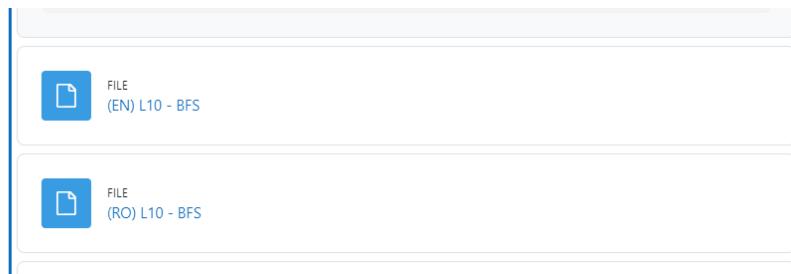


1 Assignment 9: Tutorial

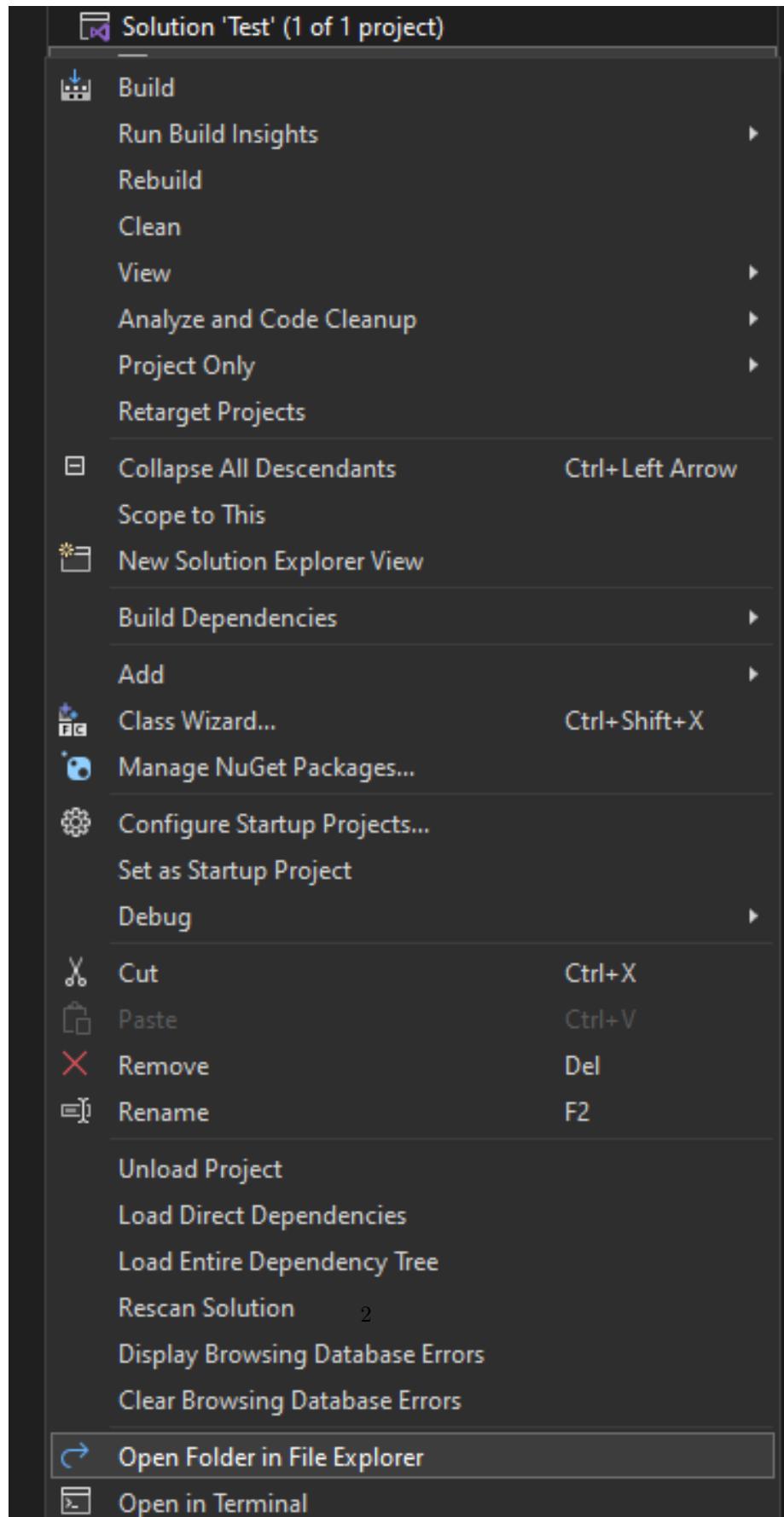
1.1 Visual Studio Project Setup

1. Go to moodle and select one of the following:

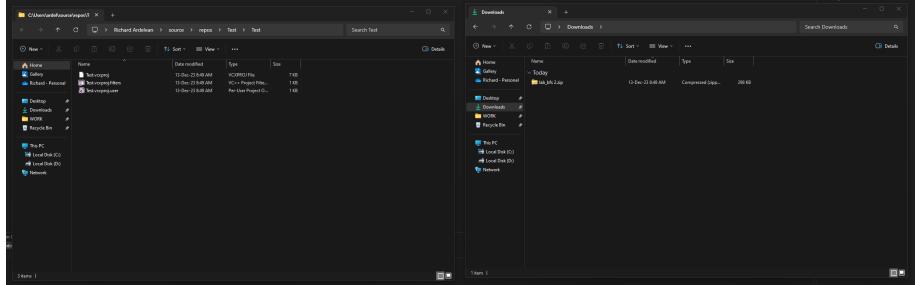


This will result in a '.zip' file being downloaded.

