Programming Technique

Lab 1 – Welcome to C++

1 Expected outcomes

- Know how to write a simple C++ program.
- Know how to include libraries and used provided functions.
- Know how to use simple tools provided in the IDE (preferable visual studio)
- Learn how to compile the codes into an executable file and run it to view the result.

2 Mandatory exercises

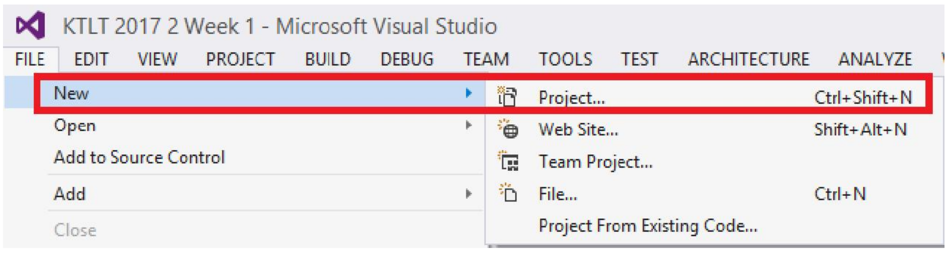
Exercise 1. Compile your first program using the following codes:

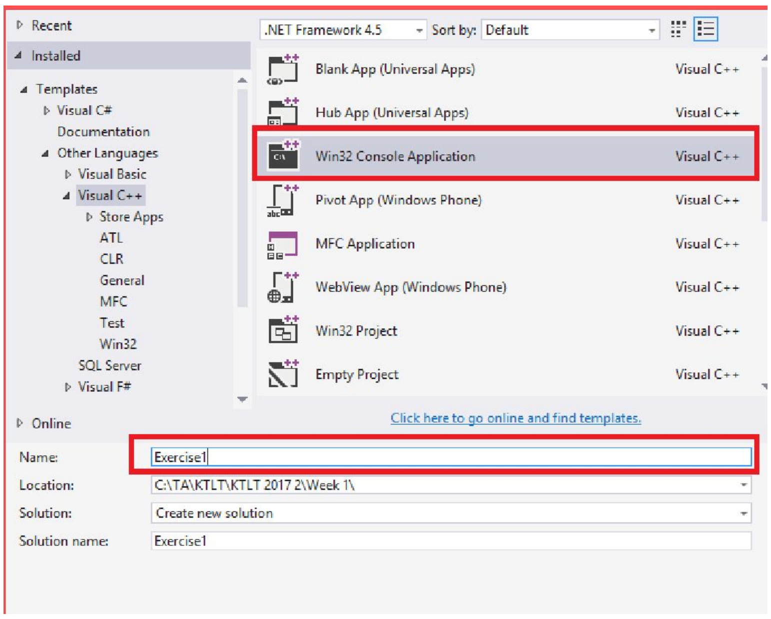
```
#include <iostream>
using namespace std;
int main() {
	// Code here
	cout << "Compilation successful!" << endl;
	return 0;
}
```

Guidelines:

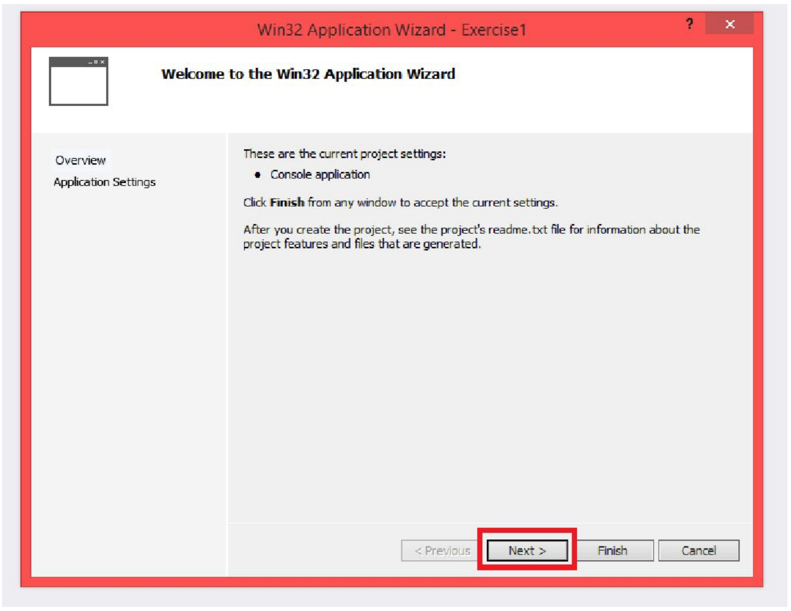
The meaning of each line in the above codes will be explained in Exercise 2. In this exercise, your task is only to compile the above codes using Visual Studio 10+.

