SCS 1310 - Object Oriented Programming

Tutorial 01

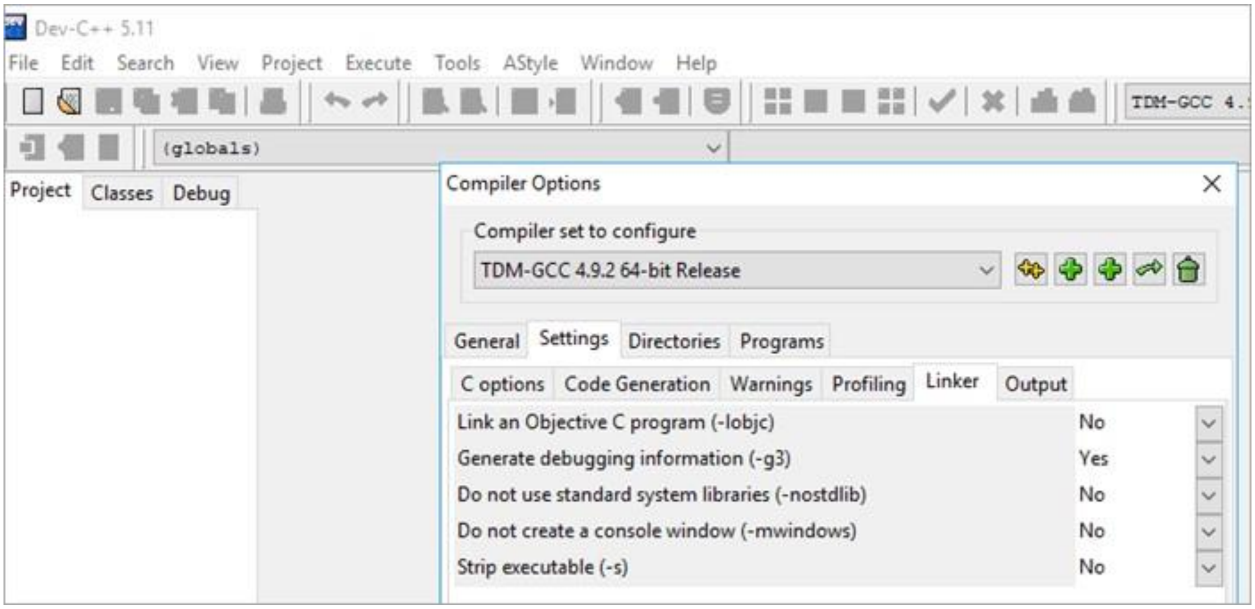
Change Linker Setting for Debugging

- First, correctly download and install DEV C++ IDE.
- After starting the DEV C++ IDE, the first thing we need to ensure is the setting for debugging information to be generated.

Follow the steps below to set the debugging information.

- To change this setting, click on **Tools -> Compiler Options.**
- Then click on the “**Settings**” tab on the dialog that pops up.
- Under “**Settings**”, we have a “**linker**” tab.
- In the “**linker**” tab there are various options shown. Set “**Yes**” for the option “**Generate Debugging Information (-g3)**”.

This is shown in the following screenshot.