2. Create an Empty C++ Project and by right-clicking on the project (not the solution) you will be able to select 'Open Folder in File Explorer'.

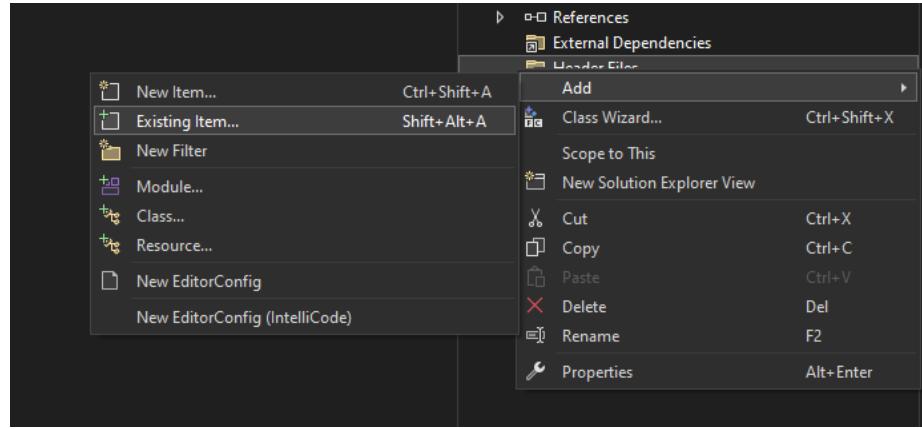


- At this point, a ‘File Explorer’ window will be opened. Open another ‘File Explorer’ with the location of your downloads (most likely the Downloads folder).

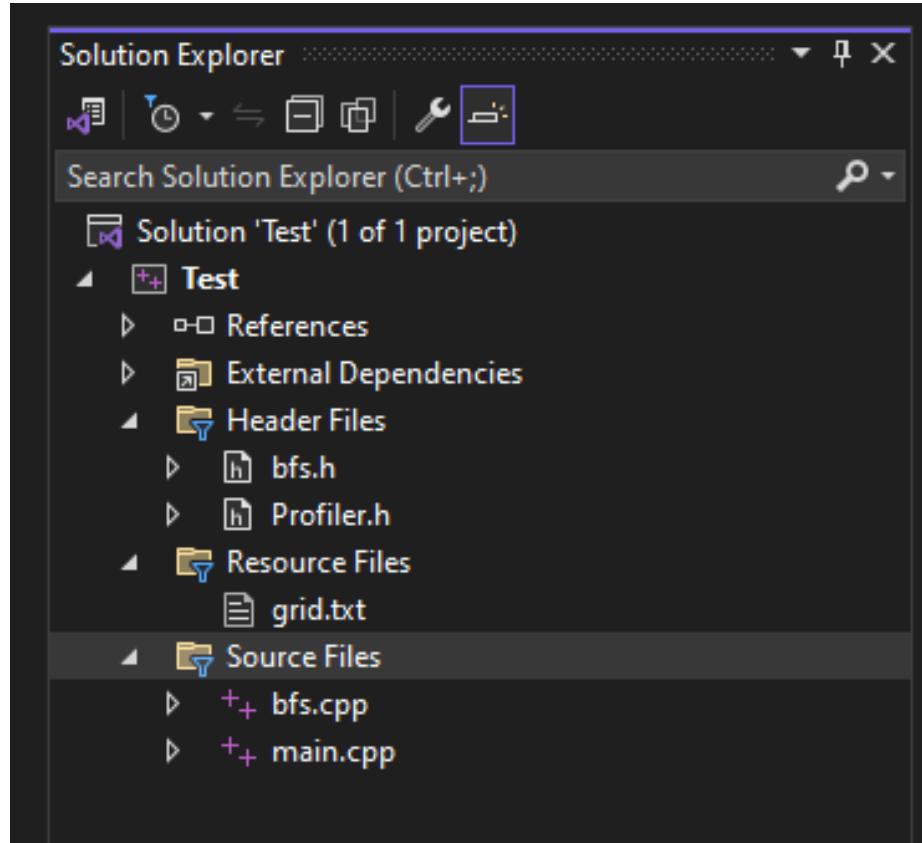


- Open the ‘.zip’ file and copy as shown below the files from the zip file to the folder of the project.

- By selecting the folders from the Visual Studio Solution Explorer ‘Header Files / Resources / Source Files’ use the following options:

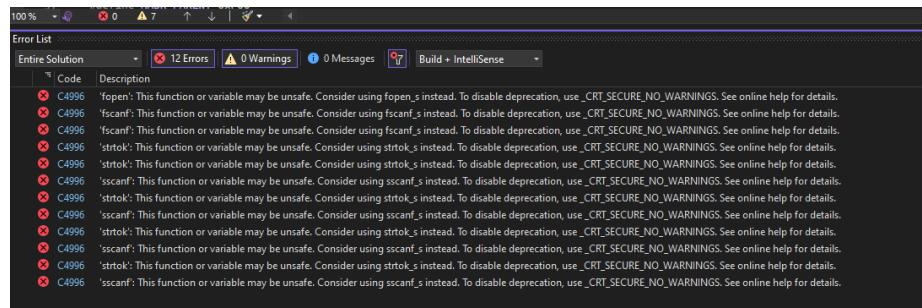


6. Such that, the following setup is accomplished:



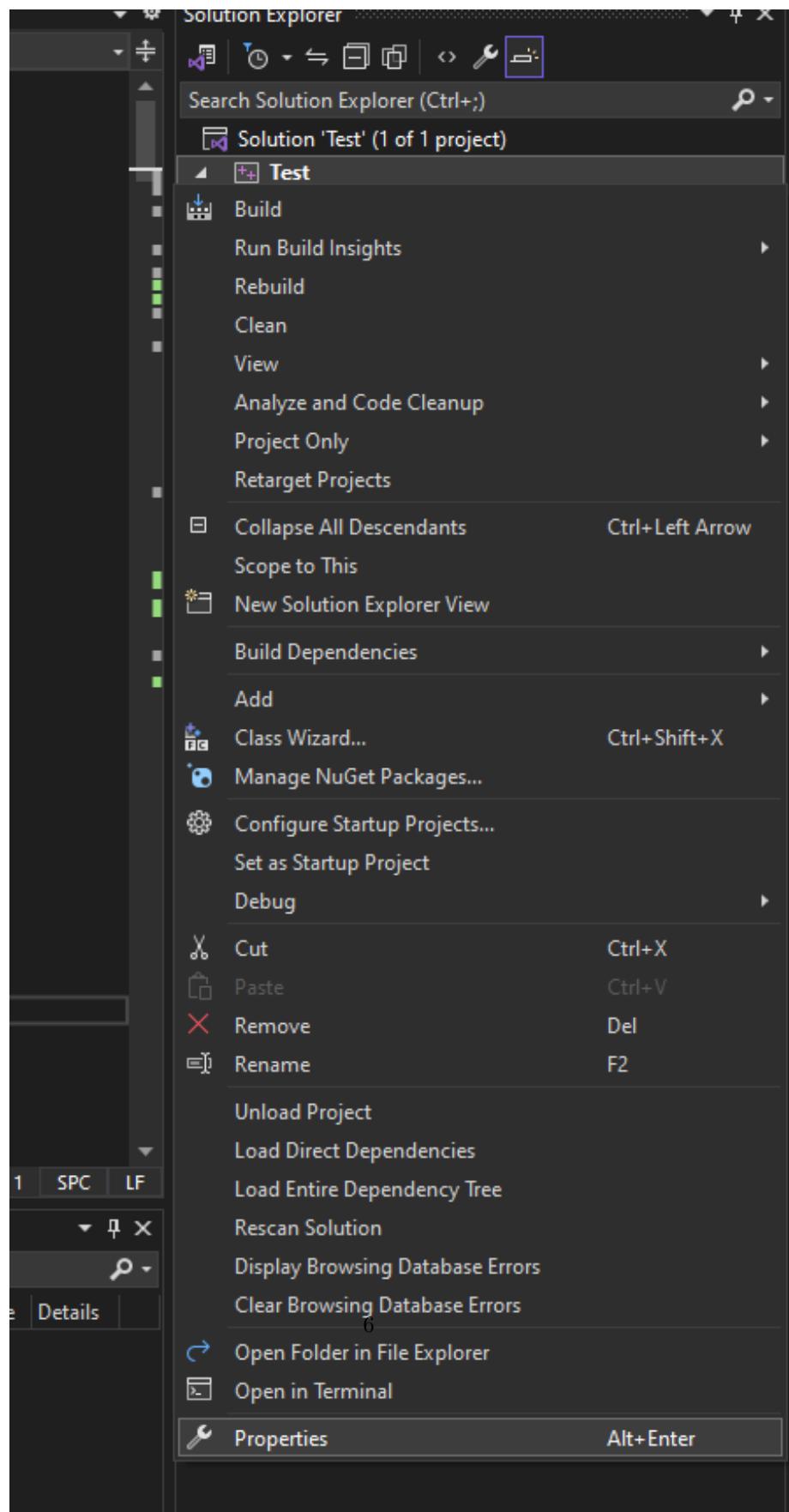
1.2 Visual Studio ‘unsafe’ error

Example:



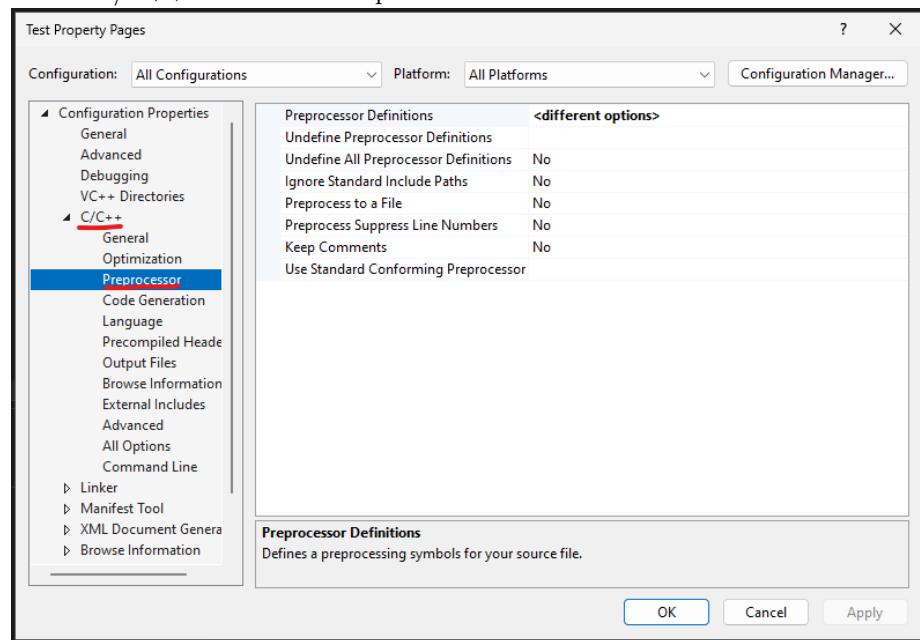
Solution:

Right-click on the project (not the Solution which will most likely have the same name) and select 'Properties'.

