Programming Essentials

(UOMU022023)

اساسيات البرمجة
2025-2024

Variables & Data Types

by
Dr Murtada Dohan
murtada.dohan@uomus.edu.iq


AL MUSTAQBAL UNIVERSITY
جامعة المستقبل
أسست عام
2010


كلية الهندسة والتقنيات
الهندسية
COLLEGE OF ENGINEERING &
TECHNOLOGY
AL MUSTAQBAL UNIVERSITY
جامعة المستقبل
2010

Recap

The Parts of a C++ Program

```
// sample C++ program
#include <iostream>
using namespace std;
int main()
{
      cout << "Hello, there!";
      return 0;
}
```

comment

preprocessor directive

which namespace to use

beginning of function named main

beginning of block for main

output statement

string literal

send 0to operating system

end of block for main

Hello World Program

```
//*************************************
// This program prints Hello World!
//*************************************
#include <iostream>
using namespace std;
int main ()
{
   cout << "Hello World!";
   return 0;
}
```

Special Characters

Character	Name	Meaning
//	Double slash	Beginning of a comment
#	Pound sign	Beginning of preprocessor directive
< >	Open/close brackets	Enclose filename in #include
( )	Open/close parentheses	Used when naming a function
{ }	Open/close brace	Encloses a group of statements
" "	Open/close quotation marks	Encloses string of characters
;	Semicolon	End of a programming statement

The `cout` Object

- Displays output on the computer screen
- You use the stream insertion operator << to send output to cout:

```
cout << "Programming is fun!";
```

The cout Object

- Can be used to send more than one item to cout:

```
cout << "Hello " << "there!";
```

Or:

```
cout << "Hello ";
cout << "there!";
```

The `cin` Object

- Standard input object
- Like `cout`, requires `iostream` file
- Used to read input from keyboard
- Information retrieved from cinwith >>
- Input is stored in one or more variables

The cin Object

Program 3-1

```
1  // This program asks the user to enter the length and width of
2  // a rectangle. It calculates the rectangle's area and displays
3  // the value on the screen.
4  #include <iostream>
5  using namespace std;
6
7  int main()
8  {
9     int length, width, area;
10
11    cout << "This program calculates the area of a ";
12    cout << "rectangle.\n";
13    cout << "What is the length of the rectangle? ";
14    cin >> length;
15    cout << "What is the width of the rectangle? ";
16    cin >> width;
17    area = length * width;
18    cout << "The area of the rectangle is " << area << ".\n";
19    return 0;
20 }
```

Program Output with Example Input Shown in Bold

```
This program calculates the area of a rectangle.
What is the length of the rectangle? 10 [Enter]
What is the width of the rectangle? 20 [Enter]
The area of the rectangle is 200.
```

The `cin` Object

- **`cin`** converts data to the type that matches the variable:

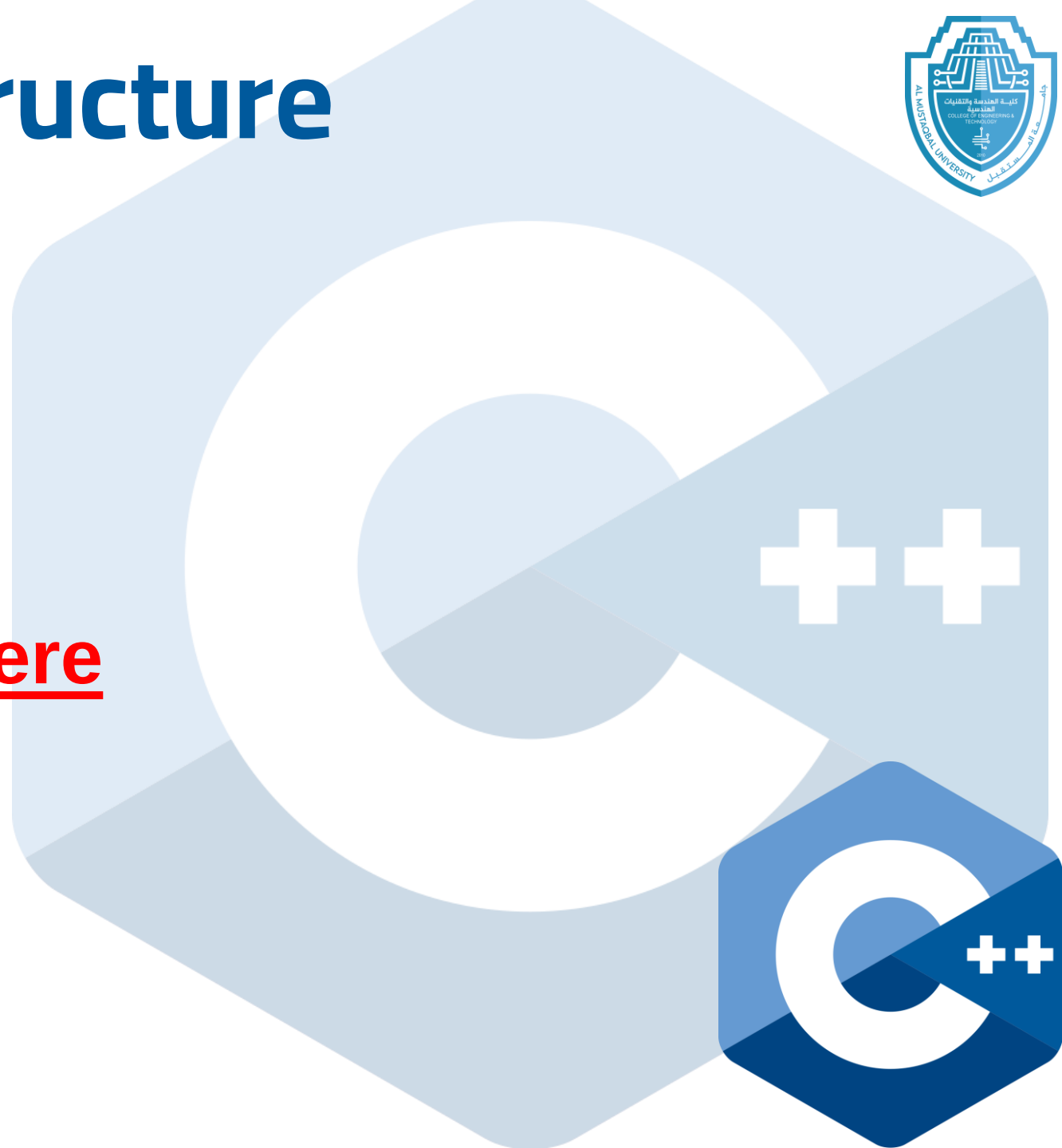
```
int height;
cout << "How tall is the room? ";
cin >> height;
```

Today Agenda

- C++ Program structure
- Variables
- Data Types
- Order of Operations
- Examples of reading Variables

C++ Program structure

```
1. #include <iostream>
2. using namespace std;
3. int main ()
4. {
5.       Write your code here
6.  return 0;
7.  }
```



What is a Variable?

- A variable is a named storage that holds data.
- Used to store values that can change during program execution.
- Every variable has:
  - A name
  - A type
  - A value

Syntax of Variable Definition

- Syntax of Variable Definition:
  - type variableName;

    or
  - type variableName = value;
- Example:
  - int age;
  - float salary = 5000.50;
  - char grade = 'A';

Data Type

Type	Minimum size (bits)	Minimum range of values (Depends upon the size on your platform)
char	8	-128 to 127
short	16	-32768 to 32767
long	32	-2147483648 to 2147483647
float	Often 32	Single precision (implementation defined) e.g. 23 bit mantissa, 8 bit exponent
double	Often 64	Double precision (implementation defined) e.g. 52 bit mantissa, 11 bit exponent
long double	≥ double	Extended precision, implementation defined
int	≥ short	varies

Variable Naming Rules

- Must begin with a **letter** or **underscore** (_)
- **Cannot** use keywords (e.g., int, return)
- **Case-sensitive** (score ≠ Score)
- Use meaningful names (int age; , not int a;)