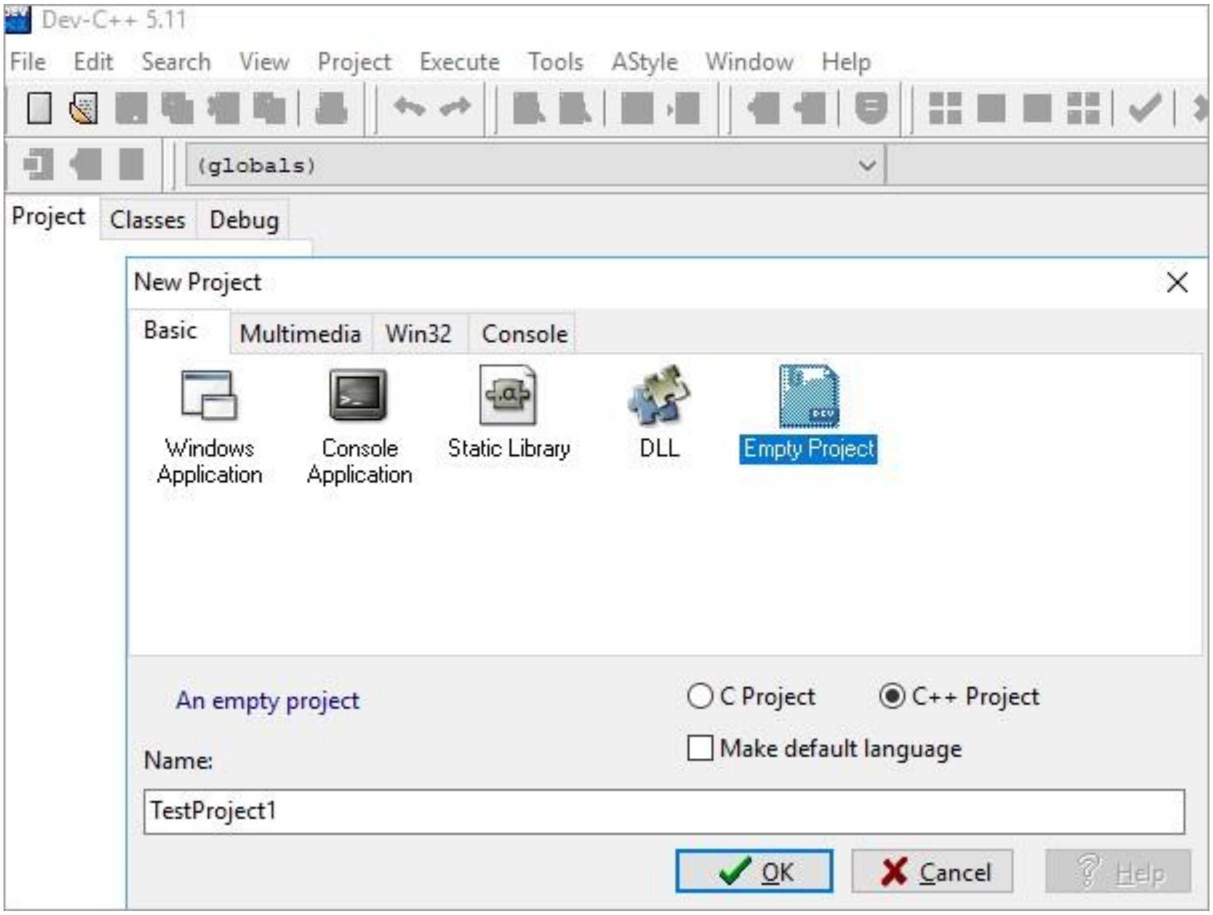
Compiler set to configure: TDM-GCC 4.9.2 64-bit Release

Link an Objective C program (-lobjc)	No
Generate debugging information (-g3)	Yes
Do not use standard system libraries (-nostdlib)	No
Do not create a console window (-mwindows)	No
Strip executable (-s)	No

Create A New Project

To create a new project in dev-C++ we need to follow the below steps:

- Click **File -> New -> Project.**
- A new dialog opens up as shown below.



An empty project

○ C Project ● C++ Project

□ Make default language

Name:

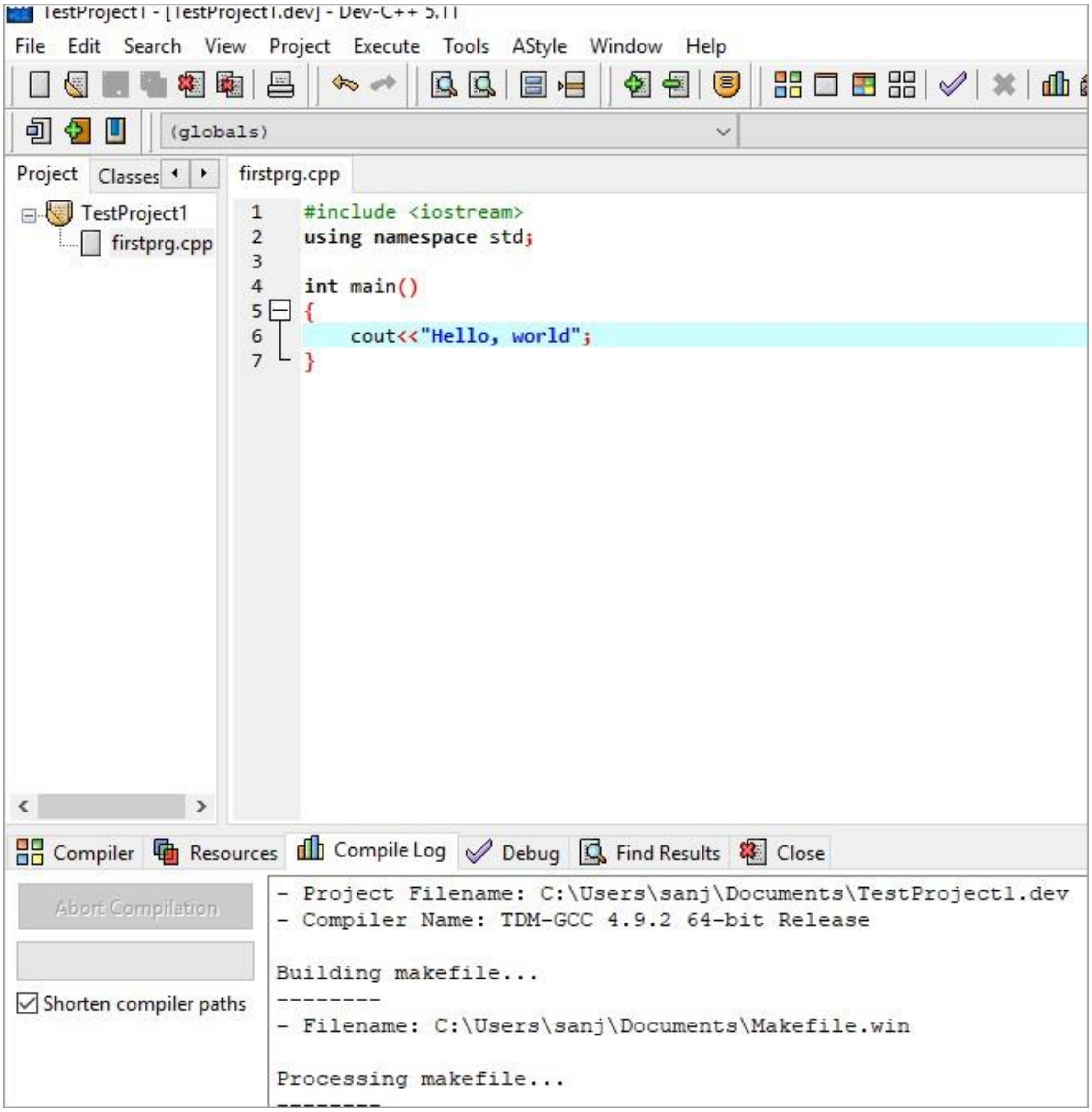
TestProject1

- Here, we can specify the project name. Make sure to select the “Empty Project” and also to check the “C++ Project” button.
- Once the entire information is provided, we can click ok and the IDE will ask for the path where the project is to be saved. When this is done, a workspace will open with the project explorer on the left-hand side that shows the project we just created.
- Now we can add or import the code files into this project.

Add Source File(s)

Adding a file to a project can be done in two ways.

1. Add a new file by clicking **Project ->New File** or Right-click on **Project Name** in the project explorer and click **New File**.
2. Another way is to add the existing files to the project. This can be done by clicking **Project ->Add to Project** or right-click on **Project Name** in the project explorer and select "**Add to Project…**" This will give a dialog to select files and import them to the project.
3. Once the files are added to the project, the workspace looks as shown below.



```
#include <iostream>
using namespace std;

int main()
{
	cout<<"Hello, world";
}
```

