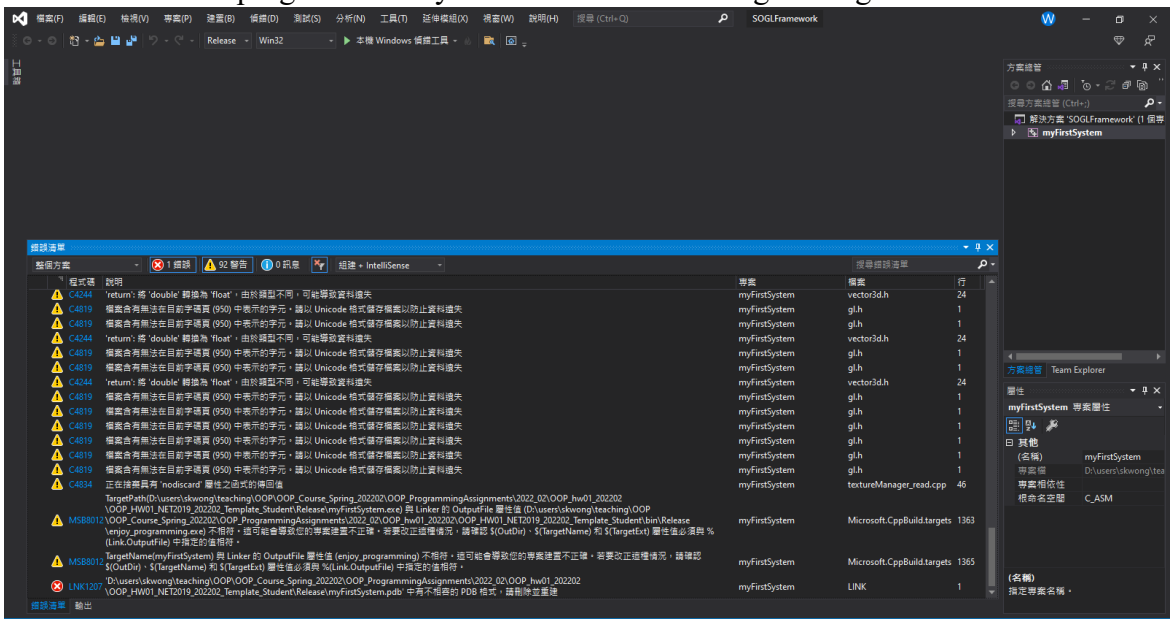
(c) Then you will see a dialogue box as follows:

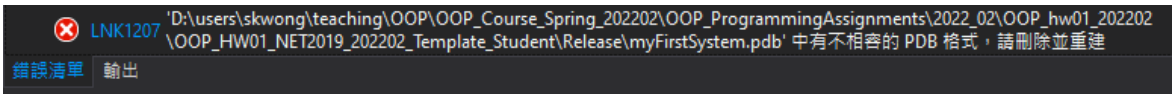


(d) Make sure that you set the compiler version to v142. Then press Confirm. Then you will the following window