Create your **Empty C++ project** by doing the following steps:




Recent
Installed
Templates
Visual C#
Documentation
Other Languages
Visual Basic
Visual C++
Store Apps
ATL
CLR
General
MFC
Test
Win32
SQL Server
Visual F#
Online
.NET Framework 4.5
Sort by: Default
Blank App (Universal Apps) Visual C++
Hub App (Universal Apps) Visual C++
Win32 Console Application Visual C++
Pivot App (Windows Phone) Visual C++
MFC Application Visual C++
WebView App (Windows Phone) Visual C++
Win32 Project Visual C++
Empty Project Visual C++
Click here to go online and find templates.

Name: Exercise1
Location: C:\TA\KTLT\KTLT 2017 2\Week 1\
Solution: Create new solution
Solution name: Exercise1



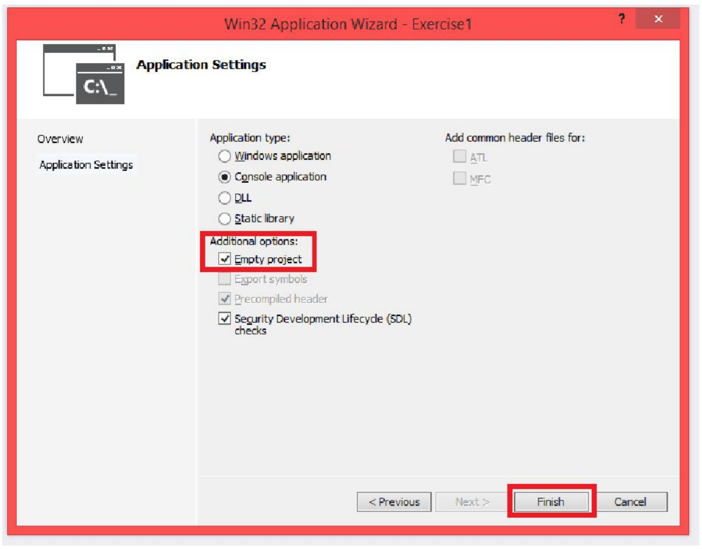
Welcome to the Win32 Application Wizard

These are the current project settings:

- Console application

Click **Finish** from any window to accept the current settings.

After you create the project, see the project's readme.txt file for information about the project features and files that are generated.

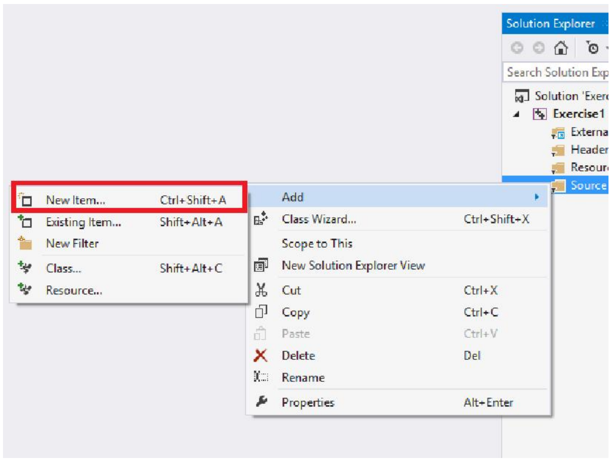


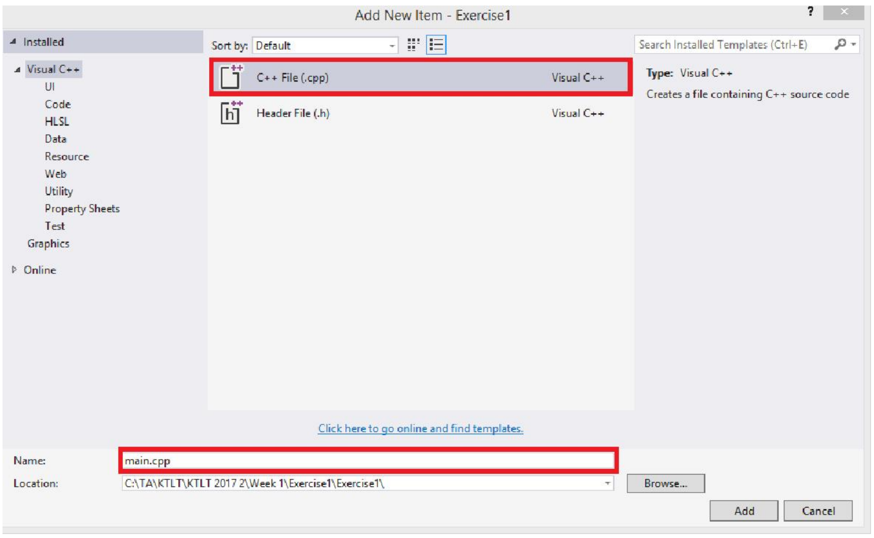
Application type:
( ) Windows application
(●) Console application
( ) DLL
( ) Static library