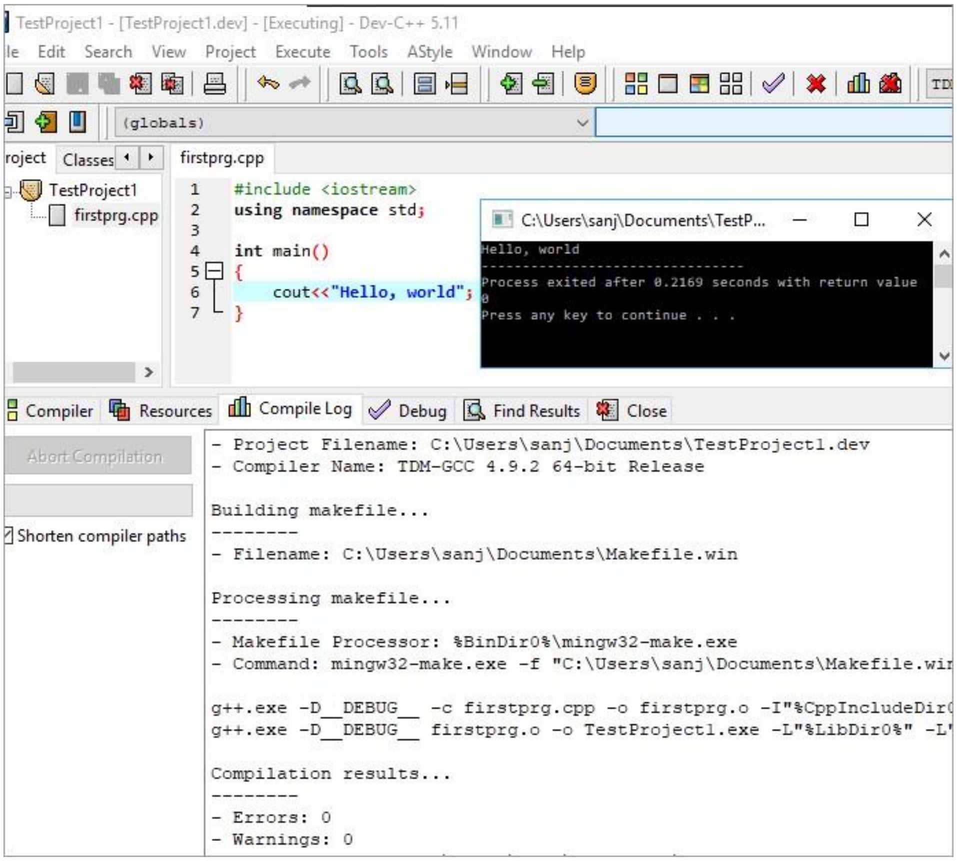
```
- Project Filename: C:\Users\sanj\Documents\TestProject1.dev
- Compiler Name: TDM-GCC 4.9.2 64-bit Release

Building makefile...
--------
- Filename: C:\Users\sanj\Documents\Makefile.win

Processing makefile...
--------
```

Compile/Build & Execute Project

When we have all the code ready for the project, we will now compile and build the project.

Follow the below steps to build and execute the dev C++ project:

- To compile the project, click **Execute -> Compile** (or click F9).
- We can see the compilation status in the “**Compile Log**” tab in the workspace.
- If there are any errors whether syntax or linker errors, then they will appear in the compiler tab.
- Once the project is compiled successfully, we need to run it.
- Click on **Execute -> Run**. (or click F10)
- The console window that gives us the output will be shown in the below screenshot.


TestProject1 - [TestProject1.dev] - [Executing] - Dev-C++ 5.11
le Edit Search View Project Execute Tools AStyle Window Help
(globals)
roject
Classes
TestProject1
firstprg.cpp
C:\Users\sanj\Documents\TestP...
Compiler
Resources
Compile Log
Debug
Find Results
Close
Abort Compilation
Shorten compiler paths

```
#include <iostream>
using namespace std;

int main()
{
    cout<<"Hello, world";
}
```

```
Hello, world
--------------------------------
Process exited after 0.2169 seconds with return value
0
Press any key to continue . . .
```

```
- Project Filename: C:\Users\sanj\Documents\TestProject1.dev
- Compiler Name: TDM-GCC 4.9.2 64-bit Release

Building makefile...
--------
- Filename: C:\Users\sanj\Documents\Makefile.win

Processing makefile...
--------
- Makefile Processor: %BinDir0%\mingw32-make.exe
- Command: mingw32-make.exe -f "C:\Users\sanj\Documents\Makefile.wi

g++.exe -D__DEBUG__ -c firstprg.cpp -o firstprg.o -I"%CppIncludeDir
g++.exe -D__DEBUG__ firstprg.o -o TestProject1.exe -L"%LibDir0%" -L

Compilation results...
--------
- Errors: 0
- Warnings: 0
```

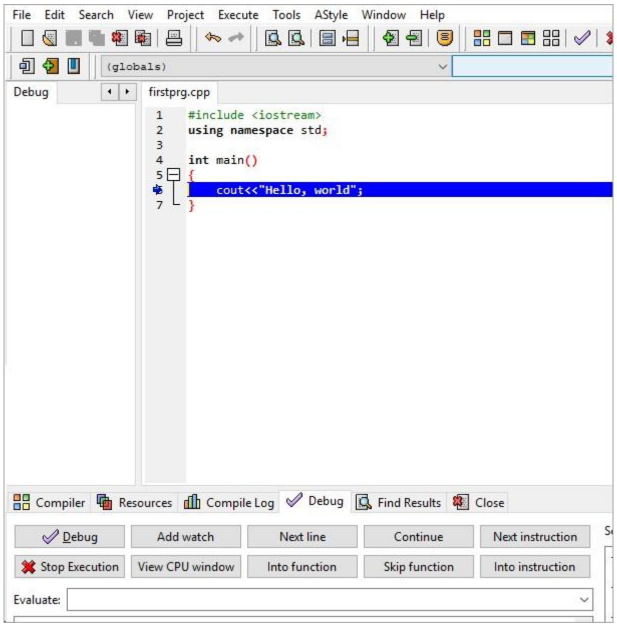
- If there are command line parameters to be passed to the program, we click on **Execute ->Parameters**. This will open a dialog using which we can pass parameters.