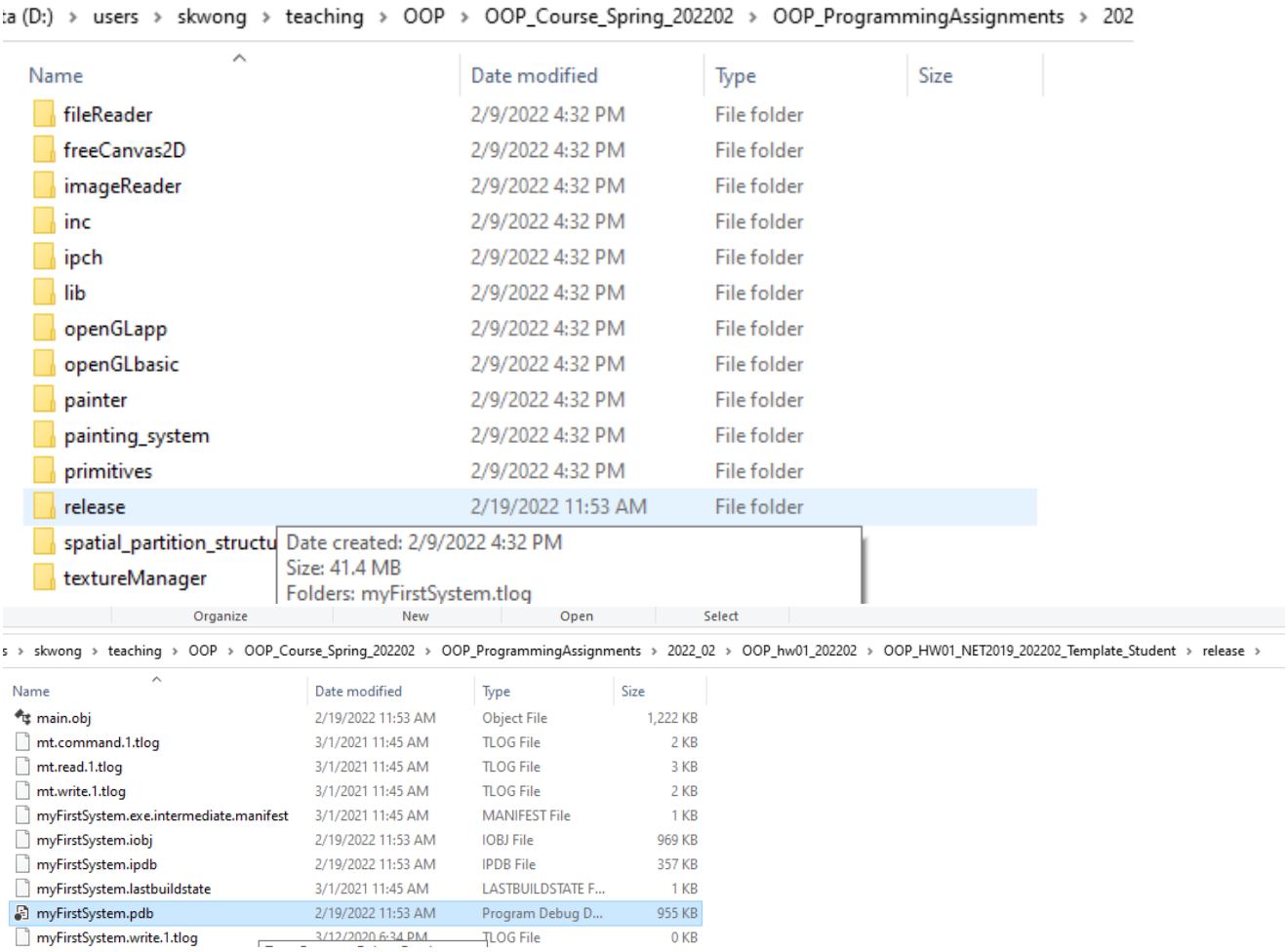


(e) Now press ALT + B + B to build the program. As the template program is quite large, it will take a while to build the program. Then you will see the following messages. There is a severe error.