Example of creating variables

Type	Description	Example
int	Integer numbers	int x = 5;
float	Decimal numbers	float y = 3.2;
double	Larger decimal numbers	double z = 7.456;
char	Single characters: A,a,B,b.. %,$,*: 0,1,..	char c = 'A';
bool	Boolean values	bool isTrue = true;

Binary Arithmetic Operators

SYMBOL	OPERATION	EXAMPLE	VALUE OF ans
+	addition	ans = 7 + 3;	10
-	subtraction	ans = 7 - 3;	4
*	multiplication	ans = 7 * 3;	21
/	division	ans = 7 / 3;	2
%	modulus	ans = 7 % 3;	1

A Closer Look at the /Operator

- / (division) operator performs integer division if both operands are integers

```
cout << 13 / 5;     // displays 2
cout << 91 / 7;     // displays 13
```

- If either operand is floating point, the result is floating point

```
cout << 13 / 5.0;   // displays 2.6
cout << 91.0 / 7;   // displays 13.0
```

A Closer Look at the % Operator

- %(modulus) operator computes the remainder resulting from integer division

```
cout << 13 % 5;     // displays 3
```

- %requires integers for both operands

```
cout << 13 % 5.0; // error
```

Mathematical Expressions

- Can create complex expressions using multiple mathematical operators
- An expression can be a literal, a variable, or a mathematical combination of constants and variables
- Can be used in assignment, cout, other statements:

```
area = 2 * PI * radius;
cout << "Area is: " << area;
```

Order of Operations

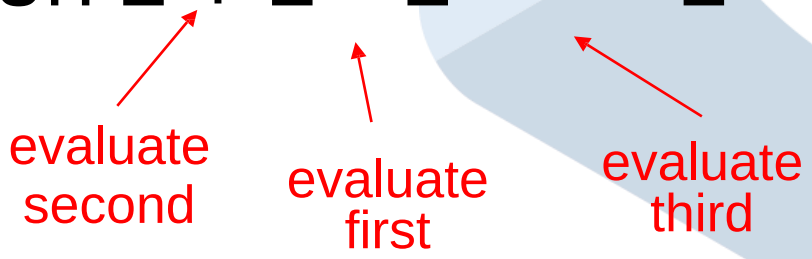
In an expression with more than one operator, evaluate in this order:

-(unary negation), in order, left to right

* / %, in order, left to right

+ -, in order, left to right

In the expression 2 + 2 * 2 – 2



Order of Operations

Table 3-2 Some Simple Expressions and Their Values

Expression	Value
`5 + 2 * 4`	13
`10 / 2 - 3`	2
`8 + 12 * 2 - 4`	28
`4 + 17 % 2 - 1`	4
`6 - 3 * 2 + 7 - 1`	6

Associativity of Operators

- -(unary negation) associates right to left
- *, /, %, +, - associate right to left
- parentheses ( ) can be used to override the order of operations:

```
 2 + 2     * 2 - 2     = 4
(2 + 2)    * 2 - 2     = 6
 2 + 2     * (2 - 2)   = 2
(2 + 2)    * (2 - 2)   = 0
```