Debugging In C++ IDE

Sometimes we may not get the desired output from our program although the program is syntactically correct. In such a situation, we can debug the program. The dev-C++ IDE provides the inbuilt debugger.

Follow the below steps to debug the program using Dev-C++ IDE:

- Click **Execute ->Debug**. (or click **F5**).
- Once the debug is clicked, we get the debug menu in the IDE, as shown below.



```
#include <iostream>
using namespace std;

int main()
{
    cout<<"Hello, world";
}
```

- Before debugging we can toggle breakpoints using F4 at a particular line of code.
- Using the debug menu, we can use options like add watches, run to cursor, into a function, etc. to efficiently debug our program.

Activities

Prepare separate codes for each activity number.
Submit a pdf file including all activities with codes and screenshots of the result followed by the activity number.

1. Write a C program and a C++ program for the following
   a. Print “hello world”
   b. Prompt the user to input an integer value

2. Write a C++ program to print the following lines:

   C++ is an Object Oriented Programming Language. It is a
   collection of commands.

3. Write a C++ program to declare two integer, one float variables and assign 20, 18, and 14.2 to them respectively. It then prints these values on the screen.

4. Write a C++ program to prompt the user to input your name, age, gender(M or F),height and print these values.

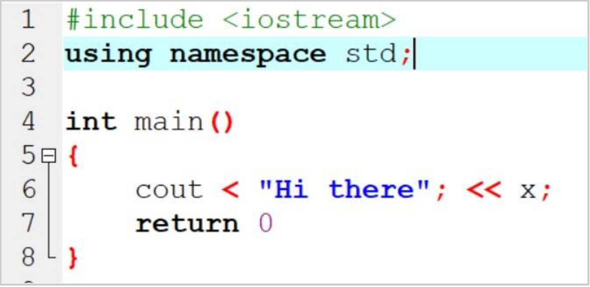
5. Write a C++ program to prompt the user to input radius value and calculate the area and volume for a sphere.

   The area of sphere = 4 * pi * radius * radius

   The volume of sphere = ⁄ * pi * radius * radius * radius Note: pi=3.14

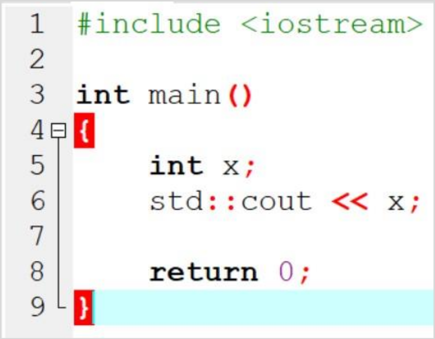
6. Identify the error in the following programs, correct and rewrite the program.

   a.



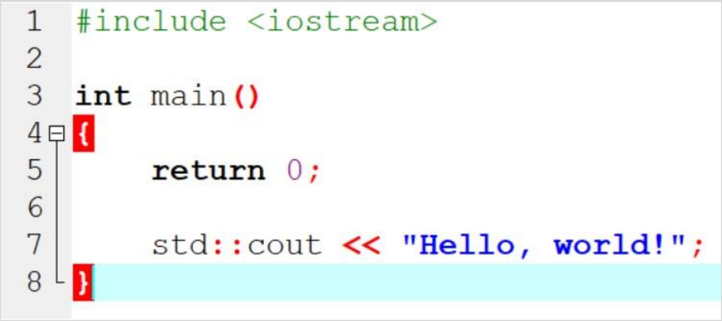
```
#include <iostream>
using namespace std;

int main()
{
    cout < "Hi there"; << x;
    return 0
}
```

b.



```
#include <iostream>

int main()
{
    int x;
    std::cout << x;

    return 0;
}
```

c.



```
#include <iostream>

int main()
{
    return 0;

    std::cout << "Hello, world!";
}
```