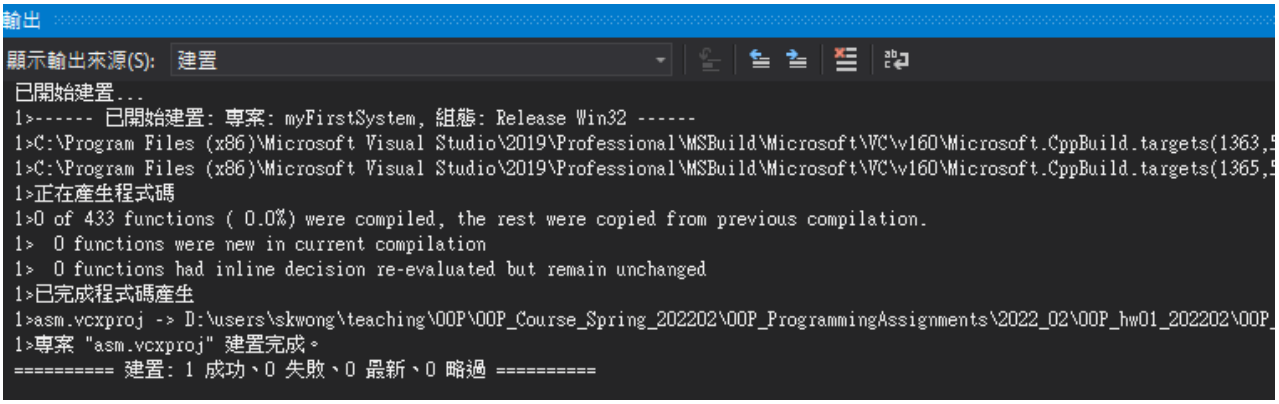




The last message tells you that we need to delete the file: **myFirstSystem.pdb** in the folder .\Release
Go to the folder and delete the file.



Delete the file: myFirstSystem.pdb.
After that, go back to Visual Studio and build the program again, by pressing ALT+B+B. We are done.



The message says that the program is built successfully. Well done!

So you see how easy it is to convert a program written on an old platform to a new platform. However, in real life if the programs are not “well-written”, we will encounter a lot of problems. We may need to rewrite a large portion of such a program. Luckily, we can convert our assignment template so easily in our case.

5.2 Situation Two: In this case, you open the project and enter Visual Studio. You need to set the compiler version to v142 manually.

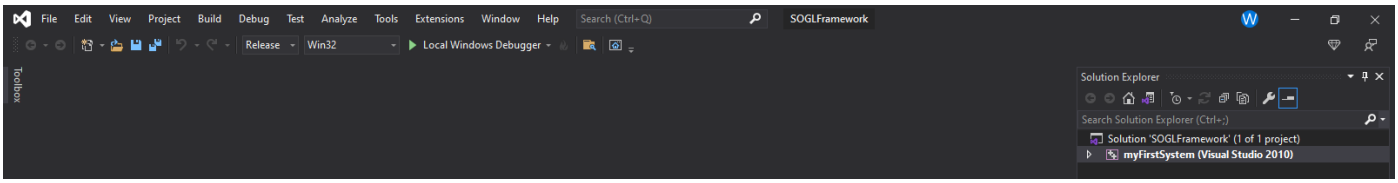


Figure 5.2.1: We are in Visual Studio 2019. On the right side, you can see the version.

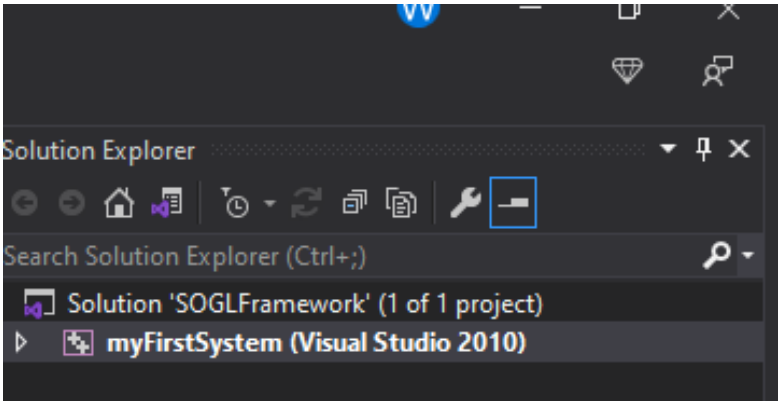


Figure 5.2.2: The VS version 2010.