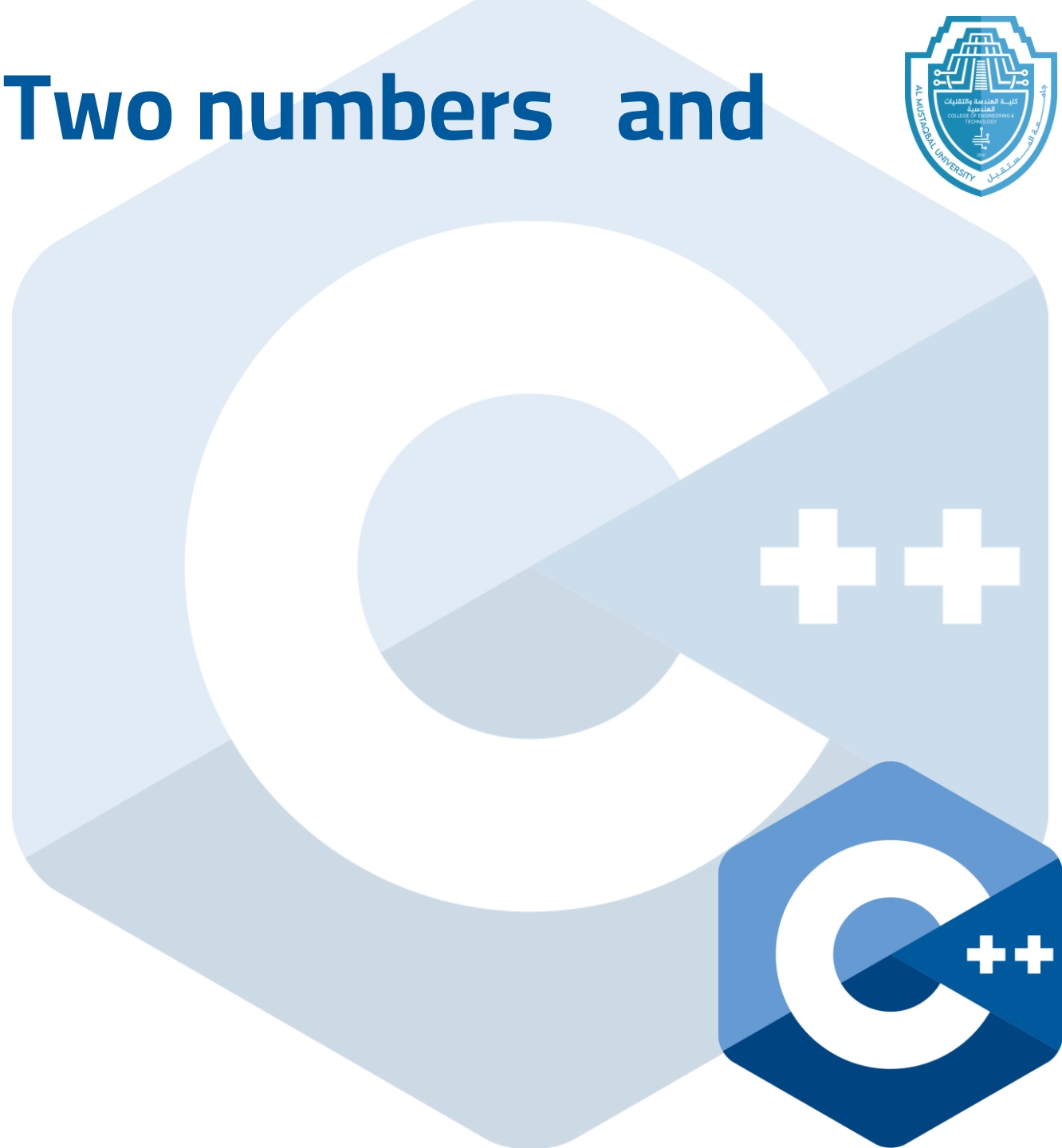
Grouping with Parentheses

Table 3-4 More Simple Expressions and Their Values

Expression	Value
`(5 + 2) * 4`	28
`10 / (5 - 3)`	5
`8 + 12 * (6 - 2)`	56
`(4 + 17) % 2 - 1`	0
`(6 - 3) * (2 + 7) / 3`	9