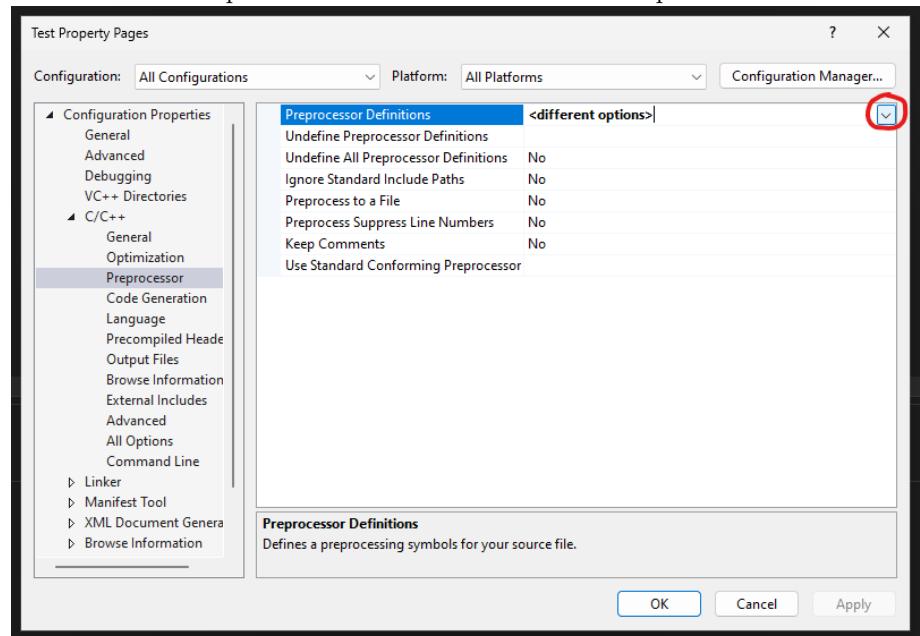


Update the ‘Configuration’ and ‘Platform’ in the upper side of the window to ‘All Configurations’ and ‘All Platforms’, respectively.

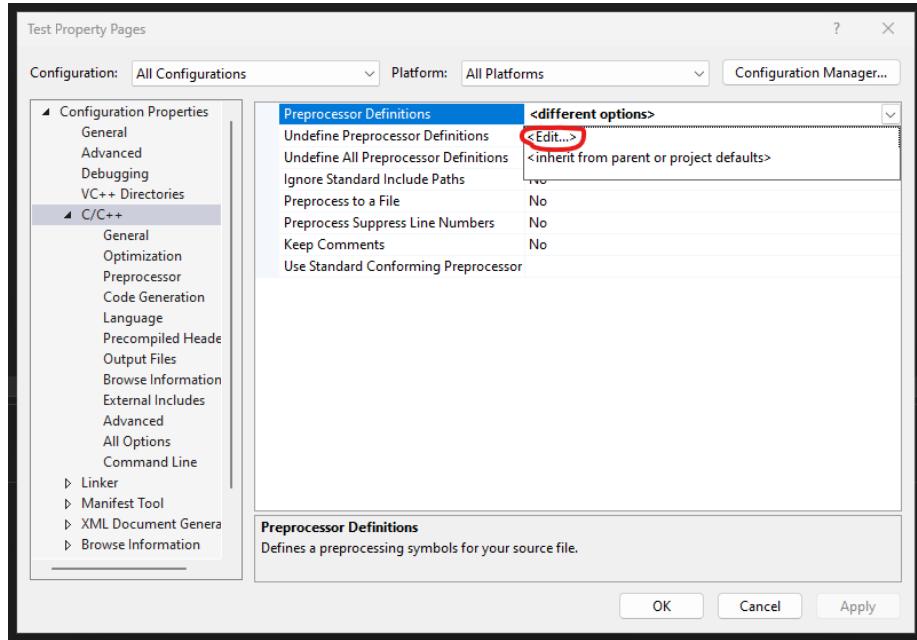
Go to ‘C/C++’ and select ‘Preprocessor’.



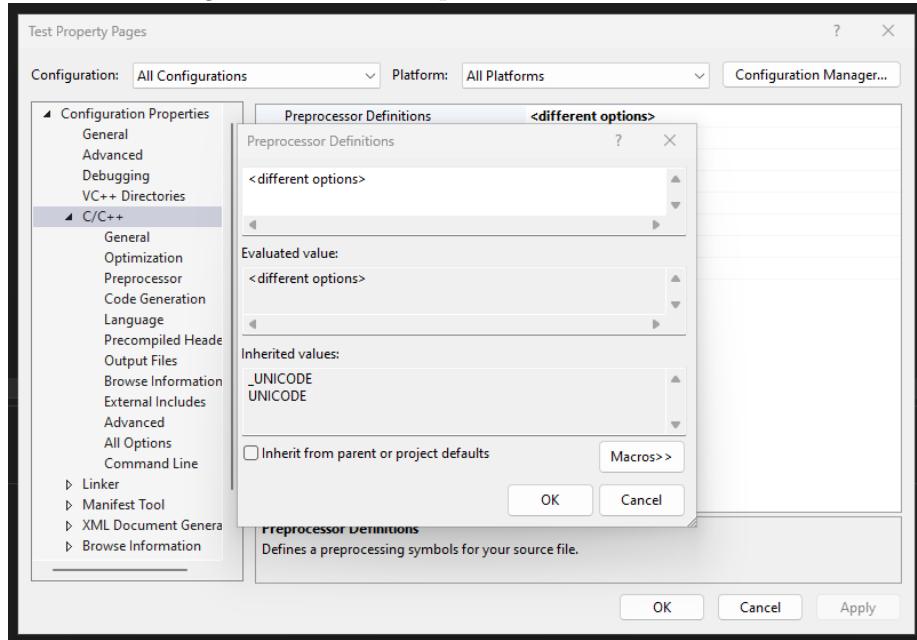
Then click on ‘Preprocessor Definitions’ and on the dropdown arrow.