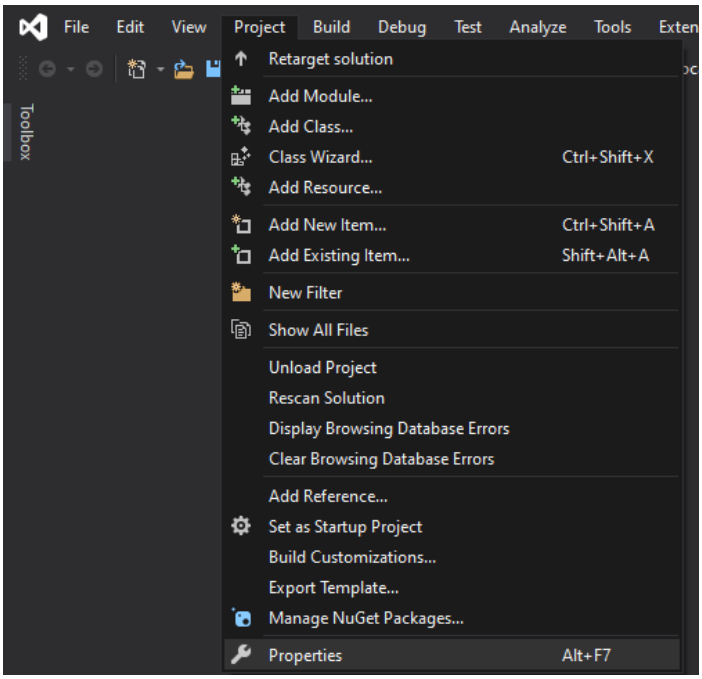


Figure 5.2.3: Now, go to Project->Project properties

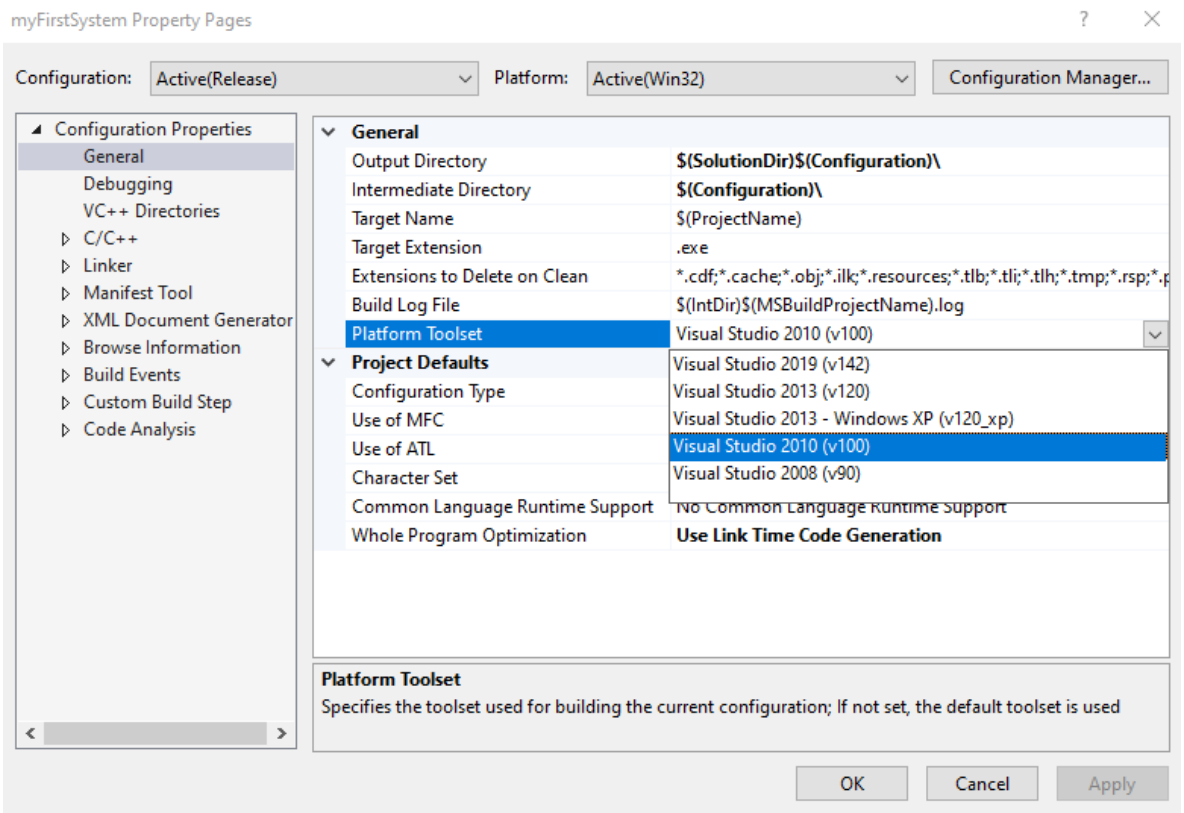


Figure 5.2.4: Now, in the project properties, on the left panel, go to: Configuration properties->General. In the dialogue of General, select Platform Toolset and set it to v142.

