# LAB 1

## C++ BASICS & FLOW OF CONTROL

# Outline

- C++ basics and flow of control reviews
- Lab1 exercise

# C++ Identifiers

- An Identifier is a name of variables constant, …
- A C++ identifier
  - Consists of a sequence of letters, digits, and the underscore character ( _ )
  - Must start with either a letter or an underscore character // avoid doing so in general
  - Is case-sensitive
  - Can be of any length // NOT true in reality
- Keywords are special identifiers
  - E.g., if, for, char, …
  - Cannot be used for user-defined entities

# C++ Variables

- Variables
  - Its name is an identifier
  - is a memory location to store data
  - Must be declared before its use

```
int number;                // declaration & definition
double width, length;  // declaration & definition
```

  - Meaningful names!
  - Naming convention: starting with a lowercase letter
    - E.g., weight, total_weight, …

# Fundamental Data Types (1/2)

**Display 1.2 Simple Types**

| TYPE NAME | MEMORY USED | SIZE RANGE | PRECISION |
|---|---|---|---|
| `short` (also called `short int`) | 2 bytes | −32,768 to 32,767 | Not applicable |
| `int` | 4 bytes | −2,147,483,648 to 2,147,483,647 | Not applicable |
| `long` (also called `long int`) | 4 bytes | −2,147,483,648 to 2,147,483,647 | Not applicable |
| `float` | 4 bytes | approximately $10^{-38}$ to $10^{38}$ | 7 digits |
| `double` | 8 bytes | approximately $10^{-308}$ to $10^{308}$ | 15 digits |

# Fundamental Data Types (2/2)

| | | | |
|---|---|---|---|
| `long double` | 10 bytes | approximately $10^{-4932}$ to $10^{4932}$ | 19 digits |
| `char` | 1 byte | All ASCII characters (Can also be used as an integer type, although we do not recommend doing so.) | Not applicable |
| `bool` | 1 byte | `true`, `false` | Not applicable |

The values listed here are only sample values to give you a general idea of how the types differ. The values for any of these entries may be different on your system. *Precision* refers to the number of meaningful digits, including digits in front of the decimal point. The ranges for the types `float`, `double`, and `long double` are the ranges for positive numbers. Negative numbers have a similar range, but with a negative sign in front of each number.

# Constants

```
double money *= (1 + 0.05); //What is 0.05?
```

```
const double RATE = 0.05; //all uppercase letters
double money *= (1 + RATE);  // better readability
RATE = 0.1;  // compilation error
```

- Named constants or declared constants (e.g., RATE)
  - Better readability and maintainability
  - Change attempts result in compilation errors!
- Named constants MUST be initialized

```
const int myWeight;  // compilation error!
```

# Arithmetic Precision

- Examples:
  - 17 / 5  evaluates to 3 in C++!
    - Both operands are integers (Integer division)
  - 17.0 / 5 equals 3.4 in C++!
    - Highest-order operand is "double type“ (Double "precision" division)
  - int intVar1 =1, intVar2=2;  intVar1 / intVar2;
    - Result: 0! (Integer division)
- Calculations done "one-by-one“
  - 1 / 2 / 3.0 / 4  performs 3 separate divisions.
    - First ➔ 1 / 2  equals 0
    - Then ➔ 0 / 3.0  equals 0.0
    - Then ➔ 0.0 / 4  equals 0.0!
- So not necessarily sufficient to change just "one operand" in a large expression

# Type Casting

- Casting for Variables
  - C style
    - double dvar = (double) ivar;
  - C++ style
    - double dvar = static_cast<double>(ivar) ;
    - static_cast<type>(expression)
- Two kinds
  - implicit — also called “automatic“
    - done for you automatically
      17 / 5.5
      casting the 17 ➔ 17.0
  - explicit type conversion
    - programmer specifies conversion with static_cast operator
      int m;
      static_cast<double>(m) / 5.5

