Object-oriented Programming

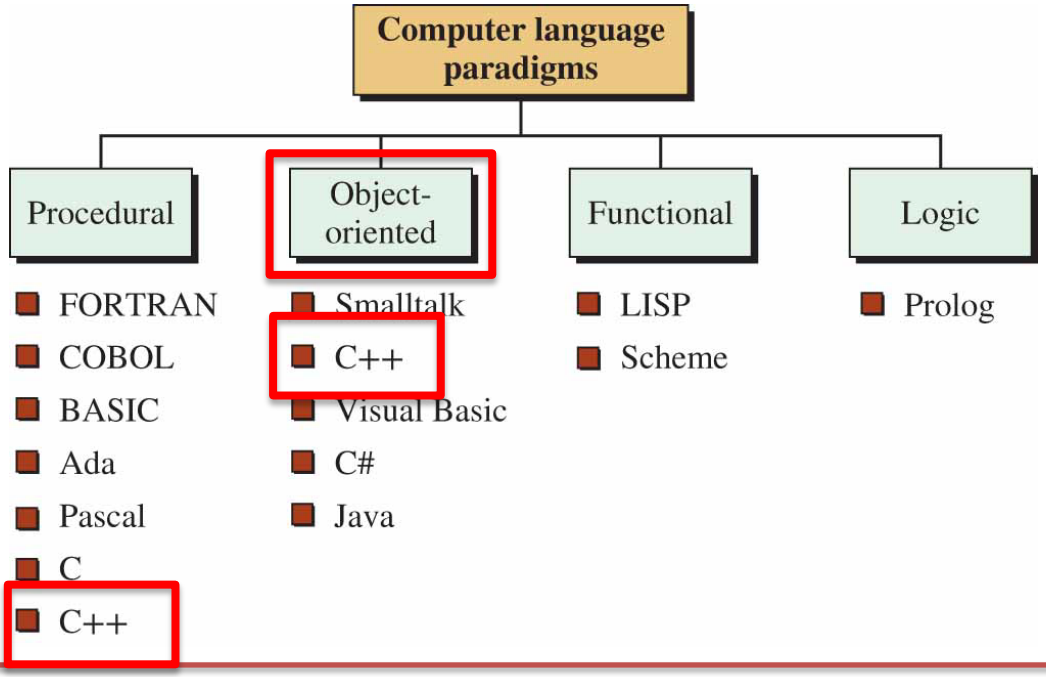
YoungWoon Cha
CSE Department
Spring 2023

Review

RECAP: LANGUAGE PARADIGMS

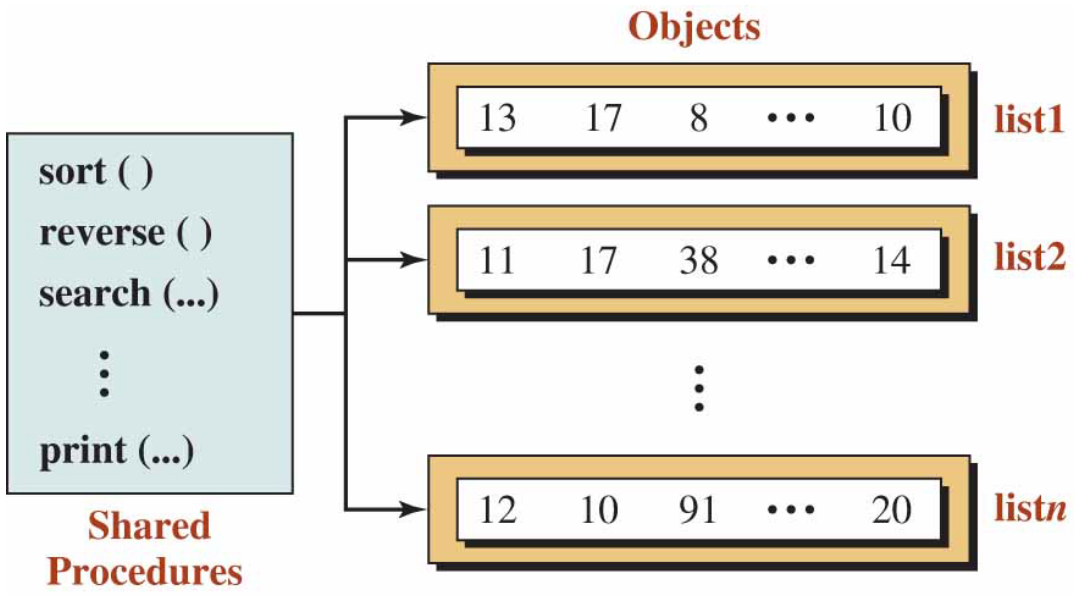
A paradigm is a model or a framework for describing how a program handles data.

Figure 1.10 ***Language paradigms***



RECAP: Object-Oriented Paradigm

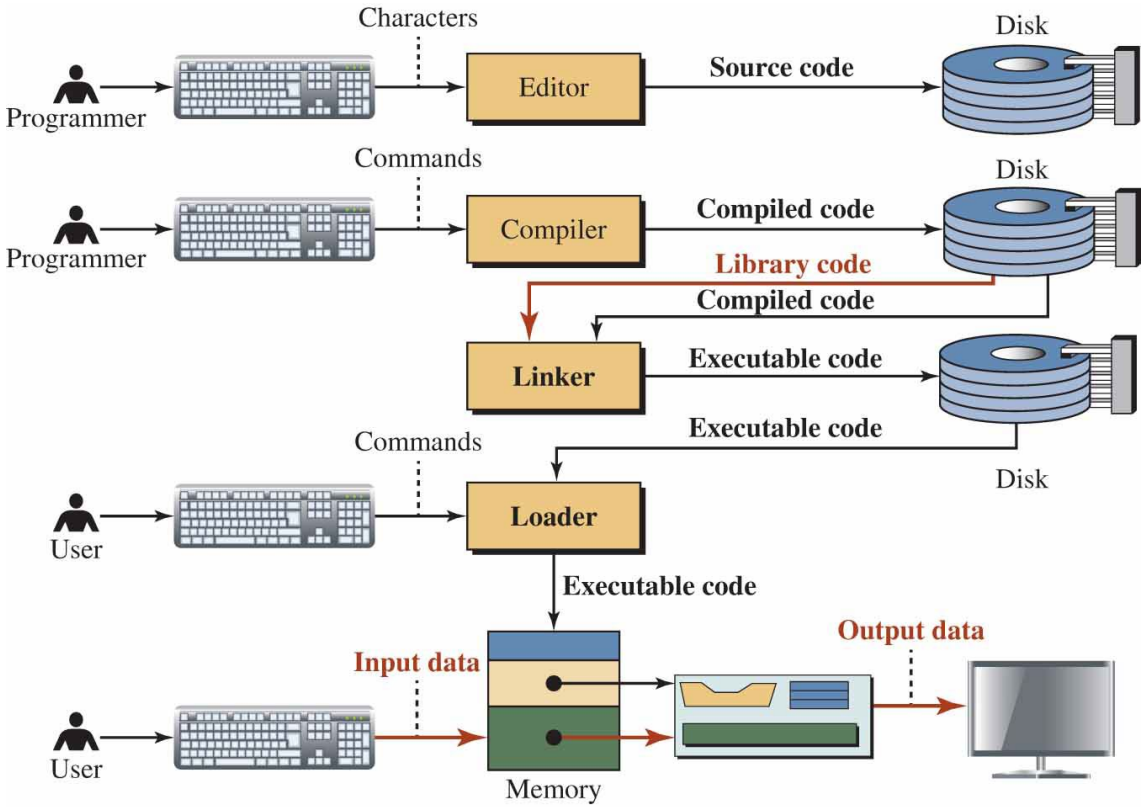
Figure 1.12 ***An example of an object-oriented paradigm***



An object is a package containing all possible operations to particular type of data structure.

RECAP: PROGRAM DEVELOPMENT

Figure 1.17 ***Writing, editing, and executing a program***



The general procedure for turning a program written in any language into machine language.

RECAP: OOP Paradigms

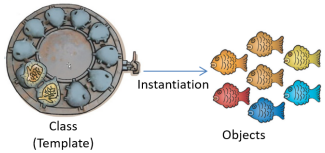
- OOP Paradigms improve the productivity in programming.

Encapsulation

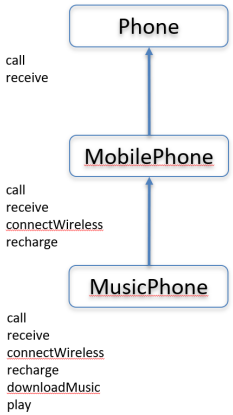
Members

```
class Circle {
    private:
        int radius;
    public:
        Circle(int r) { radius = r; }
        double getArea() { return 3.14*radius*radius; }
};

Circle c1;
Circle c2;
```



Inheritance




```
class Phone {
    void call();
    void receive();
};
```

```
class MobilePhone  : public Phone{
    void connectWireless();
    void recharge();
};
```

```
class MusicPhone  : public MobilePhone{
    void downloadMusic();
    void play();
};
```

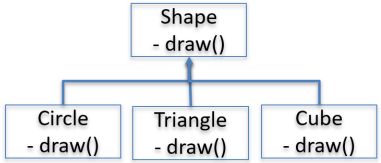
Polymorphism

```
void add(int a, int b) { ... }
void add(int a, int b, int c) { ... }
void add(int a, double d) { ... }
```

add function overloading

```
        2 + 3          --> 5
   "Male" + "Female"          --> "MaleFemale"
Color red + Color blue   --> Color purple
```

+ operator overloading



draw function overriding

Generic Programming

Without Generic Programming:

```
void swap(int & x, int & y)      { int tmp = x; x = y; y = tmp;    }
void swap(long & x, long & y){ long tmp = x; x = y; y = tmp; }
…
```

With Generic Programming

```
template <typename T>
    void swap(T & x, T & y) {  T tmp = x; x = y; y = tmp; }
```

Hello World ! In C++

First Simple Program

Program 2.1 *The first simple program*

```
#include <iostream>

int main()
{
    std :: cout << "This is a simple program in C++ ";
    std :: cout << "to show the main structure." << std :: endl;
    std :: cout << "We learn more about the language ";
    std :: cout << "in this chapter and the rest of the book.";
    return 0;
}
```

Run:

```
This is a simple program in C++ to show the main structure.
We learn more about the language later in the book.
```

Program Background

Case Sensitivity

C++ language is case sensitive. The term that names an entity needs to be used as it is defined without changing the case of its letters.

Program Lines

Line numbers are not part of the program and should not be included in the source code.
They are used in this book for reference to each line.

Indention

Line indention improves the readability of a program and we strongly recommend it.