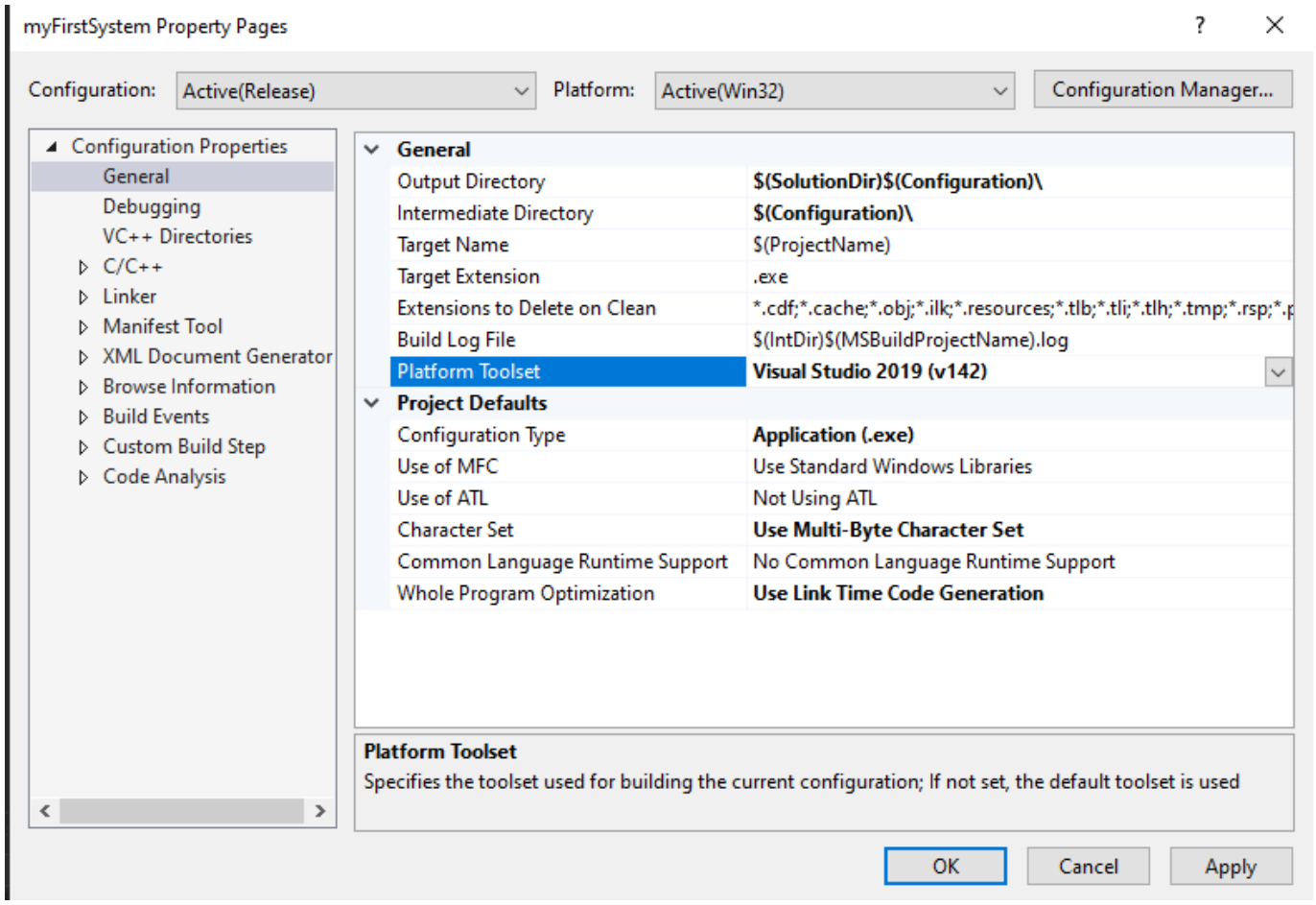


Figure 5.2.5: After setting the Platform Toolset to v142, click "Apply" and then "OK".