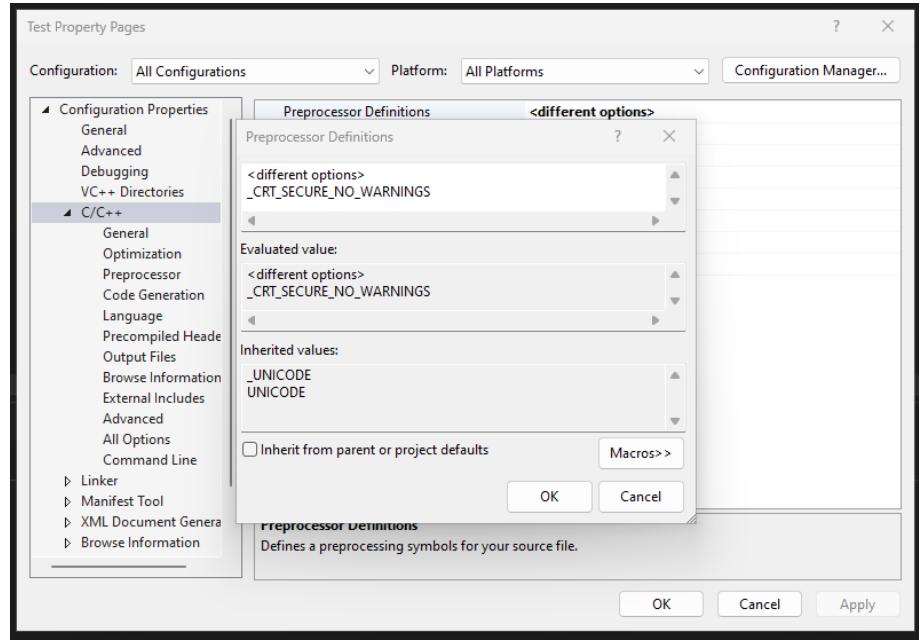
Select ‘Edit’



And the following new window will open:



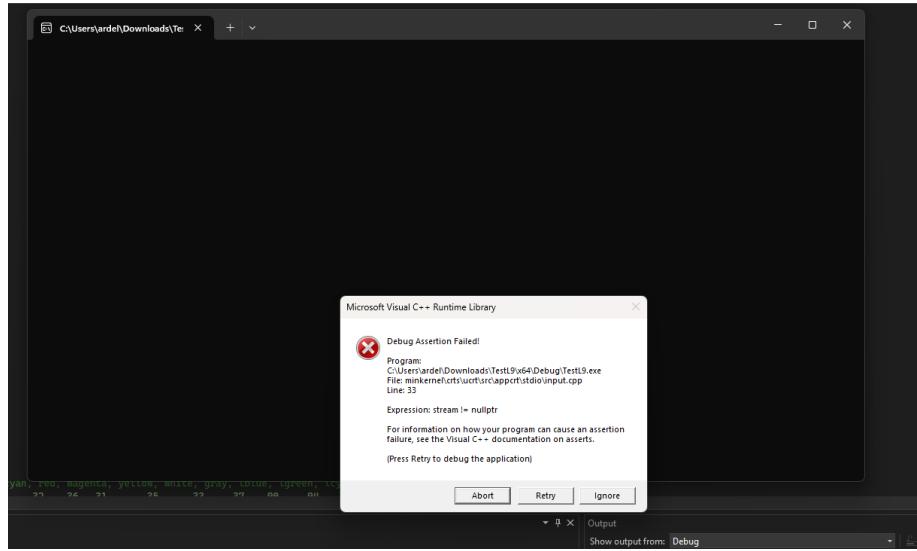
Introduce '_CRT_SECURE_NO_WARNINGS' under '<different options>' as shown below.



Click on ‘OK’ for all opened windows until you are returned to the main Visual Studio window of the project, and you will be able to run the project.

1.3 Visual Studio ‘Assertion’ Error

This indicates that you have not followed the tutorial. Go back to the first page and make sure that you have moved the files from the ‘Downloads’ folder to the ‘Project’ folder. You might also need to *remove* all the files from the Visual Studio IDE and *re-add* them by hand.