# Libraries

- C++ standard libraries
  - Input/output, math, strings, …
- #include <Library_Name>
  - directive to "add" contents of the specified library file to your program
  - called "preprocessor directive"
    - Executes before compilation, and simply “copies” library file into your program file

# Namespaces

- Namespaces defined:
  - collection of name definitions

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

  - includes entire standard library of name definitions

# Console Input/Output

- I/O objects cin for input, cout for output, cerr for error output
- Defined in the C++ library called <iostream>
- Must have these lines (called pre-processor directives) near start of file:

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

  - Tells C++ compiler to use appropriate library so we can use the I/O objects cin, cout, cerr

# Console Output

- Any data can be outputted to display screen
  - Variables
  - Constants
  - Literals
  - Expressions (which can include all of above)
- cout << numberOfGames << " games played.";
  - “value” of variable numberOfGames and literal string “games played.” are outputted
- Cascading: multiple values in one cout
- New lines in output
  - cout << "Hello World\n";
  - cout << "Hello World" << endl;

# Console Input

- cin >> num;
  - waits on-screen for keyboard entry
  - value entered at keyboard is "assigned" to num
- ">>" (extraction operator) points opposite
  - Think of it as "pointing toward where the data goes“
  - no literals allowed for cin
    - Must input to a variable
    - cin >> 23; // compilation error!

# Branch Mechanisms

- if-else statements
  - Choice of two mutually exclusive statements based on condition expression
  - Syntax:

```
if(<Boolean_expression>){
    <true_statement>
}else{
    <false_statement>
}
```

# Multiway if-else (1/2)

- Avoid "excessive" indenting
- Syntax :

**Multiway if–else Statement**

**SYNTAX**

```
if (Boolean_Expression_1)
    Statement_1
else if (Boolean_Expression_2)
    Statement_2
        .
        .
        .
else if (Boolean_Expression_n)
    Statement_n
else
    Statement_For_All_Other_Possibilities
```

# Multiway if-else (2/2)

- Example :

**EXAMPLE**

```
if ((temperature < -10) && (day == SUNDAY))
    cout << "Stay home.";
else if (temperature < -10) //and day != SUNDAY
    cout << "Stay home, but call work.";
else if (temperature <= 0) //and temperature >= -10
    cout << "Dress warm.";
else //temperature > 0
    cout << "Work hard and play hard.";
```

The Boolean expressions are checked in order until the first `true` Boolean expression is encountered, and then the corresponding statement is executed. If none of the Boolean expressions is *true*, then the *Statement_For_All_Other_Possibilities* is executed.

# Switch Statement (1/3)

- Controlling expression MUST return an integral value
  - OK: char, int, bool, enum
  - not OK: float, double, …
- Case labels must also be integral values
- break and default are optional
- Execution "falls thru" until break

```
example:

case 'A':
case 'a':
    cout << "Excellent: you got an A!\n";
    break;

case 'B':
case 'b':
    cout << "Good: you got a B!\n";
    break;
```

# Switch Statement (2/3)

- Syntax :

**switch Statement**

**SYNTAX**

```
switch (Controlling_Expression)
{
    case Constant_1:
        Statement_Sequence_1
        break;
    case Constant_2:
        Statement_Sequence_2
        break;
                .
                .
                .
    case Constant_n:
        Statement_Sequence_n
        break;
    default:
        Default_Statement_Sequence
}
```

*You need not place a **break** statement in each case. If you omit a **break**, that case continues until a **break** (or the end of the **switch** statement) is reached.*

# Switch Statement (3/3)

- Example :

**EXAMPLE**

```
int vehicleClass;
double toll;
cout << "Enter vehicle class: ";
cin >> vehicleClass;

switch (vehicleClass)
{
    case 1:
        cout << "Passenger car.";
        toll = 0.50;
        break;
    case 2:
        cout << "Bus.";
        toll = 1.50;
        break;
    case 3:
        cout << "Truck.";
        toll = 2.00;
        break;
    default:
        cout << "Unknown vehicle class!";
  }
```