First Simple Program Line-by-Line Analysis Lines 1-10

Line1: Preprocessor Directive

```
#include <iostream>
```

```
#include <stdio.h> // C style
```

No semicolon should be put after any include directive.
The compiler may generate an error if it finds one.

// for standard input and output libraries.
E.g. cout, cin, istream, ostream, iostream …

Line3: Function Header

```
int main()
```

```
void main() { // not standard
    ............
}
```

Execution of each C++ program starts with the *main* function, which means that each program must have one function named *main*.

First Simple Program Line-by-Line Analysis Line 6

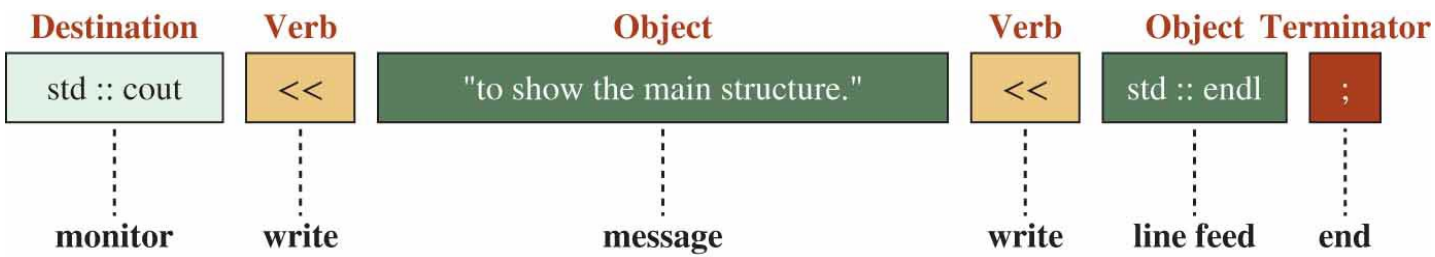
Lines 6: The Second Line of Body

```
std :: cout << "to show the main structure. "  << std :: endl;
```

printf("text\n"); // C style

std::cout << "Hello" << **'\n'**;
std::cout << "Hello" << **std::endl**;

Figure 2.2 Analysis of line 6




```
int n=3;
char c='#';
std::cout << c << 5.5 << '-' << n << "hello" << true;
```

```
std::cout << "n + 5 =" << n + 5;
std::cout << f(); // return value of the function f().
```

First Simple Program Line-by-Line Analysis Lines 7-9

Lines 7 and 8: Similar to line 5 and 6

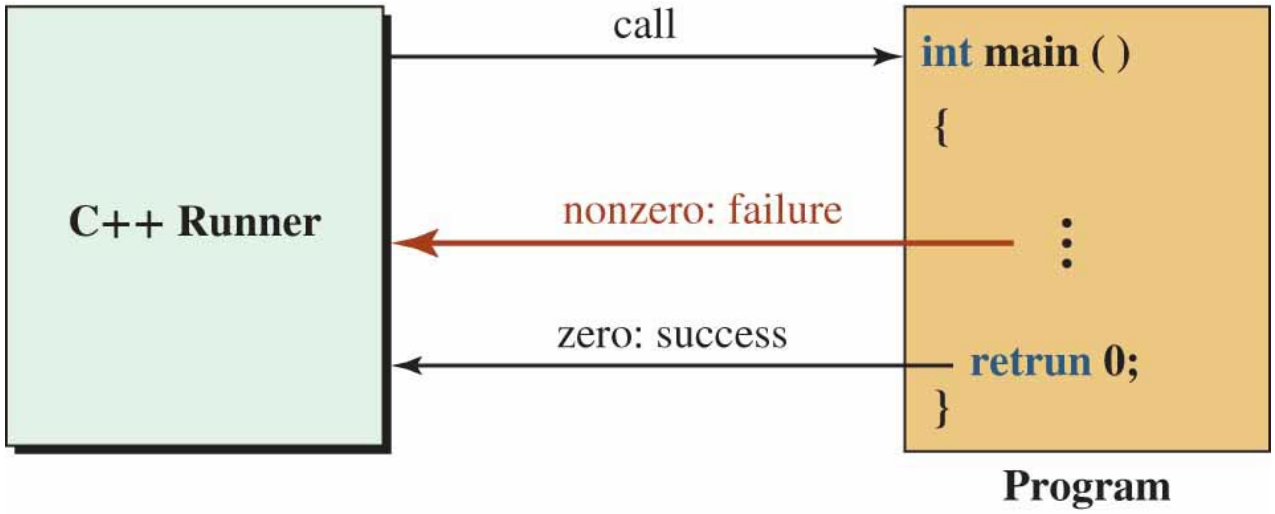
```
std :: cout << "We learn more about the language ";
std :: cout << "in this chapter and the rest of the book.";
```

Lines 9:

```
return 0;
```

```
int main() {
    ............
    // return 0; // can be omitted for the convenience.
}
```

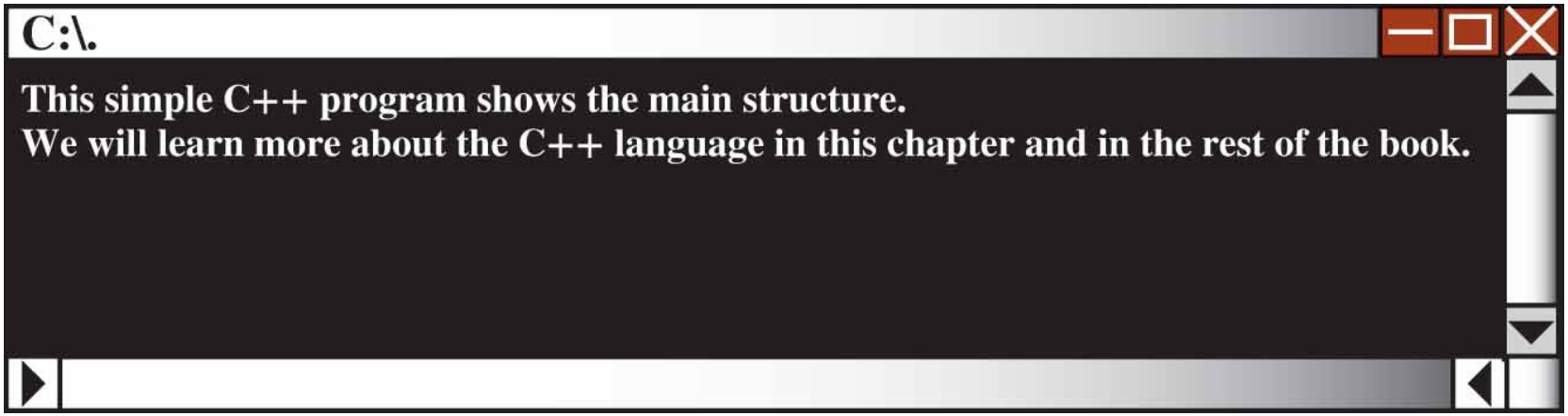
Figure 2.3 Relationship between runner and program



First Simple Program Output Analysis

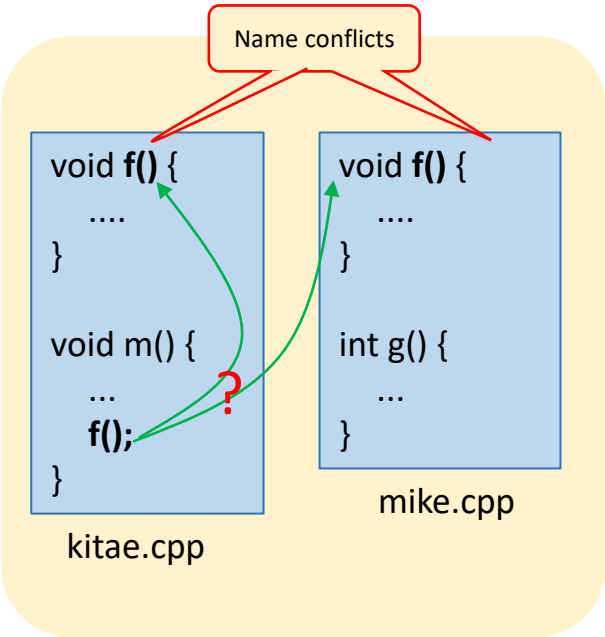
Program Output

Figure 2.4 *The console (cout object)*

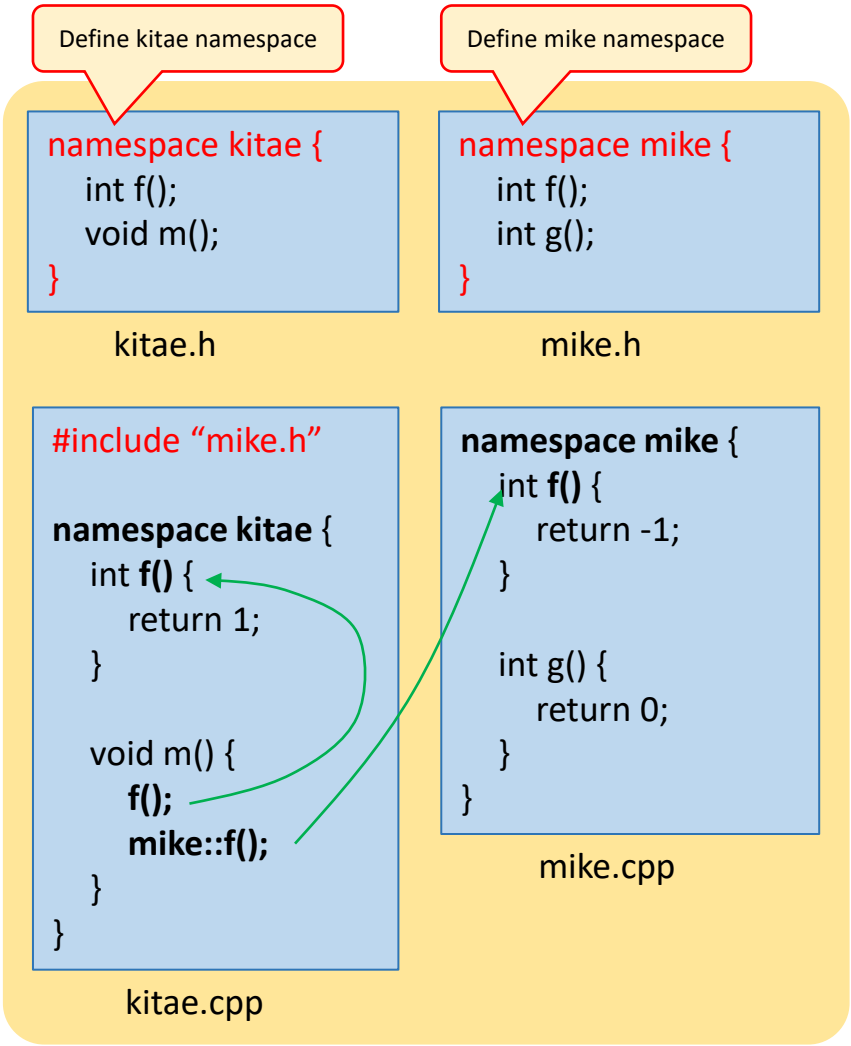


Namespace

- Identifier Conflicts
  - e.g. max(), std::max().
- Namespace keyword resolves it.