Additional options:
[✓] Empty project
[ ] Export symbols
[✓] Precompiled header
[✓] Security Development Lifecycle (SDL) checks

Add common header files for:
[ ] ATL
[ ] MFC

Create a text file called **main.cpp** to store the codes:

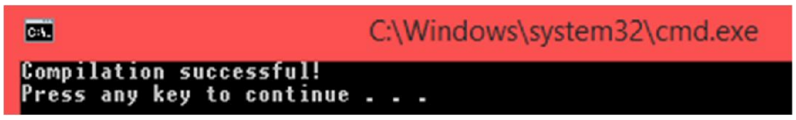




Type: Visual C++
Creates a file containing C++ source code

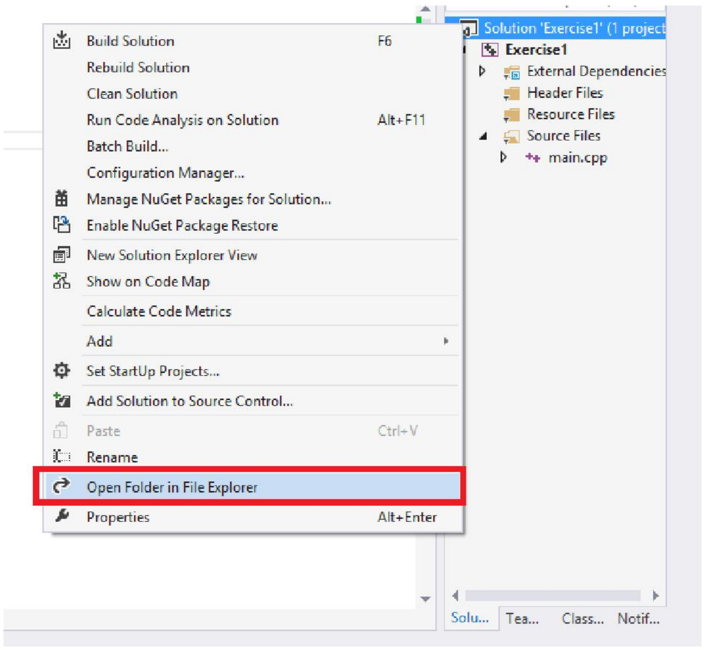
Name: main.cpp
Location: C:\TA\KTLT\KTLT 2017 2\Week 1\Exercise1\Exercise1\

Copy and paste the codes into main.cpp. On the top of the IDE, hit Build->Build solution. This will build your codes into an executable program in Windows. If there is no error, hit CTRL+F5 to run the program:



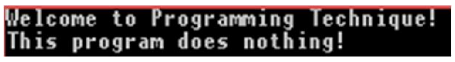
```
C:\Windows\system32\cmd.exe
Compilation successful!
Press any key to continue . . .
```

If you ever wonder where the built **executable program** (.exe) go, you can do this:



In the solution folder, go into "Debug" folder and you will see the .exe file.

Exercise 2. Write a program that prints "Welcome to Programming Technique!" and "This program does nothing!" in two separate lines.



```
Welcome to Programming Technique!
This program does nothing!
```

Guidelines:

For a C++ program to be executable, it must have a `main` function. We will discuss what a function is later into the course. For this exercise, writing a simple `main` function like this is enough:

```
int main() {
        // Code here
        return 0;
}
```

Now that we know what a C++ program must consist of, we can start breaking the main problem into 2 smaller tasks:

prints "Welcome to Programming Technique!" and "This program does nothing!" in **two separate lines**. You need to ask yourself "How do I print something in C++" and "How do I print them on two separate lines" and work from there.

1) To print in C++, we need to include the built-in library that allows us to print:

```
#include <iostream>
```

And, to print any saying or text, write the code as follow:

```
std::cout << "ABC";
```

- std::cout << "Something"; allow us to print "Something" onto the console.
- We need std:: because cout belong to the std namespace (standard namespace). This is needed if you have written your own cout, for example my::cout. In that case, you need the prefix std:: and my:: to know which cout you are using.
- If you only use std::cout in your program, you can omit the std:: by calling:

```
using namespace std;
```

with this, when printing, you only need to type cout:

```
cout << "ABC";
```

2) You need to know how to print two separate lines. You might think that this codes:

```
cout << "ABC";
cout << "DEF";
```

Will print "ABC" and "DEF" on two lines. They don't. To do it, you can explicitly specify a "new line" symbol – "\n":

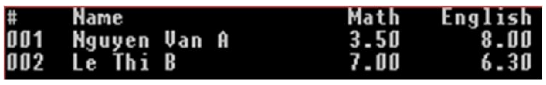
```
cout << "ABC\nDEF";
```

\n is completely similar to when you hit Enter on your keyboard. The lines will break into two.