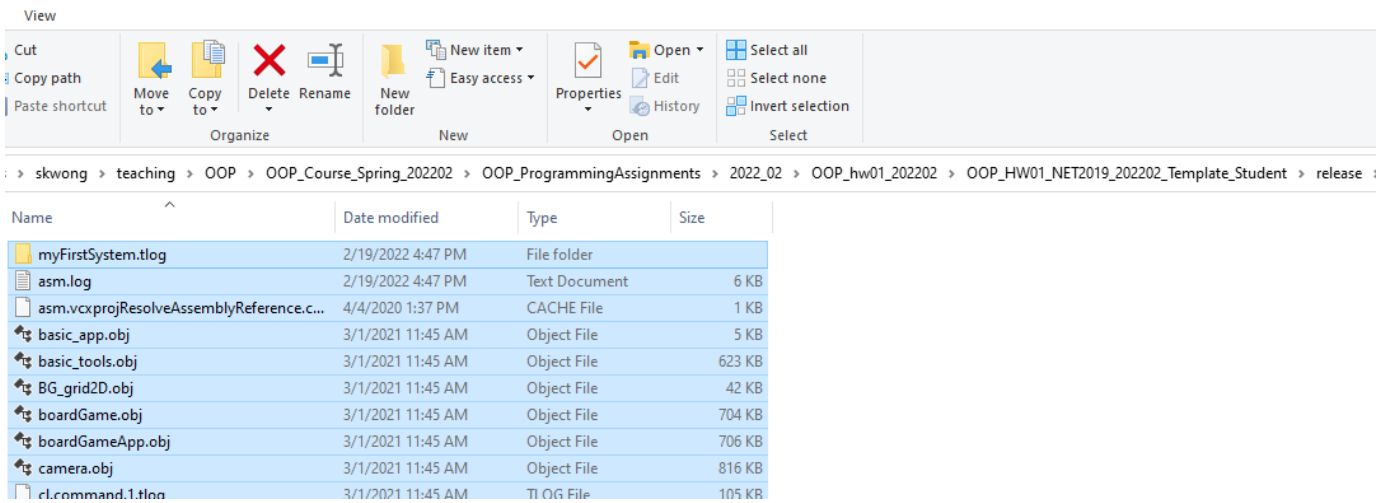


Figure 5.2.6: Now, delete all the .obj files and the pdb file in the folder ./Release.

In Visual Studio, press ALT+B+B to build the program. Make sure that a message shows "build successfully". Yea, you can rebuild (CTRL+ALT+F7) everything too :D

6. Subsystem: Graph Manager

You implement a graph manager. The system will use the graph manager to create an object. After that, the system will call the functions of the object to draw curves with different types. A curve is drawn for a given x interval [xMin, xMax]. The system can draw a lot of curves at the same time. However, a maximum number of curves is set so that you can interact with the system by using the keyboard input.

You need to implement two classes:

- CURVE_FUNCTION
- GRAPH_MANAGER

You must modify these four files

- mySystem_Curve.h and mySystem_Curve.cpp
- mySystem_GraphManager.h and mySystem_GraphManager.cpp

to finish this assignment.

The workflow of the system is as follows:

(1) Instantiate an object of GRAPH_MANAGER, call mGraphManager.
(2) mGraphManager instantiates an array of CURVE_FUNCTION objects, namely mCurves. Each object generates one curve based on two parameters c and d, and the interval of x, and other information.
(3) For every step, mGraphManager extracts the information of each object of mCurves (or each curve). For each curve, the system will get the number of sample points, interval of x, and other data. Then the system draws the curve based on the sample points.