Build Error! (compile error)



The name conflict resolved!

std::

- One of the namespaces in C++ standard
  - cout, cin, endl in <iostream> header
- Usage
  - std::cout, std:: cin, std:: endl
- Omitting std::
  - Use **using** directive

```
using std::cout; // omitting std:: only for cout
.................................
cout << "Hello" << std::endl;
```

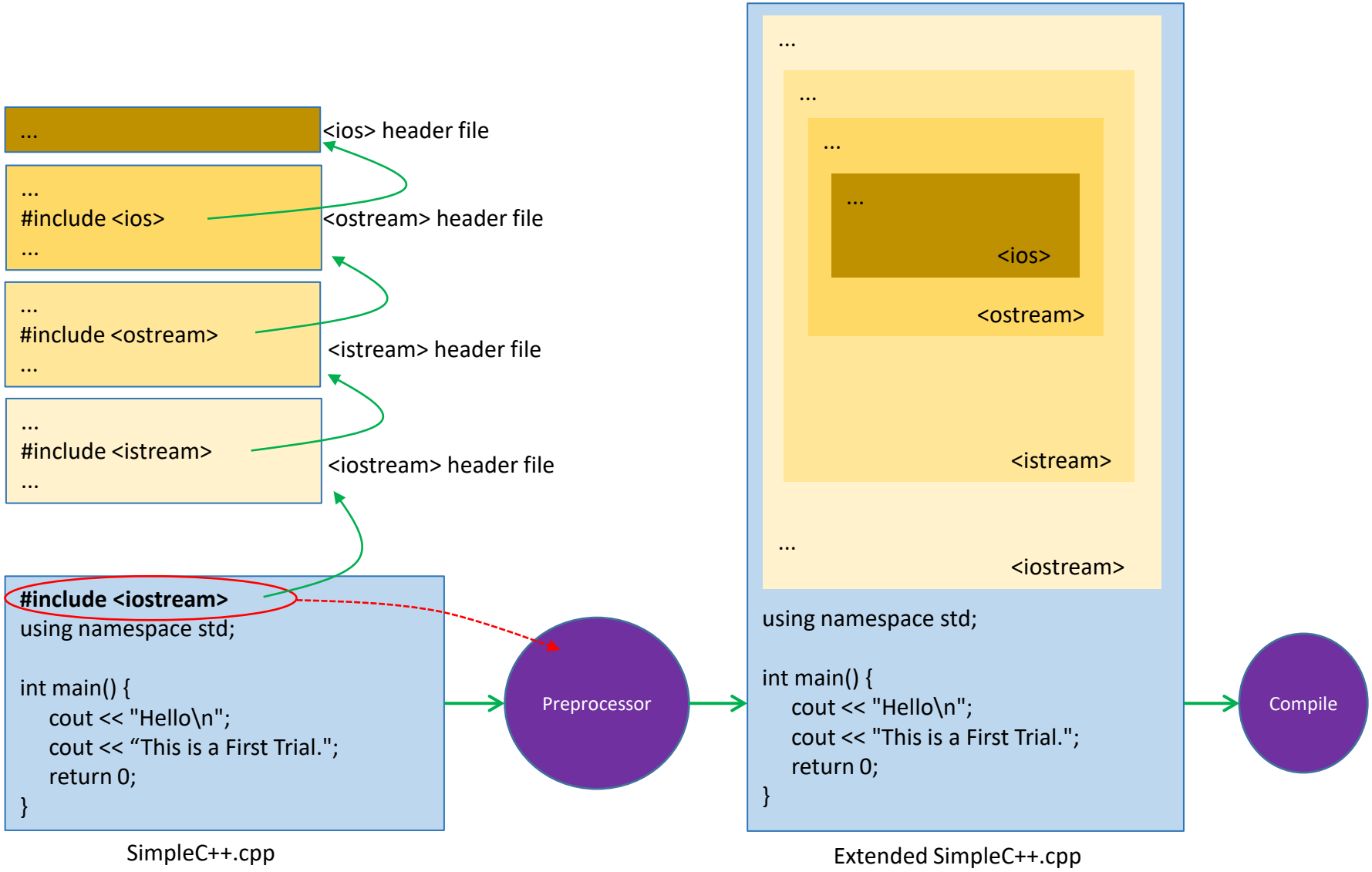
Omitting std::

```
using namespace std; // Omitting std:: for all names declared in the std namespace
.................................
cout << "Hello" << endl;
```

Omitting std::

Omitting std::

Preprocessor and #include <iostream>



#include <header file> v.s. #incluee "header file"

- Instruction to the location to find the header file.
  - #include <header file>
    - ✓ Instructs the compiler to look for the 'header file' in the folder where the compiler is installed.
    - ✓ *e.g. #include <string>, #include <iostream>*
  - #include "header file"
    - ✓ Instructs the compiler to look for the 'header file' in the developer's project folder of the include folder specified by the developer as a compile option.
    - ✓ *e.g. #include "myfile1.h", #include "myfile2.h"*

Second Simple Program

Program 2.2 *The second simple program*

```
/*************************************************************
 * This program shows how we can print a square of asterisks.   *
 *************************************************************/
#include <iostream>
using namespace std;
int main ()

{
    // Printing a square of asterisks
    cout << "******" << endl;
    cout << "******" << endl;
    cout << "******" << endl;
    cout << "******" << endl;
    cout << "******" << endl;
    cout << "******";
    return 0;
}
```