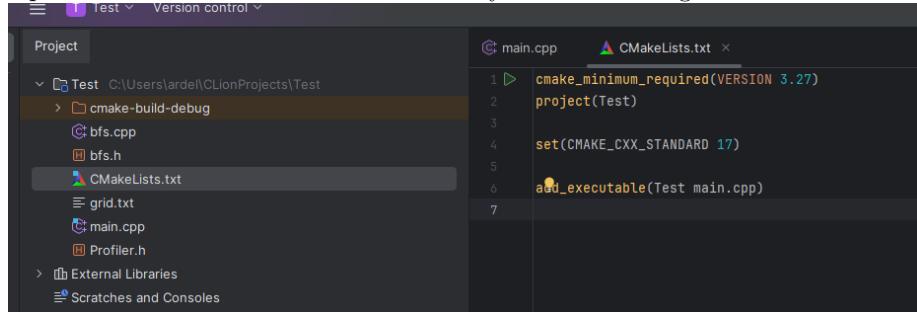


1.4 CLion undefined Error

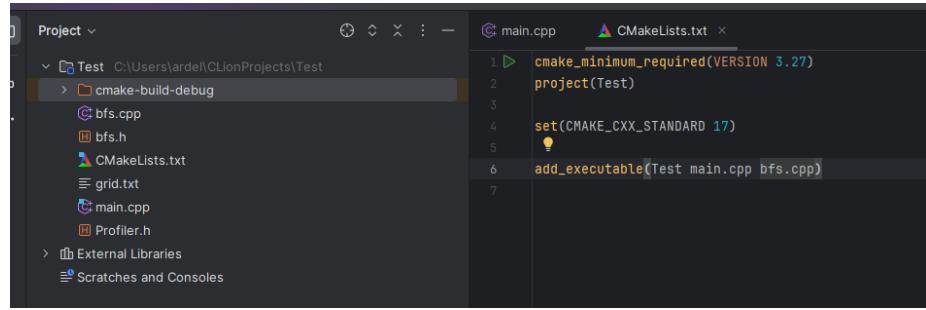
Example:

Solution:

Open the ‘CMakeLists.txt’ file and modify in the following manner:



```
add_executable(Test main.cpp)  
into  
add_executable(Test main.cpp bfs.cpp)
```

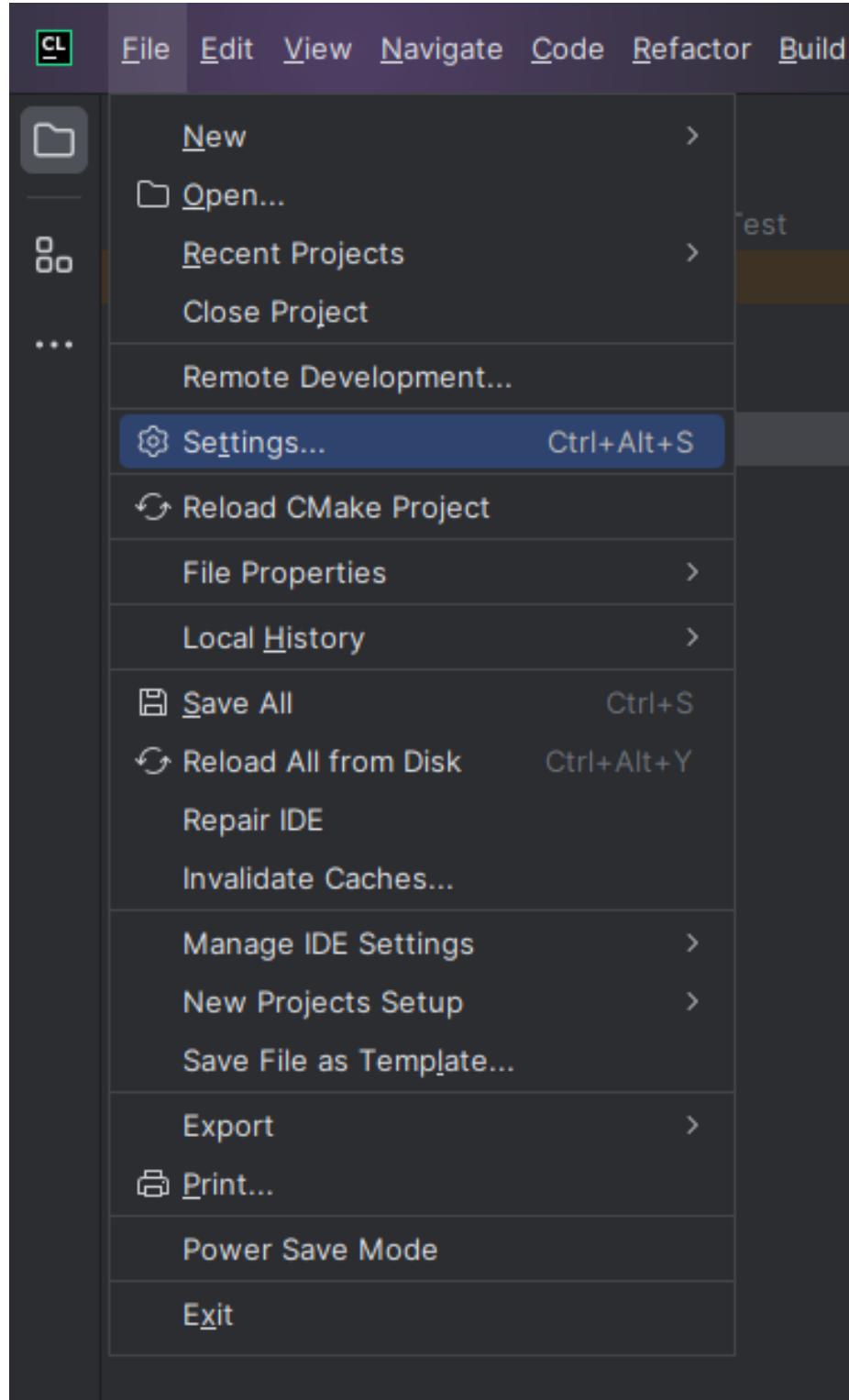


The screenshot shows the CLion IDE interface. On the left, the Project tool window displays a CMake project named 'Test' located at 'C:\Users\ardel\CLionProjects\Test'. Inside the project, there is a 'cmake-build-debug' folder containing files: 'bfs.cpp', 'bfs.h', 'CMakeLists.txt', 'grid.txt', 'main.cpp', and 'Profiler.h'. Below these are sections for 'External Libraries' and 'Scratches and Consoles'. On the right, the main editor window shows the 'CMakeLists.txt' file with the following content:

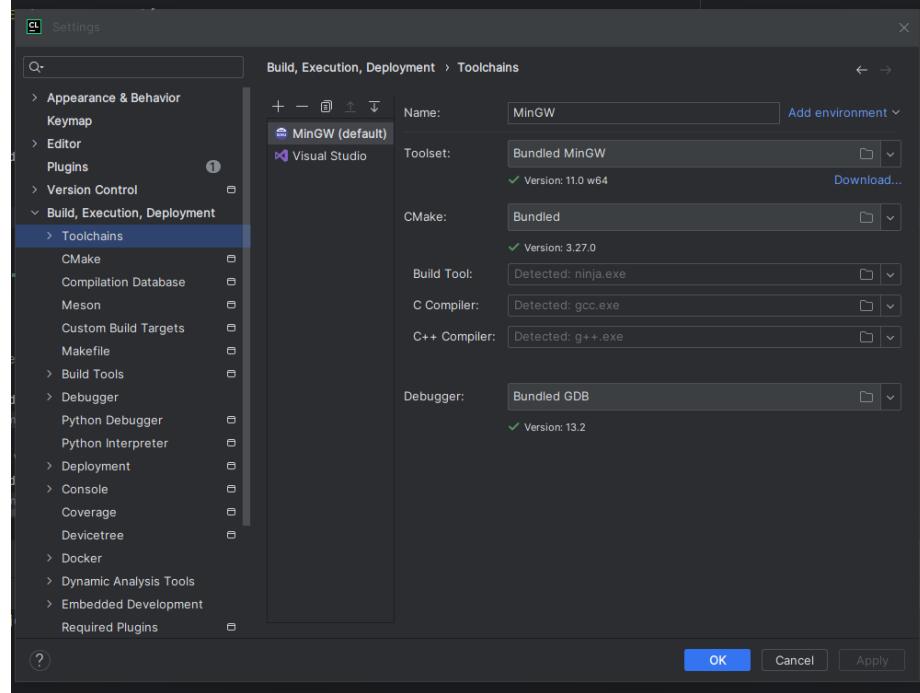
```
1 cmake_minimum_required(VERSION 3.27)
2 project(Test)
3
4 set(CMAKE_CXX_STANDARD 17)
5
6 add_executable(Test main.cpp bfs.cpp)
```

1.5 CLion Visual Studio – Option 1 (slower)

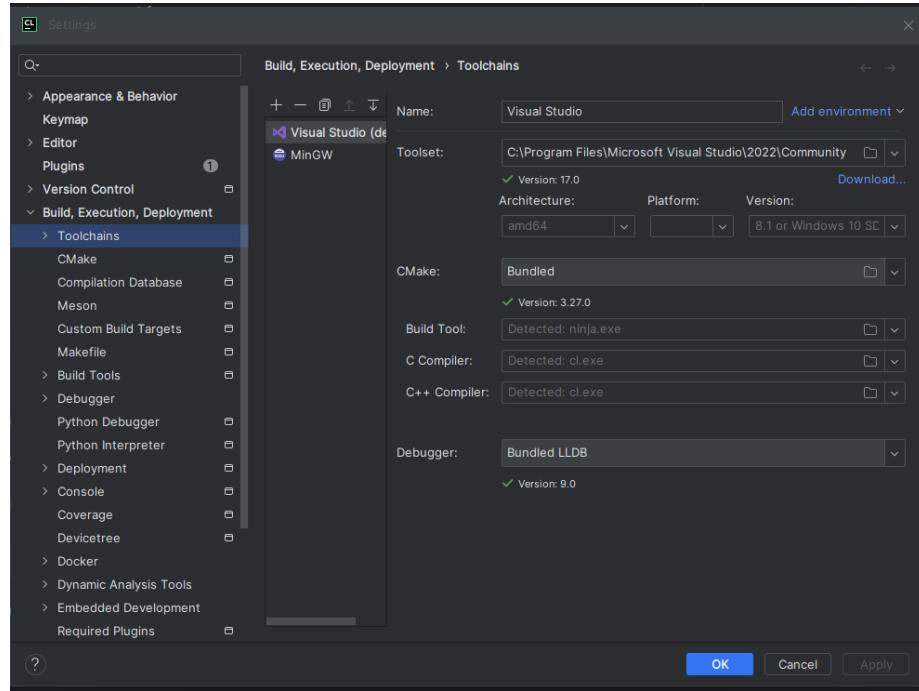
Go to File -> Settings:



In the newly opened window, go to ‘Build, Execution, Deployment’ -> ‘Toolchains’



Using the arrows, move Visual Studio as default:



1.6 CLion MinGW – Option 2 (faster, requires external console)

1.6.1 CLion Clear error

Example:

A screenshot of the CLion run interface. The tab bar shows 'Run' and 'Test'. The main area is titled 'Test' and contains a terminal window. The terminal output shows a command being run: 'C:\Users\ardel\CLionProjects\Test\cmake-build-debug\Test.exe'. Below this, an error message is displayed in red text: "'clear' is not recognized as an internal or external command, operable program or batch file." Navigation icons for the terminal are visible on the left.

Solution:

Go to the main.cpp file and scroll to lines 113-117 in the displayGrid function.