You implement CURVE_FUNCTION so that an object of CURVE_FUNCTION is able to perform tasks for drawing a curve, including the following:

- setNumOfSamplePoints. Return the number of the sample points, N. The system will compute N sample points to draw the curve.
- getIntervalOfX. Return the interval of x, i.e., [minimum x, maximum x]. This interval is important for the system to compute the sample points. Then the sample points are drawn as line segments.
- getValue. Compute y value of a function for a given x value. This pair (x,y) is a sample point.
- getBoundaryPoint. Return one boundary point of the curve. There are two boundary points which are the leftmost point and rightmost point of the curve.
- getExtremePoints. Compute the extreme sample points of the curve. For an interior sample point (x,y), there must have one point pL on the left and another point pR on the right. An interior sample point (x,y) is an extreme sample point if one of the following conditions is satisfied:
  - (1) $y > pL.y$ and $y > pR.y$
  - (2) $y < pL.y$ and $y < pR.y$

The x-coordinates of the sample points pArr are computed as follows:

pArr[i].x = xMin + f (xMax - xMin)

where f = i/(N – 1), [xMin, xMax] is the interval of x, and 0 <= i < N.

The job of the object of CURVE_FUNCTION returns information to the system. And then the system will draw the curves.

You implement GRAPH_MANAGER so that an object of GRAPH_MANAGER is able to perform the tasks, including the following:

- getNumCurves. Return the number of curves.
- getNumOfSamplePoints. Return the number of sample points of all curves.
- `setCurves_Random_C(). This function randomly generates parameter c inside the interval [mC_LowerBound, mC_UpperBound] for all the curves.`
- getC_Lowest. Return the lowest value of parameter c of all curves.
- getC_Largest. Return the largest value of parameter c of all curves.
- getC_Average. Return the average value of parameter c of all curves.

Output: You will see one or more curves, as illustrated in Figure 6.1. Figure 6.2 shows the information of the curves at the top bar.

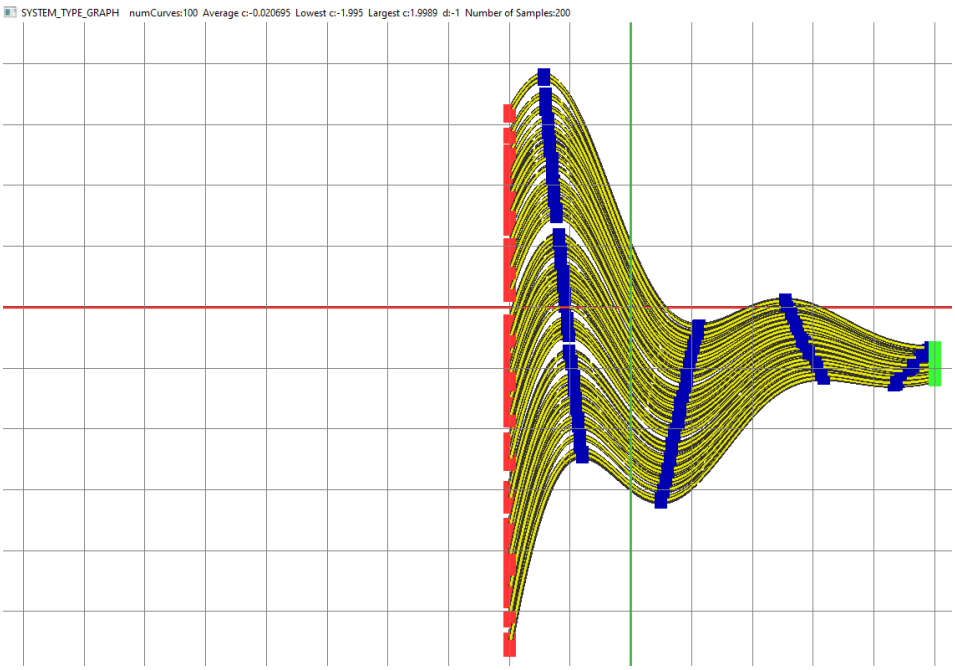


Figure 6.1: The blue points are extreme sample points. The leftmost and rightmost points of the curves are drawn in different colors. As can be seen, the boundary points are on the left or right side of the curve segments. If you change the number of sample points, you will see that the blue points will be moving.