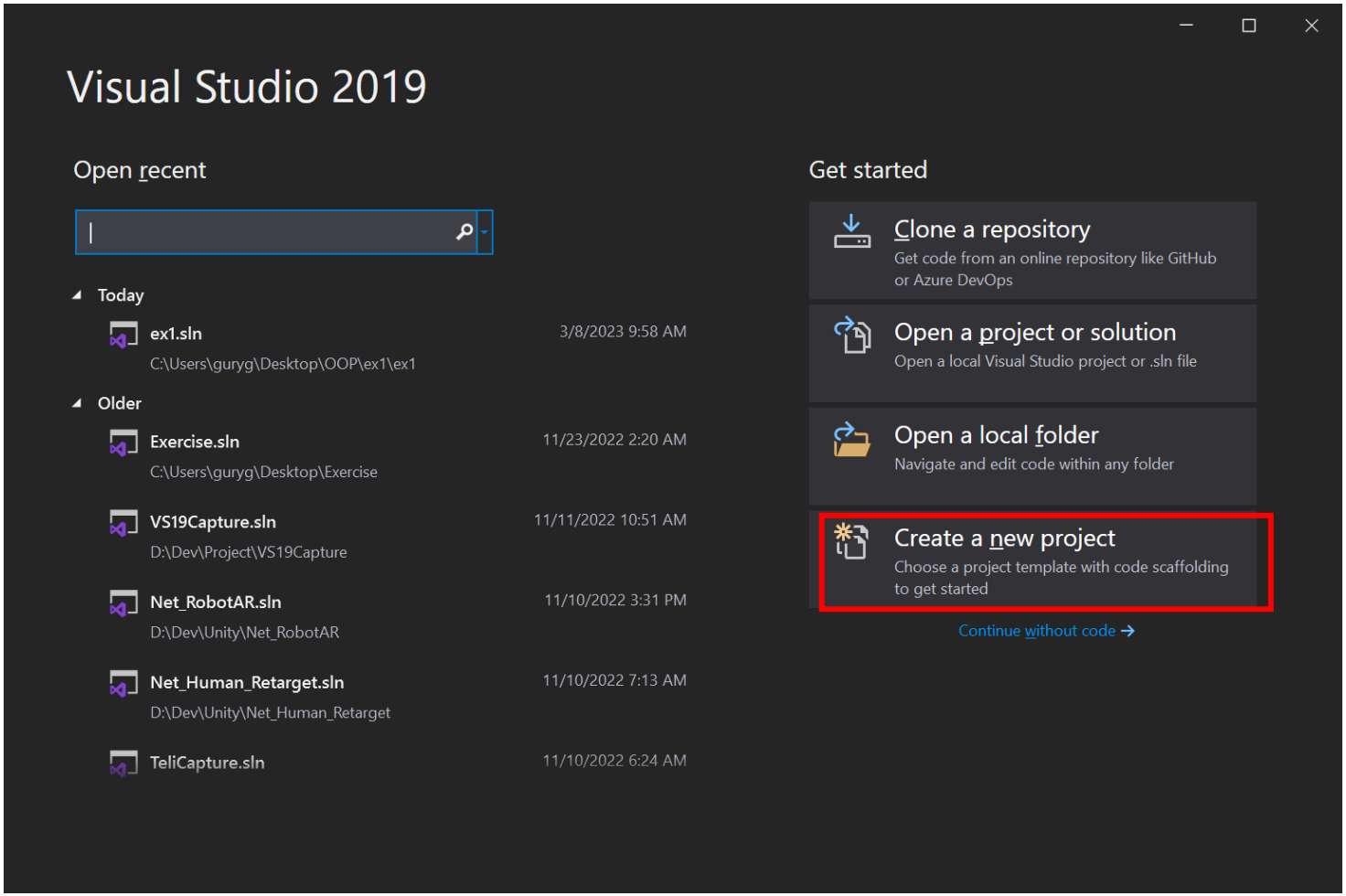
Output Second Simple Program

Program 2.2 *The second simple program*

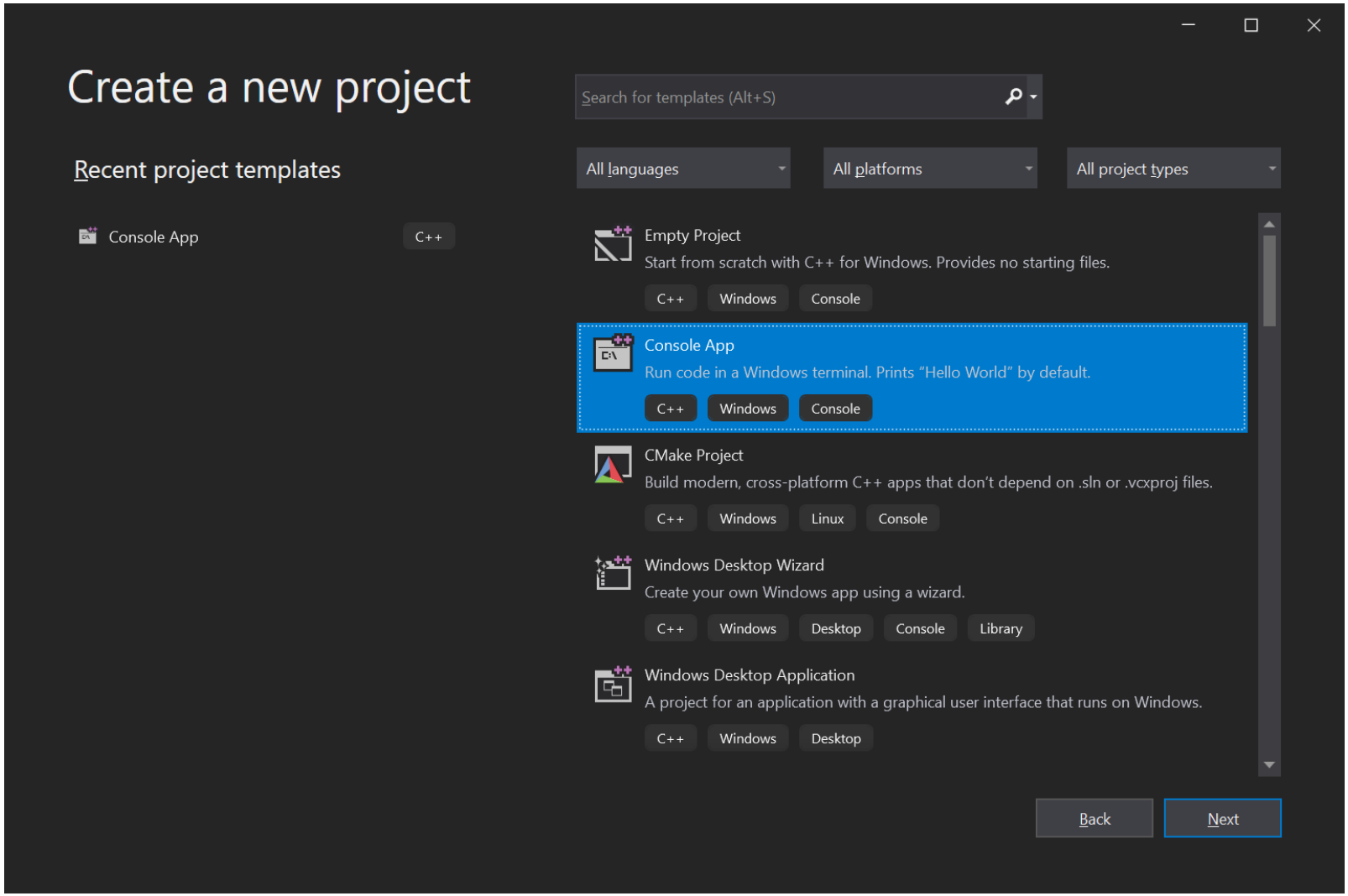
```
Run:
******
******
******
******
******
******
```

Lab: Using Visual Studio

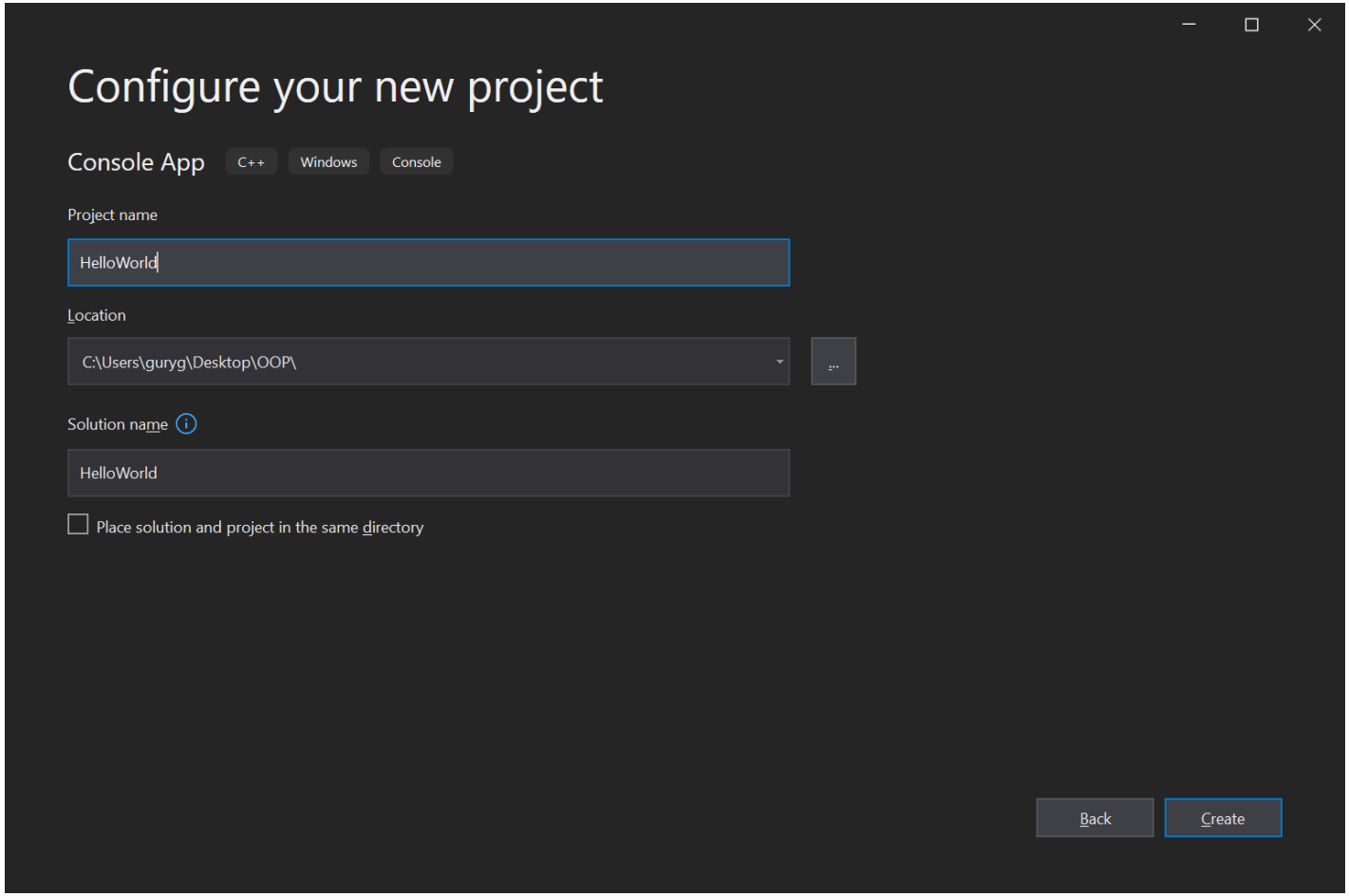
Creating a New Project



Creating a New Project



Creating a New Project



Creating a New Project

File Edit View Git Project Build Debug Test Analyze Tools Extensions Window Help Search (Ctrl+Q) HelloWorld

Debug x86 Local Windows Debugger Live Share

HelloWorld.cpp

HelloWorld (Global Scope)

```
// HelloWorld.cpp : This file contains the 'main' function. Program execution begins and e
//

#include <iostream>

int main()
{
    std::cout << "Hello World!\n";
}

// Run program: Ctrl + F5 or Debug > Start Without Debugging menu
// Debug program: F5 or Debug > Start Debugging menu

// Tips for Getting Started:
//   1. Use the Solution Explorer window to add/manage files
//   2. Use the Team Explorer window to connect to source control
//   3. Use the Output window to see build output and other messages
//   4. Use the Error List window to view errors
//   5. Go to Project > Add New Item to create new code files, or Project > Add Existing I
//   6. In the future, to open this project again, go to File > Open > Project and select
```

130 % No issues found Ln: 1 Ch: 1 SPC CRLF

Output

Show output from:

Error List Output Find Symbol Results

Solution Explorer

Search Solution Explorer (Ctrl+;)

Solution 'HelloWorld' (1 of 1 project)

HelloWorld

References

External Dependencies

Header Files

Resource Files

Source Files

Solution E... Git Changes Property...

Creating project 'HelloWorld'... project creation successful.

Add to Source Control

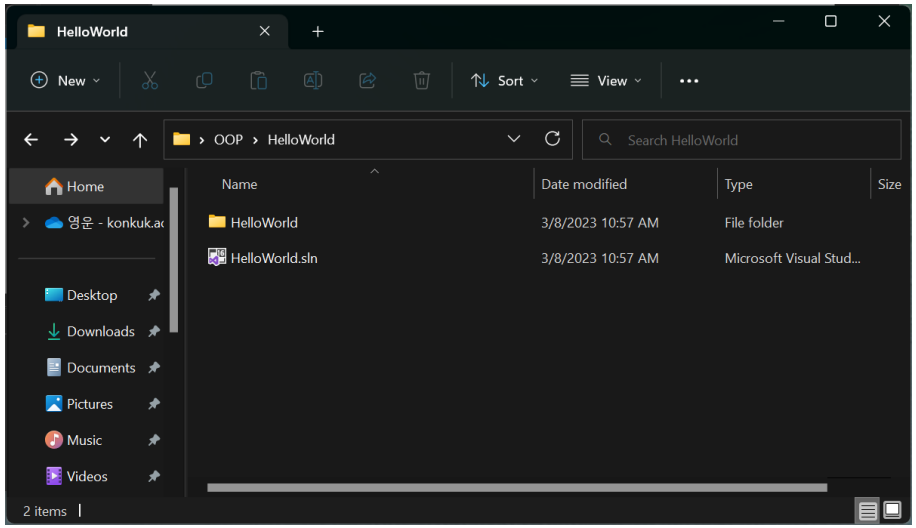
Creating a New Project

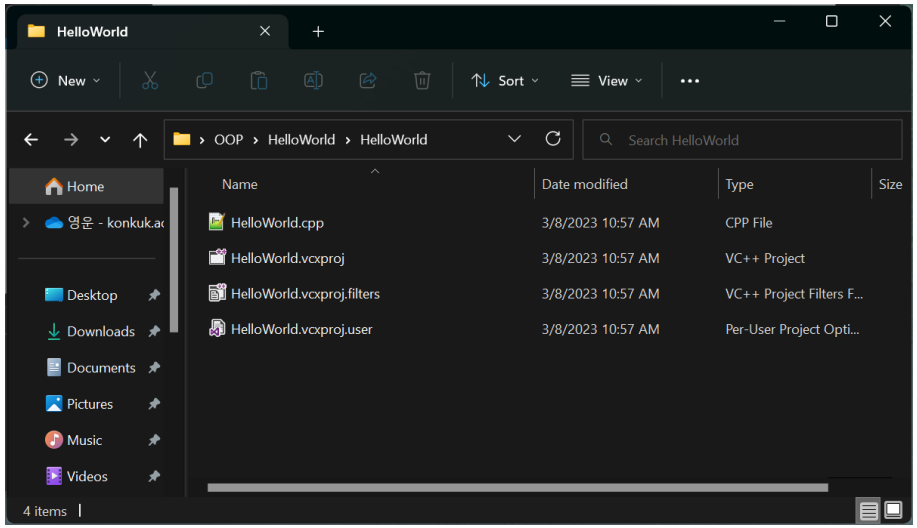
Type the code here

```
#include <iostream>

int main()
{
    std::cout << "This is a simple program in C++ ";
    std::cout << "to show the main structure." << std::endl;
    std::cout << "We learn more about the language ";
    std::cout << "in this chapter and the rest of the book.";
    return 0;
}
```