*If you forget this* **break**, *then passenger cars will pay $1.50.*

# Loops

- 3 Types of loops in C++
  - while
  - do-while
    - always enters the loop body at least once
  - for
    - appropriate for “counting” loops

# while Loop Syntax

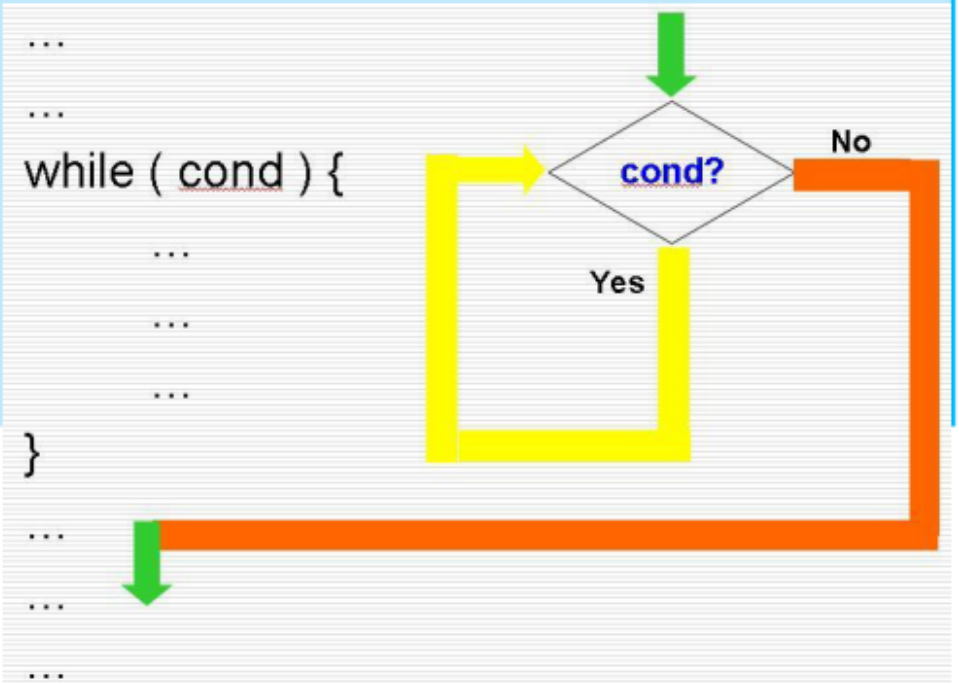
**Syntax for while and do-while Statements**

**A while STATEMENT WITH A SINGLE STATEMENT BODY**

```
while (Boolean_Expression)
    Statement
```

**A while STATEMENT WITH A MULTISTATEMENT BODY**

```
while (Boolean_Expression)
{
    Statement_1
    Statement_2
      .
      .
      .
    Statement_Last
}
```