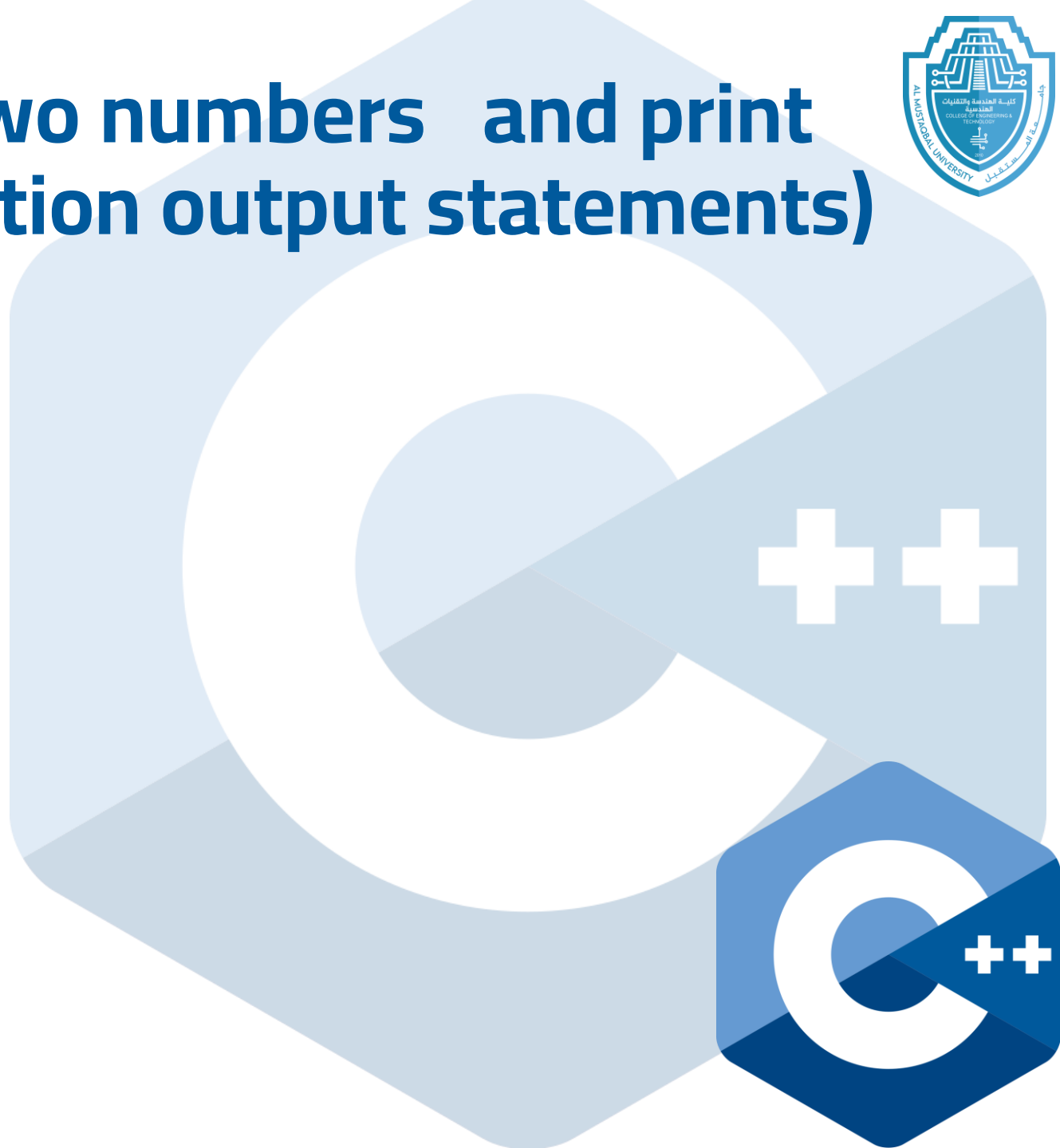
Example of Reading Two numbers and print the average

```
#include <iostream>
 using namespace std;
 int main ()
 {
int x;
int y;
cin>>x;
cin>>y;
int result= (x+y)/2;
cout<<results;
return 0;
  }
```



Example of Reading Two numbers and print the average (with addition output statements)

```
#include <iostream>
 using namespace std;
 int main ()
 {
 int x;
 int y;
 cout<<“Enter the value of x:”;
 cin>>x;
 cout<<“Enter the value of y:”;
 cin>>y;
 int result= (x+y)/2;
 cout<<“result=”<<results;
 return 0;
  }
```



Example of Reading Three numbers and implement the current equation the y=x*2+y*2+z*2

```
#include <iostream>
 using namespace std;
 int main ()
 {
int x, y, z, results;
cin>>x>>y>>z>>resutls;
result= (x*2)+(y*2)+(z*2);
cout<<results;
return 0;
 }
```

Let’s try C++

Install Visual Studio and familiarise yourself with its interface.