Creating a New Project





Build & Run

File Edit View Git Project Build Debug Test Analyze Tools Extensions Window Help Search (Ctrl+Q) HelloWorld

Debug x86 Local Windows Debugger Live Share

HelloWorld.cpp*

HelloWorld (Global Scope) main()

```
#include <iostream>

int main()
{
    std::cout << "This is a simple program in C++ ";
    std::cout << "to show the main structure." << std::endl;
    std::cout << "We learn more about the language ";
    std::cout << "in this chapter and the rest of the book.";
    return 0;
}
```

130 % No issues found

Output

Show output from:

Error List Output Find Symbol Results

Solution Explorer

Search Solution Explorer (Ctrl+;)

Solution 'HelloWorld' (1 of 1 project)

HelloWorld

References

External Dependencies

Header Files

Resource Files

Source Files

HelloWorld.cpp

Solution E... Git Changes Property...

Build

Clean

View

Analyze and Code Cleanup

Project Only

Retarget Projects

Train IntelliCode Poirot Model

Scope to This

New Solution Explorer View

Build Dependencies

Add

Class Wizard... Ctrl+Shift+X

Manage NuGet Packages...

Set as Startup Project Alt+S

Debug

Cut Ctrl+X

Paste Ctrl+V

Remove Del

Rename F2

Unload Project

Load Direct Dependencies of Project

Load Entire Dependency Tree of Project

Rescan Solution

Display Browsing Database Errors

Clear Browsing Database Errors

Open Folder in File Explorer

Open in Terminal

Nsight User Properties

Properties Alt+Enter

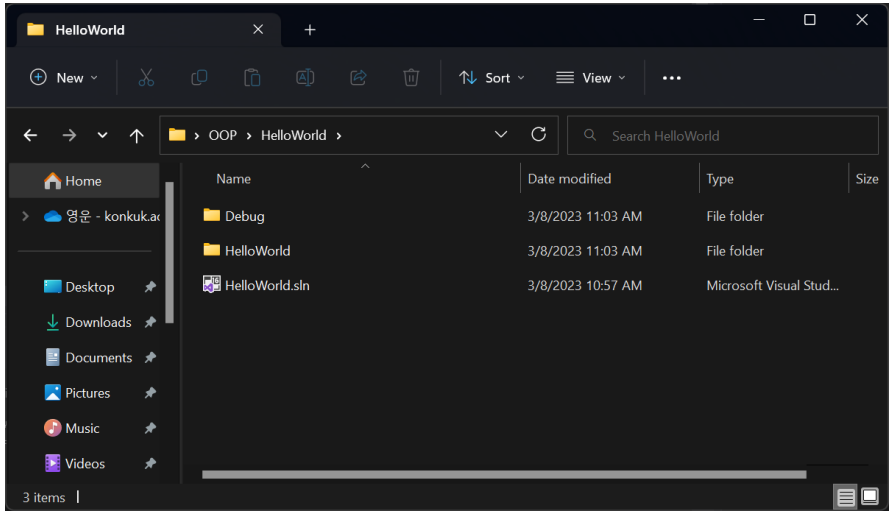
This item does not support previewing Add to Source Control

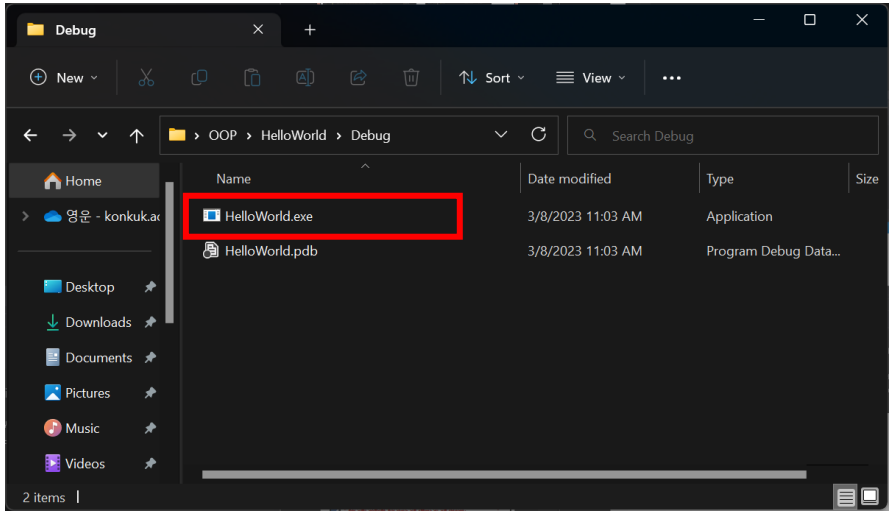
Build & Run

Compile result

```
Output
Show output from: Build
Build started...
1>------ Build started: Project: HelloWorld, Configuration: Debug Win32 ------
1>HelloWorld.cpp
1>HelloWorld.vcxproj -> C:₩Users₩guryg₩Desktop₩OOP₩HelloWorld₩Debug₩HelloWorld.exe
========== Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped ==========
```

Executable file here!





Build & Run

File Edit View Git Project Build Debug Test Analyze Tools Extensions Window Help

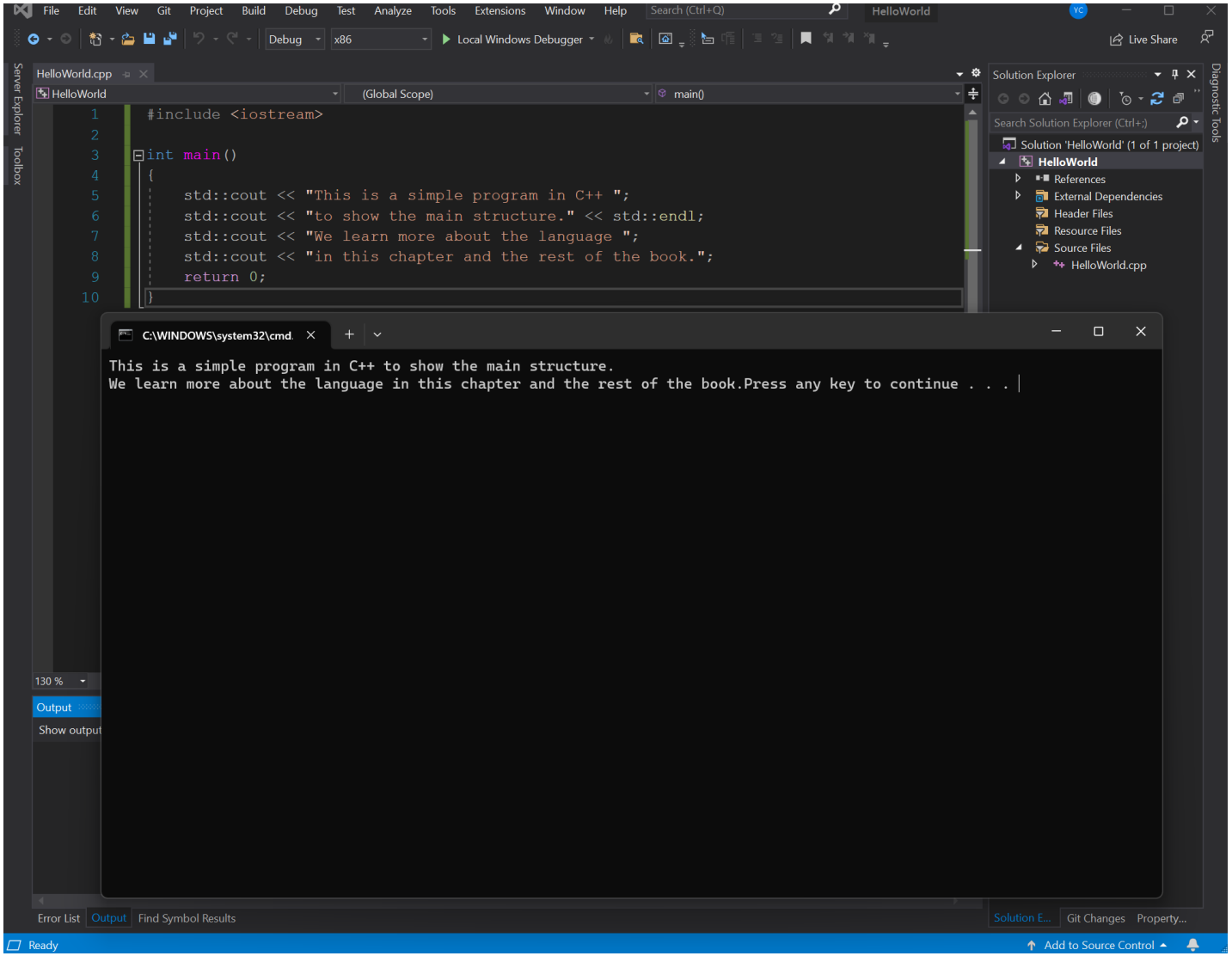
Windows
Graphics
Start Debugging F5
Start Without Debugging Ctrl+F5
Attach Unity Debugger
Apply Code Changes Alt+F10
Performance Profiler... Alt+F2
Relaunch Performance Profiler Shift+Alt+F2
Launch Python Profiling...
Attach to Process... Ctrl+Alt+P
Other Debug Targets
Step Into F11
Step Over F10
Toggle Breakpoint F9
New Breakpoint
Delete All Breakpoints Ctrl+Shift+F9
Clear All DataTips
Export DataTips ...
Import DataTips ...
Options...
HelloWorld Debug Properties

```
his is a simple program in C++ ";
o show the main structure." << std::endl;
e learn more about the language ";
n this chapter and the rest of the book.";
```

```
1>------ Rebuild All started: Project: HelloWorld, Configuration: Debug Win32 ------
1>HelloWorld.cpp
1>HelloWorld.vcxproj -> C:¥Users¥guryg¥Desktop¥OOP¥HelloWorld¥Debug¥HelloWorld.exe
========== Rebuild All: 1 succeeded, 0 failed, 0 skipped ==========
```