The screenshot shows the CLion IDE interface. On the left is the project tree, with 'Test' selected. Inside 'Test' are 'cmake-build-debug', 'bfs.cpp', 'bfs.h', 'CMakeLists.txt', 'grid.txt', 'main.cpp', and 'Profiler.h'. Under 'External Libraries' and 'Scratches and Consoles' are empty entries. On the right is the code editor for 'main.cpp'. The code is as follows:

```
98     return "/\\";
99 }else if((x & MASK_PARENT) == MASK_DOWN){
100     return "\\";
101 }else if((x & MASK_PARENT) == MASK_LEFT){
102     return "< ";
103 }else if((x & MASK_PARENT) == MASK_RIGHT){
104     return "> ";
105 }else{
106     return " ";
107 }
108 }
109 void displayGrid(const Grid *grid, int lastCommand)
110 {
111     int i, j;
112 #ifdef _MSC_VER
113     system("cls");
114 #else
115     system( Command: "clear" );
116 #endif
117 }
```

Modify in the following manner:

In the else branch from line 116
system("clear");
to
system("cls");

The screenshot shows the CLion IDE interface with the same project structure as the first screenshot. The code editor for 'main.cpp' has been modified. The change is located at line 116, where 'system("clear");' has been replaced by 'system("cls");'. The rest of the code remains the same.

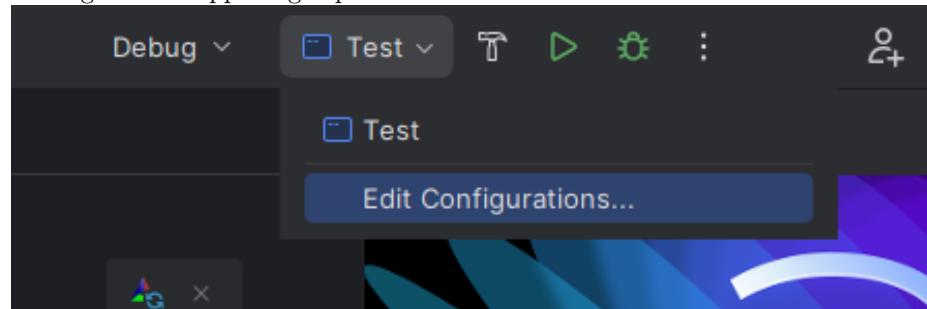
```
98     return "/\\";
99 }else if((x & MASK_PARENT) == MASK_DOWN){
100     return "\\";
101 }else if((x & MASK_PARENT) == MASK_LEFT){
102     return "< ";
103 }else if((x & MASK_PARENT) == MASK_RIGHT){
104     return "> ";
105 }else{
106     return " ";
107 }
108 }
109 void displayGrid(const Grid *grid, int lastCommand)
110 {
111     int i, j;
112 #ifdef _MSC_VER
113     system("cls");
114 #else
115     system( Command: "cls" );
116 #endif
117 }
```

1.6.2 CLion not showing grid



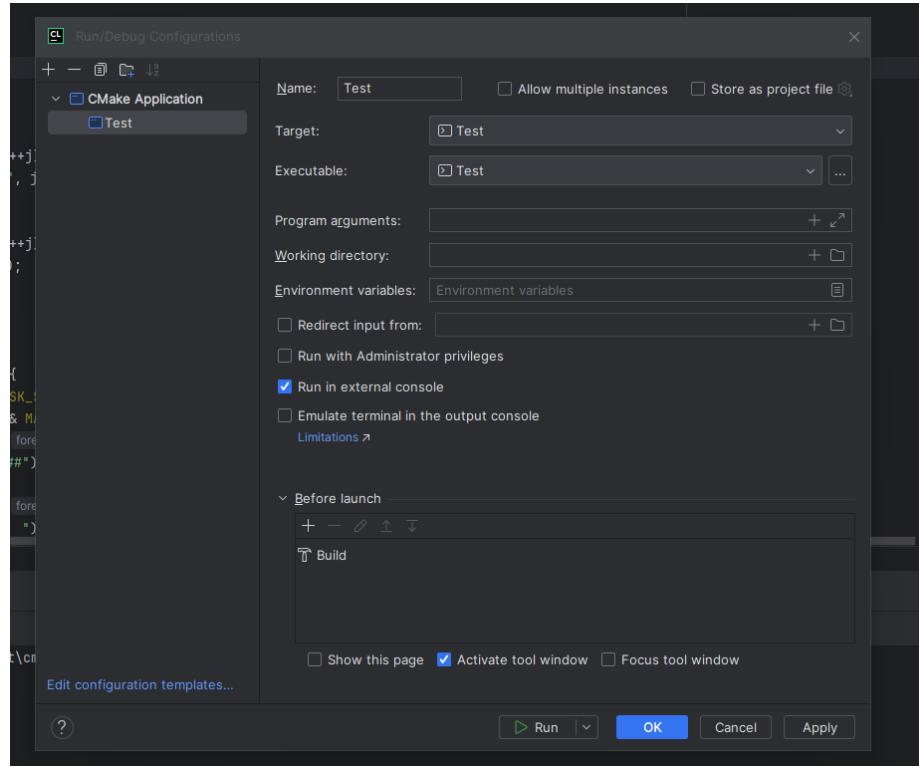
Copy the 'grid.txt' file into the 'cmake-build-debug' folder.

Then go to the upper right part of the screen:



Select 'Edit Configurations' and check the following boxes:

- Run in external console



1.7 Mac run command

```
g++ main.cpp bfs.cpp -std=c++11 && ./a.out
```