Another way to create a new line is to use the built-in std::endl:

```
cout << "ABC" << endl << "DEF";
```

Exercise 3. Write a program to print a few rows of student records:



```
#    Name                 Math   English
001  Nguyen Van A         3.50      8.00
002  Le Thi B             7.00      6.30
```

The formatting must be correct:

- The first, second, third and fourth columns have the widths of 5, 15, 10 and 10 respectively.
- The first two columns are left-aligned and the last two are right-aligned.
- The precision of the scores is 2.

Guidelines:

- To set the width of a column or field, use std::setw(width) of the library iomanip:

```
cout << std::setw(10) << "ABC";
```

- To set the alignment, use std::left, std::right or std::internal:

```
cout << std::right << std::setw(10) << "ABC";
```

- To set precision of real (float or double) numbers, use std:setprecision(precision). To pad zeros at the end of the number, use std::fixed:

```
cout << std::setprecision(5) << std::fixed << 3.3 << "ABC" << 4;
```

Take notice of what happens when you use 4 instead of 4.0.

Exercise 4. Compute some basic operations and print them:

a) 3 / 2
b) 4 % 2
c) ‘a’ + 4
d) ‘a’ + ‘b’
e) “a” + 4
f) “a” + ‘b’

Notice the strange results in some cases and try to explain them.

Guidelines:

Apart from printing string, std::cout also allow use to print anything as long as it belongs to one of the categories: string, number, character, boolean, etc. Basically, C++ built-in types are printable with cout.

```
cout << 34991 + 1234;
```

Exercise 5. Print the following values in binary, base 8 (octal) and base 16 (hex):

a) Integers: 14, 32, -13, -72.
b) Real numbers: 22.5, 13.1, 4.6, -33.5.
c) Characters: ‘a’, ‘%’, ‘+’.

Guidelines:

- To print a number in base 2, we can use the bitset function provided in bitset library:

```
#include <bitset>

cout << bitset<32>(2) << endl;
```

The above line print number 2 as a string of 32 binary bits.

- To print a number in base 8, use keyword oct.
- To print a number in base 16, use keyword hex.

Exercise 6. Print a number after shifting it a few bits to the left or to the right. For example:

- $2_{10} = 0010_2$, shifting it 2 bits to the left, the result is 1000_2.
- $2_{10} = 0010_2$, shifting it 2 bits to the right, the result is 0000_2.

Calculate the following:

a) 15, shifting 3 bits to the left, bitset is 4.
b) 28, shifting 4 bits to the right, bitset is 5.

Guidelines:

In C++, you can shift bits of integer-type data by using the operators << (left-shifting) or >> (right-shifting).

For example:

```
10 << 3;
```

Means that we shift 10 by 2 bits to the left.

Exercise 7. Given the following table:

Binary	Oct	Dec	Hex
1011			
		324	
			A4DE

	337		
		15	

Fill out the rest of the cells of the table.

Guidelines:

1) To convert a number X in decimal to binary, keep dividing it by 2:
   - If the remainder is 0 then add 0 to the left of the bit string.
   - If the remainder is 1 then add 1 to the left of the bit string.
   - If X = 0, stop.
2) To convert a number X in decimal to octal or hex, doing the same as above but with 8 or 16 instead of 2.
3) To convert a number X in octal to decimal, do the following:
   - Multiply each digit in X with 8^k and sum them up.
   - For example, $342_8 = (3 * 8^2 + 4 * 8^1 + 2 * 8^0)_{10} = 226_{10}$
4) To convert a number X in binary or hex to decimal, do the same as (3) but with 2 or 16 instead.
5) To convert a number in binary to octal:
   - Group the bit string into groups of three.
   - Convert each group of three bits into an octal digit.

Example: 11100101 = 011 100 101 (if the number of bits is not divided by 3, pad 0 to the left of the string until it is).

$011\ 100\ 101_2 = 3 \quad 4 \quad 5 = 345_8$

Conversion table:

Binary	000	001	010	011	100	101	110	111
Octal	0	1	2	3	4	5	6	7

6) To convert a number in binary to hex: do the same as (5) but each group has 4 bits instead of 3. The following is the conversion table for binary to hex:

Binary	0000	0001	0010	0011	0100	0101	0110	0111
Octal	0	1	2	3	4	5	6	7
Binary	1000	1001	1010	1011	1100	1101	1110	1111
Octal	8	9	A	B	C	D	E	F

7) To convert between octal and hex, convert the number to binary first then convert it to the base you want.