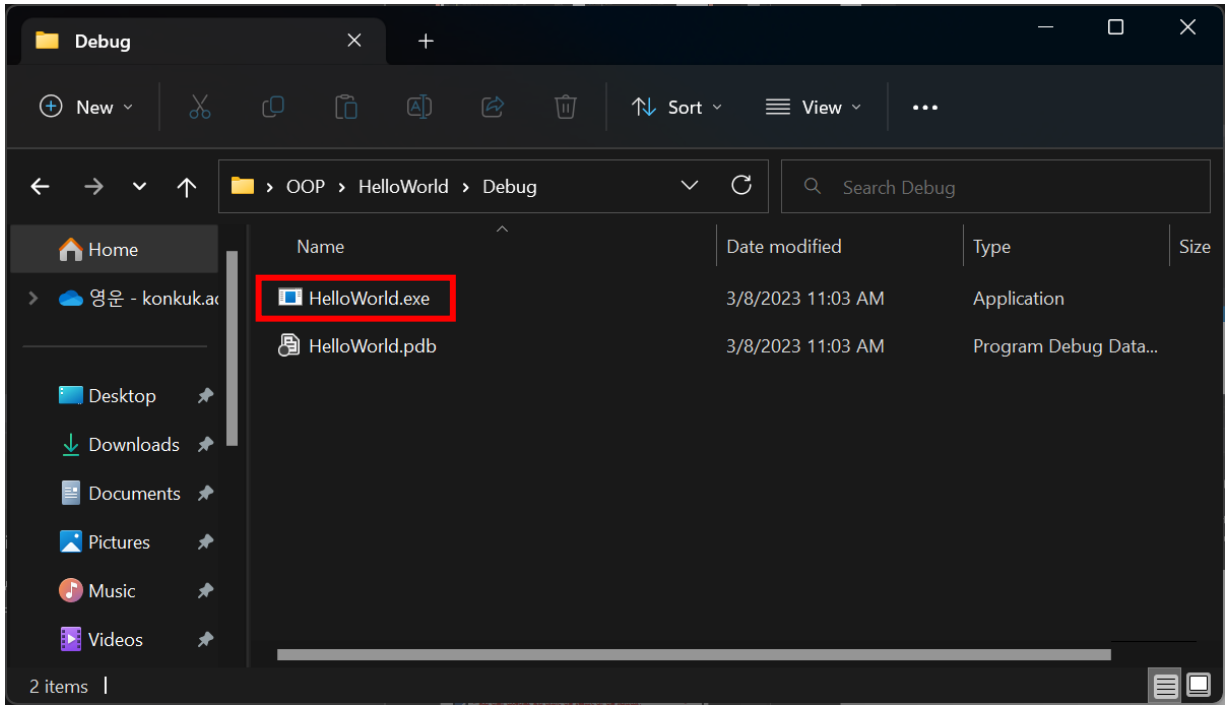
Solution Explorer
Solution 'HelloWorld' (1 of 1 project)
HelloWorld
References
External Dependencies
Header Files
Resource Files
Source Files
HelloWorld.cpp

Build & Run


File Edit View Git Project Build Debug Test Analyze Tools Extensions Window Help
Search (Ctrl+Q)
HelloWorld
Debug
x86
Local Windows Debugger
Live Share
HelloWorld.cpp
HelloWorld
(Global Scope)
main()
#include <iostream>
int main()
{
std::cout << "This is a simple program in C++ ";
std::cout << "to show the main structure." << std::endl;
std::cout << "We learn more about the language ";
std::cout << "in this chapter and the rest of the book.";
return 0;
}
Solution Explorer
Search Solution Explorer (Ctrl+;)
Solution 'HelloWorld' (1 of 1 project)
HelloWorld
References
External Dependencies
Header Files
Resource Files
Source Files
HelloWorld.cpp
C:\WINDOWS\system32\cmd.
This is a simple program in C++ to show the main structure.
We learn more about the language in this chapter and the rest of the book.Press any key to continue . . .
130 %
Output
Show output
Error List
Output
Find Symbol Results
Solution E...
Git Changes
Property...
Ready
Add to Source Control

Build & Run

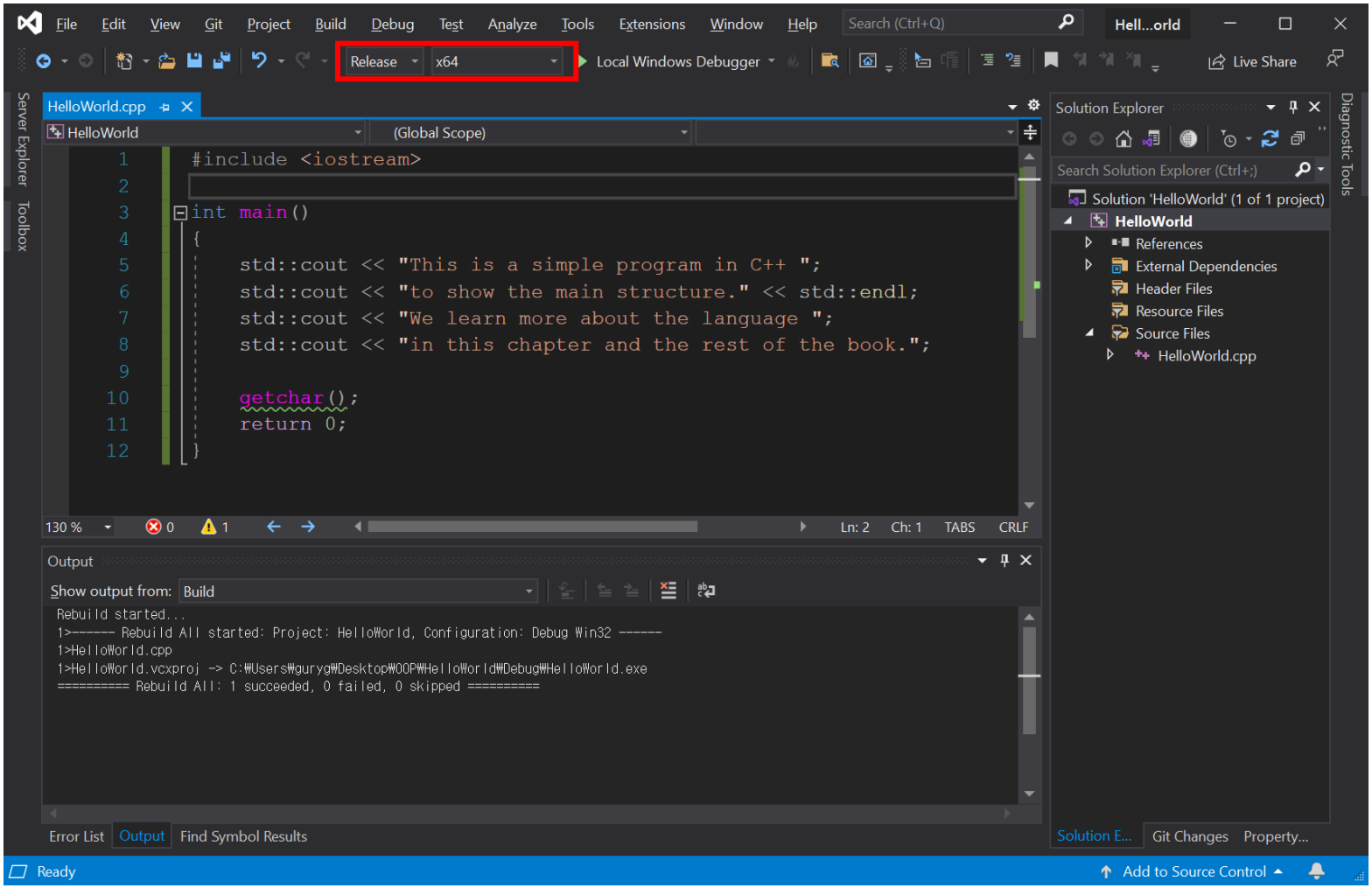
Try to run the exe file directly !



What happens ?

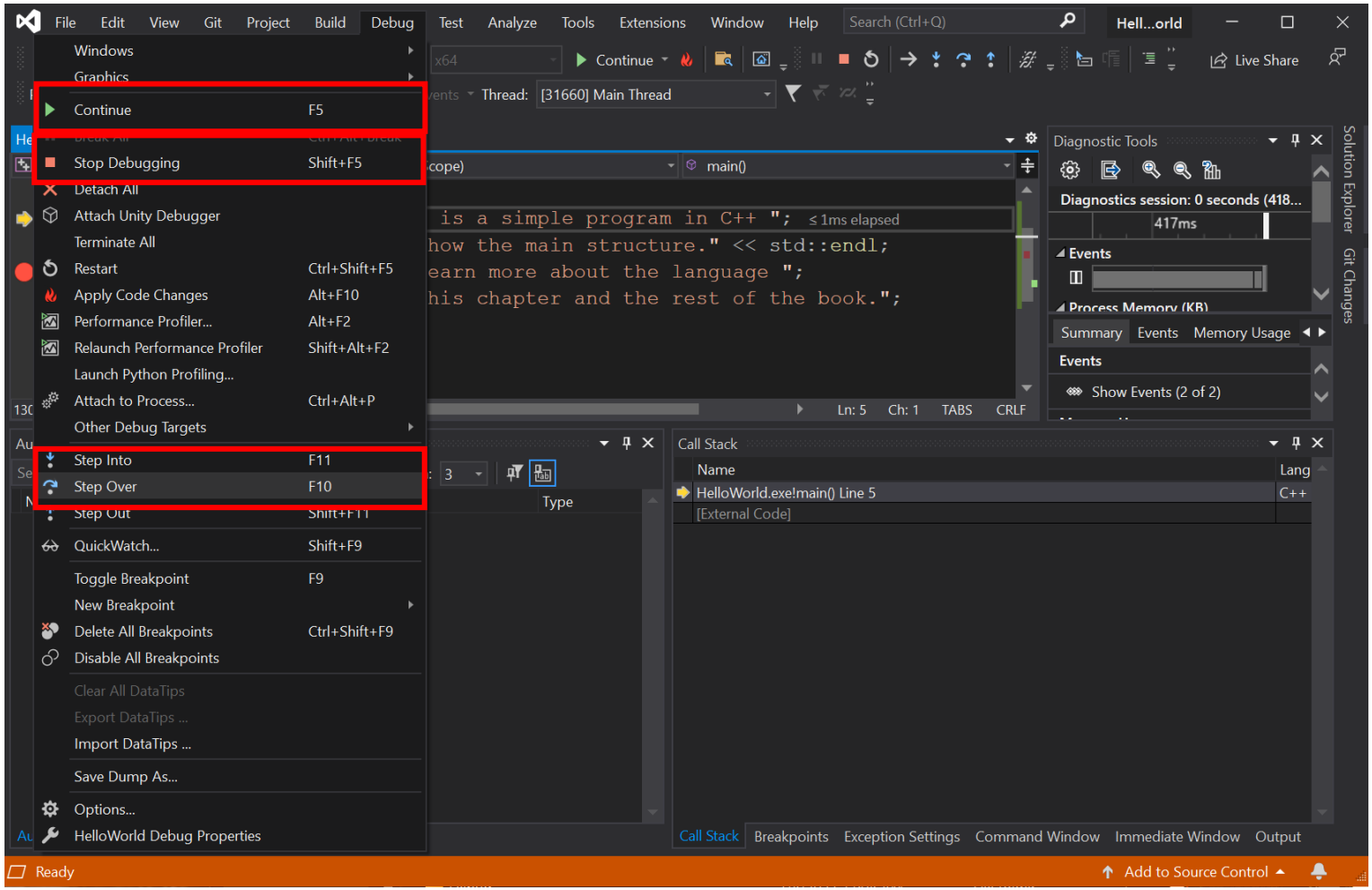
Build & Run

Change the “Build mode” and the “Platform”.