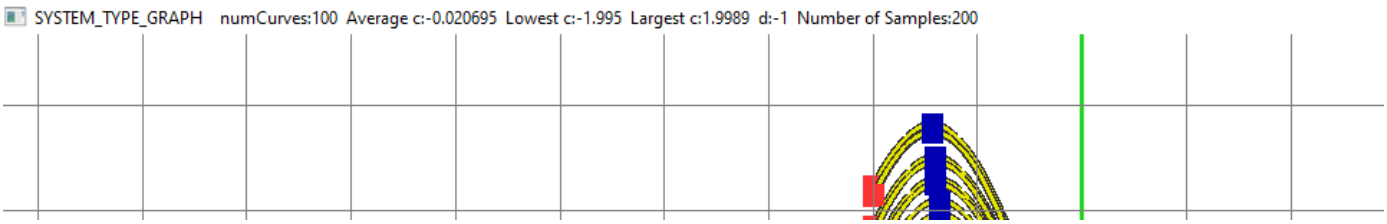


Figure 6.2: The top bar shows the information of the curves, including number of curves, average c value, lowest c value, largest c value, d value, and number of sample points.

You should refer to the files to see more functions that you need to implement.

Each function has two parameters, c and d.

The three functions y(x) are as follows:

Function 1 (CURVE_TYPE_EXPONENTIAL):

$w = (x / 2.5)$

$y = w / 10 - 1 + (c + \sin(4\ d\ w)) * e^{-w}$

Function 2 (CURVE_TYPE_CONSINE): $y = c\ x - d\ x \cos(x)$

Function 3 (CURVE_TYPE_QUADRATIC): $y = x^2 + c\ x + d$

The parameter c of all the curves is randomly generated inside the interval [`mC_LowerBound, mC_UpperBound`] in a uniform manner. For example, you can implement the following function to set the value of parameter c to all the curves:

```
void GRAPH_MANAGER::setCurves_Random_C()
{
	for (int i = 0; i < MAX_NUM_CURVES; ++i) {
		mCurves[i].setRandom_C(mC_LowerBound, mC_UpperBound);
	}
}
```

The following highlights some items that you need to implement. Please refer to the files to see more functions that you need to implement. You can add new data members or member functions.

The minimum number of curves is `MIN_NUM_CURVES`. Set `MIN_NUM_CURVES` to 1.

The maximum number of curves is `MAX_NUM_CURVES`. Set `MAX _NUM_CURVES` to 1000.

The detail requirement of the program is listed as follow.

1. Set the interval of x as [-2, 5].
2. Set the default number of sample points as MAX_NUM_SAMPLE_POINTS / 2.
3. Implement the three functions
4. Press 'F1': Set the number of curves to 1. Then show the curve(s).
5. Press 'F2': Set the number of curves to the maximum value. Then show the curve(s). Read the files to find out the maximum value.
6. Press '1': Change the curve type to CURVE_TYPE_EXPONENTIAL.
7. Press '2': Change the curve type to CURVE_TYPE_COSINE.
8. Press '3': Change the curve type to CURVE_TYPE_QUADRATIC.
9. Press 'k': Decrease the number of sample points of all the curves, i.e., mNumOfSamplePoints -= delta_num.
10. Press 'l' (lower case L): increase the number of sample points of all the curves, i.e., mNumOfSamplePoints += delta_num.
11. Press '<': Decrease the number of curves by 20 each time, i.e., mNumCurves -= 20. Regenerate randomly parameter c for all the curves.
12. Press '>' : increase the number of curves by 20 each time, i.e., mNumCurves += 20. Regenerate randomly parameter c for all the curves.
13. Press 'n': Decrease parameter d of all the curves, i.e., mParam_D -= mDelta_D.
14. Press 'm': increase parameter d of all the curves, i.e., mParam_D += mDelta_D.

Note that there are lower and upper bounds for mNumOfSamplePoints, mNumCurves, and mParam_D. Please the files for detail.