# do-while Loop Syntax

**A do-while STATEMENT WITH A SINGLE-STATEMENT BODY**

```
do
    Statement
while (Boolean_Expression);
```

**A do-while STATEMENT WITH A MULTISTATEMENT BODY**

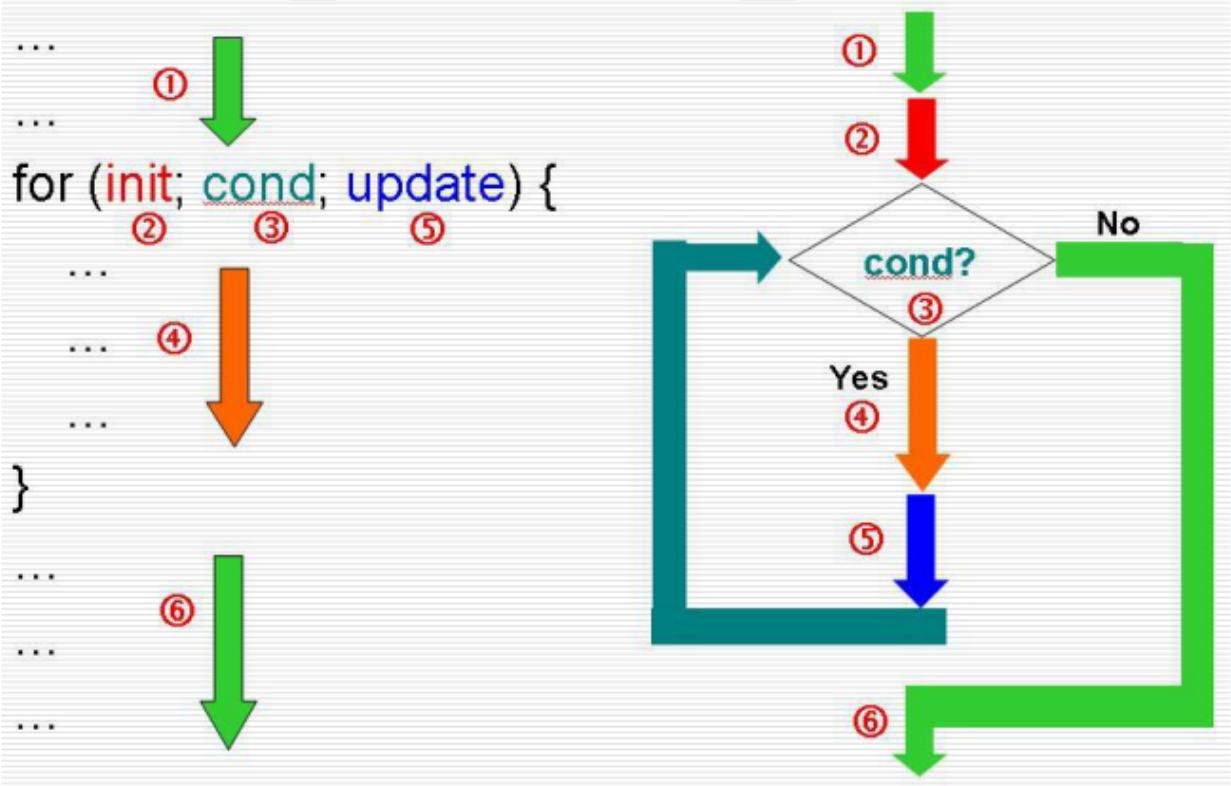
```
do
{
    Statement_1
    Statement_2
      .
      .
      .
    Statement_Last
} while (Boolean_Expression);
```

*Do not forget the final semicolon.*

# for Loop Syntax

**Syntax:**

```
for (Init_Action; Bool_Cond; Update_Action)
    Body_Statement
```



# Lab Exercise (1/2)

- Write a C++ program such that
  - it can collect people's age
  - it can compute the average age of these people
  - it can compute the number of adults (age≧18)
  - it will eliminate the negative values and show warning messages
- Use unsigned constant to store the legal age
- The variables to store the average age should be double type, number of people and number of adults should be unsigned type, so you are asked to use type casting to calculate the mean value of the ages

# Lab Exercise (2/2)

- Example :

```
Key in Y to insert more personal information or N to exit : N
Exit from inserting personal information.
The number of people is 0
The mean value of their age is 0
The number of adult is 0
請按任意鍵繼續 . . .
```

```
Key in Y to insert more personal information or N to exit : Y
  Key in the age of the person : 12
Key in Y to insert more personal information or N to exit : Y
  Key in the age of the person : -50
  You keyed in a negative number, please key in again.
Key in Y to insert more personal information or N to exit : Y
  Key in the age of the person : 0
Key in Y to insert more personal information or N to exit : Y
  Key in the age of the person : 23
Key in Y to insert more personal information or N to exit : Y
  Key in the age of the person : 40
Key in Y to insert more personal information or N to exit : N
Exit from inserting personal information.
The number of people is 4
The mean value of their age is 18.75
The number of adult is 2
請按任意鍵繼續 . . .
```