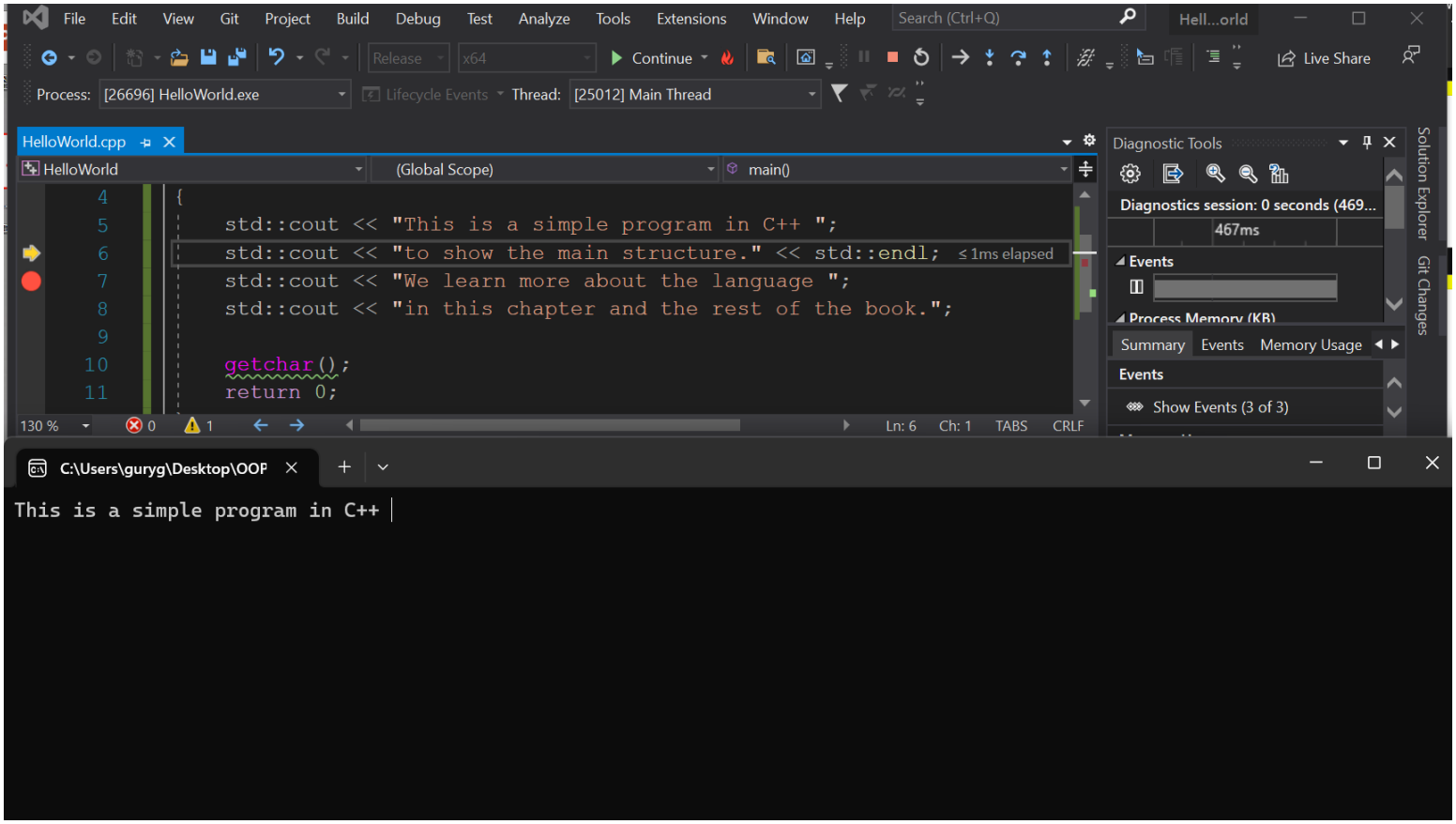


What happens ?

Debugging


File Edit View Git Project Build Debug Test Analyze Tools Extensions Window Help
Search (Ctrl+Q)
Hell...orld
Windows
Graphics
Continue F5
Stop Debugging Shift+F5
Detach All
Attach Unity Debugger
Terminate All
Restart Ctrl+Shift+F5
Apply Code Changes Alt+F10
Performance Profiler... Alt+F2
Relaunch Performance Profiler Shift+Alt+F2
Launch Python Profiling...
Attach to Process... Ctrl+Alt+P
Other Debug Targets
Step Into F11
Step Over F10
Step Out Shift+F11
QuickWatch... Shift+F9
Toggle Breakpoint F9
New Breakpoint
Delete All Breakpoints Ctrl+Shift+F9
Disable All Breakpoints
Clear All DataTips
Export DataTips ...
Import DataTips ...
Save Dump As...
Options...
HelloWorld Debug Properties
Continue
Live Share
Thread: [31660] Main Thread
main()
is a simple program in C++ ";
how the main structure." << std::endl;
earn more about the language ";
his chapter and the rest of the book.";
Diagnostic Tools
Diagnostics session: 0 seconds (418...
417ms
Events
Process Memory (KB)
Summary Events Memory Usage
Show Events (2 of 2)
Call Stack
Name
Lang
HelloWorld.exe!main() Line 5
C++
[External Code]
Type
Call Stack Breakpoints Exception Settings Command Window Immediate Window Output
Ready
Add to Source Control

Debugging


File Edit View Git Project Build Debug Test Analyze Tools Extensions Window Help
Search (Ctrl+Q)
Release x64 Continue Live Share
Process: [26696] HelloWorld.exe Lifecycle Events Thread: [25012] Main Thread
HelloWorld.cpp
HelloWorld (Global Scope) main()
{
std::cout << "This is a simple program in C++ ";
std::cout << "to show the main structure." << std::endl; ≤ 1ms elapsed
std::cout << "We learn more about the language ";
std::cout << "in this chapter and the rest of the book.";
getchar();
return 0;
Diagnostic Tools
Diagnostics session: 0 seconds (469...
467ms
Events
Process Memory (KB)
Summary Events Memory Usage
Events
Show Events (3 of 3)
Solution Explorer Git Changes
130 % 0 1 Ln: 6 Ch: 1 TABS CRLF
C:\Users\guryg\Desktop\OOP
This is a simple program in C++

Exercise

- Correct the runtime error in the code using debugging.

```
#include <iostream>

int main()
{
    int num1 = 10;
    int num2 = 0;

    int num3 = num1 + num2;
    int num4 = num1 - num2;
    int num5 = num1 * num2;
    int num6 = num1 / num2;

    std::cout << num3 << std::endl;
    std::cout << num4 << std::endl;
    std::cout << num5 << std::endl;
    std::cout << num6 << std::endl;

    getchar();
    return 0;
}
```

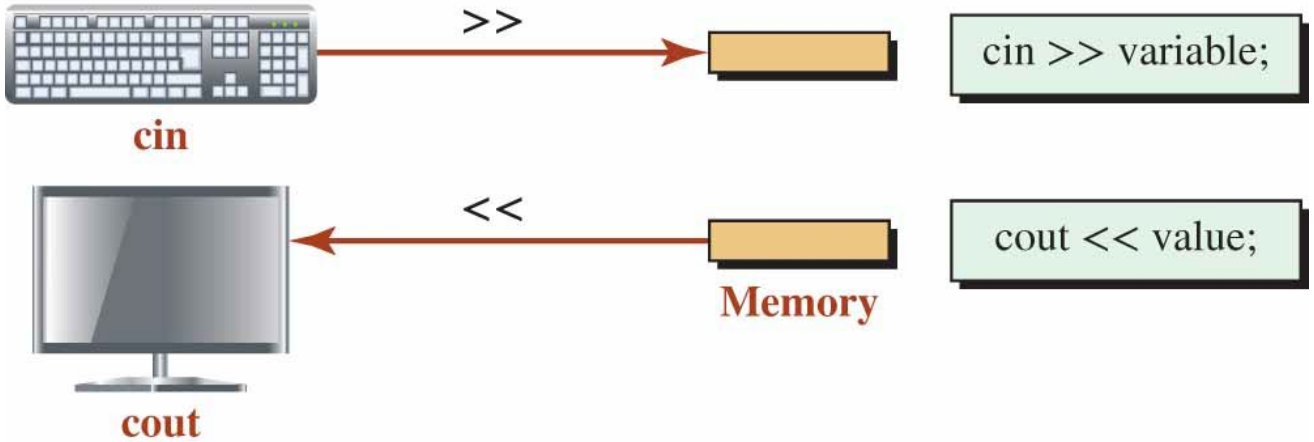
```
1>------ Rebuild All started: Project: HelloWorld, Configuration: Release x64 ------
1>HelloWorld.cpp
1>Generating code
1>Previous IPDB not found, fall back to full compilation.
1>C:₩Users₩guryg₩Desktop₩OOP₩HelloWorld₩HelloWorld₩HelloWorld.cpp(11): warning C4723: potential divide by 0
1>All 2 functions were compiled because no usable IPDB/IOBJ from previous compilation was found.
1>Finished generating code
1>HelloWorld.vcxproj -> C:₩Users₩guryg₩Desktop₩OOP₩HelloWorld₩x64₩Release₩HelloWorld.exe
1>Done building project "HelloWorld.vcxproj".
========== Rebuild All: 1 succeeded, 0 failed, 0 skipped ==========
```

C++ Basics

The cin and cout Object

The *cin* object needs to see a variable;
the *cout* object needs to see a value.

Figure 2.9 ***Keyboard and monitor as source and destination***



Third Simple Program

Program 2.3 *A program that adds two values*

```
/*****************************************************************
 * This program get the values for two numbers from the keyboard,     *
 * add them together and print the result on the monitor.             *
 *****************************************************************/
#include <iostream>
using namespace std;

int main ()
{
     // Definition
     int num1;
     int num2;
     int sum;
     // Getting inputs
     cout << "Enter the first number: ";
     cin >> num1;
     cout << "Enter the second number: ";
     cin >> num2;
     // Calculation and storing result
     sum = num1 + num2;
```

Third Simple Program Output

Program 2.3 *A program that adds two values*

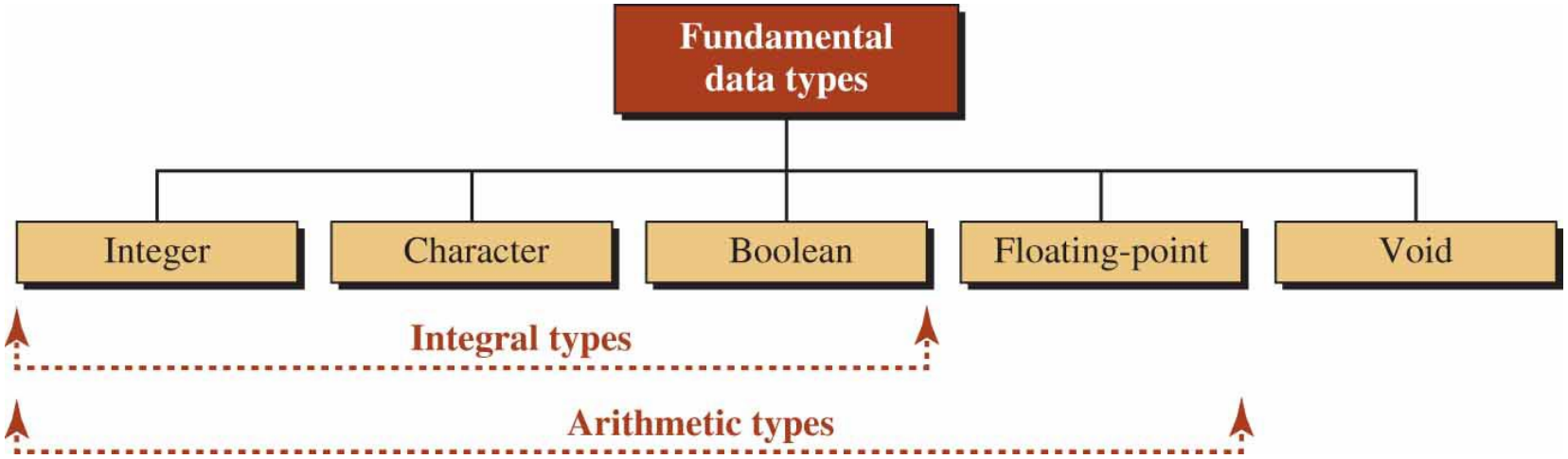
```
    // Display output
    cout << "The sum is: " << sum;
    return 0;
}
```

```
Run:
Enter the first number: 23
Enter the second number: 35
The sum is: 58
```

```
Run:
Enter the first number: 7
Enter the second number: 110
The sum is: 117
```

DATA TYPES

Figure 2.12 ***Fundamental Data Types***



Integer Data Type

Table 2.4 ***Ranges of integers in a typical machine machine***

Type	Sign	Range	
short int	signed	−32,768	+32,767
	unsigned	0	65,536
int	signed	−2,147,483,648	+2,147,483,647
	unsigned	0	4,294,967,295
long int	signed	−2,147,483,648	+2,147,483,647
	unsigned	0	4,294,967,295

Size Of Unsigned Integers

Program 2.6 *Finding size of integer types*

```
/*************************************************************
 * A program to find the size of all three integer types      *
 *************************************************************/
#include <iostream>
using namespace std;

int main ()
{
     cout << "Size of short int is" << sizeof (short int) << "bytes." << endl;
     cout << "Size of int is" << sizeof (int) << "bytes." << endl;
     cout << "Size of long int is " << sizeof (long int) << " bytes. " << endl;
     return 0;
}
```

Run:

```
Size of short int: 2 bytes.
Size of int: 4 bytes.
Size of long int: 4 bytes.
```

Initialization With Integer Literals

Program 2.7 *Initialization with integer literals*

```
/*****************************************************************
 * Using some literal values as variable initializers          *
 *****************************************************************/
#include <iostream>
using namespace std;

int main ()
{
    // Declaration and initialization
    int x = -1245;
    unsigned int y = 1245;
    unsigned int z = -2367;
    unsigned int t = 14.56;
    // Outputting initialized values
    cout << x << endl;
    cout << y << endl;
    cout << z << endl;
    cout << t;
```

Integer Literals Output

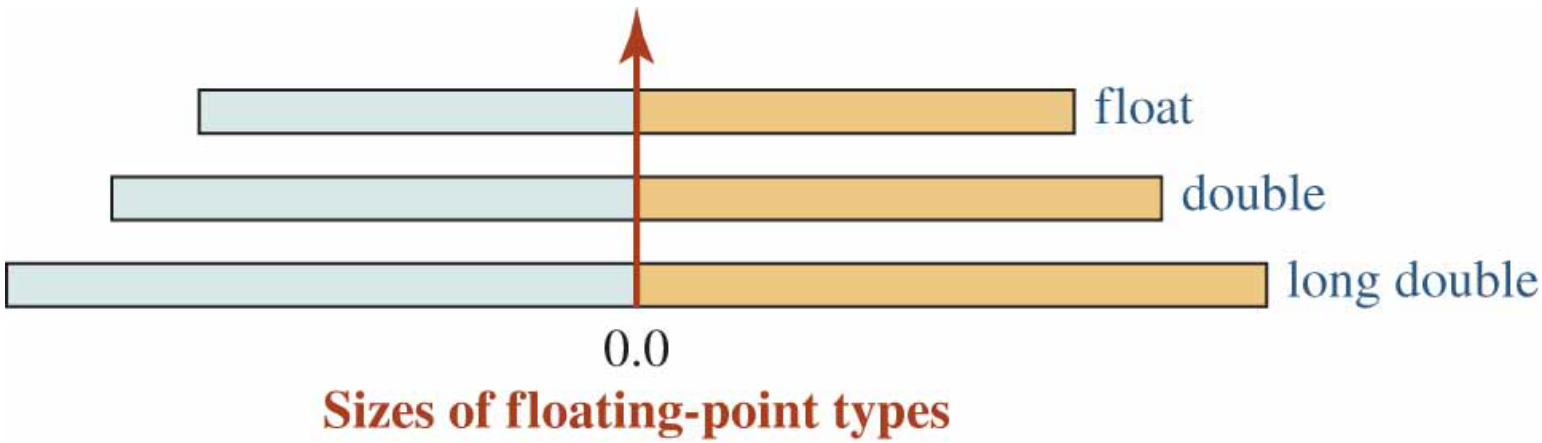
Program 2.7 *Initialization with integer literals*

```
    return 0;
}
```

```
Run:
Value of x: -1245 // OK.
Value of y: 1245  // OK.
Value of z: 4294964929 // Logical error. A negative value is changed to
positive.
Value of t: 14 // The value is truncated.
```

Floating-Point Data Type Size

Figure 2.14 ***Relative size of floating-point type***



Floating-Point Variables

- **We can define floating-point variables in the same way that we have done with integers.**
- **We can initialize them when we declare them or we can assign values to them later.**

Size of Floating-Point Data Type

Table 2.6 ***Suffix to define size of literal***

Floating-point Type	*Suffixes*	*Example*
Float	f or F	12.23F, 12345.45F, –1436F
double	None	1425.36, 1234.34, 123454
long double	l or L	2456.23L, 143679.00004 L, -0.02345L

The default size of a floating-point literal is *double*.

Character Data Type

A character literal is always enclosed in a pair of single quotes.

Table 2.5 ***Some Special Characters***

Sequence	Description	Sequence	Description
\n	New line (line feed)	\f	Form feed
\t	Tab	\’	Single quote
\b	Backspace	\”	Double quote
\r	Carriage Return	\\	Backslash

C-String (C Style)

- C-string method – a character array terminated by '\0'.

C-string array

Not a string!

```
char name1[6] = {'G', 'r', 'a', 'c', 'e', '\0'}; // name1 is the "Grace" string.
char name2[5] = {'G', 'r', 'a', 'c', 'e'}; // name2 is just a character array, not a string.
```

```
char name5[10] = "Grace";
```

name5[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]
'G'	'r'	'a'	'c'	'e'	'\0'	'\0'	'\0'	'\0'	'\0'

"Grace" string

Initialized with '\0'

C-String (C Style)

- C-string method – a character array terminated by '\0'.

```
char name[6]; // A char array that can store 5 characters
cin >> name; // Reads a string from the keyboard and stores it in the name array
```

Apple

Key input

name [0]	[1]	[2]	[3]	[4]	[5]
'A'	'p'	'p'	'l'	'e'	'\0'

The string "Apple"

String Class (C++)

- **To use the C++ string, we need to include the header file <string> in our program as shown below:**

```
# include <string>
```

- **We declare a variable of type string using the following definition:**

```
string name;
```

Using String Class

Program 2.13 *Using string class*

```
/*****************************************************************
 * This program prints the full name of a person given the first, *
 * the middle, and the last name.                                 *
 *****************************************************************/
#include <iostream>
#include <string> // Need to use the string class
using namespace std;

int main ()
{
     // Defining variables
     string first;
     string initial;
     string last;
     string space = " ";
     string dot = ".";
     string fullName;
     // Input data for first name, initial, and last name
     cout << "Enter the first name: ";
     cin >> first;
```

String Class Output

Program 2.13 *Using string class*

```
21      cout << "Enter the initial: ";
22      cin >> middle;
23      cout << "Enter the last name: ";
24      cin >> last;
25      // Formation of full name using concatenation operator
26      fullName = first + space + initial + dot + space + last;
27      // Outputting full name
28      cout << "The full name is: " << fullName;
29      return 0;
30  }
```

```
Run:
Enter the first name: John
Enter the initial: A
Enter the last name: Brown
The full name is: John A. Brown
```

In-class Exercise

- **Given an array with <u>all distinct elements</u>, find the largest <u>two distinct</u> elements in an array.**
  - Define a function:
    - *print2largest(int arr[], int arr_len)*
    - *get2largest(int arr[], int arr_len, int * first, int * second)*
  - Test set:

```
Input: arr[] = {10, 4, 3, 50, 23, 90}
Output: 90, 50
```

```
Input: arr[] = {99, 77, 11, 15, 88, 1}
```

```
Input: arr[] = {10,9636, 2401, 777, 2080, 1, 50}
```

End of Class

Thank you

E-mail: youngcha@konkuk.ac.kr