C++ AND OOP CONCEPTS

Dr. Yuzana Win
CEIT Department

CONTENTS

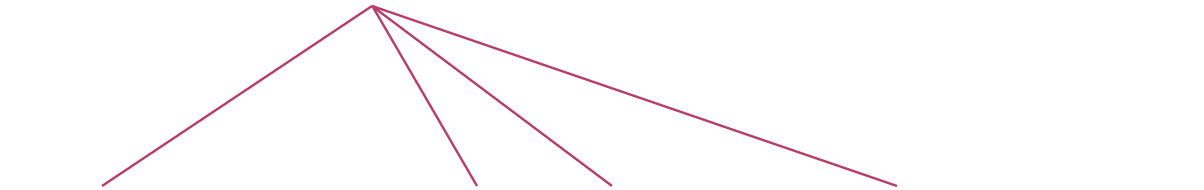
- Introduction to C++
- Difference between C and C++
- Types of Programming Techniques
- Object-oriented Programming in C++

INTRODUCTION TO PROGRAMMING LANGUAGE

- **What is a programming language?**
  - A programming language is a vocabulary and set of grammatical rules for instructing a computer to perform specific tasks.

PROGRAMMING LANGUAGE

- Machine Language
- Assembly Language
- High-level Language



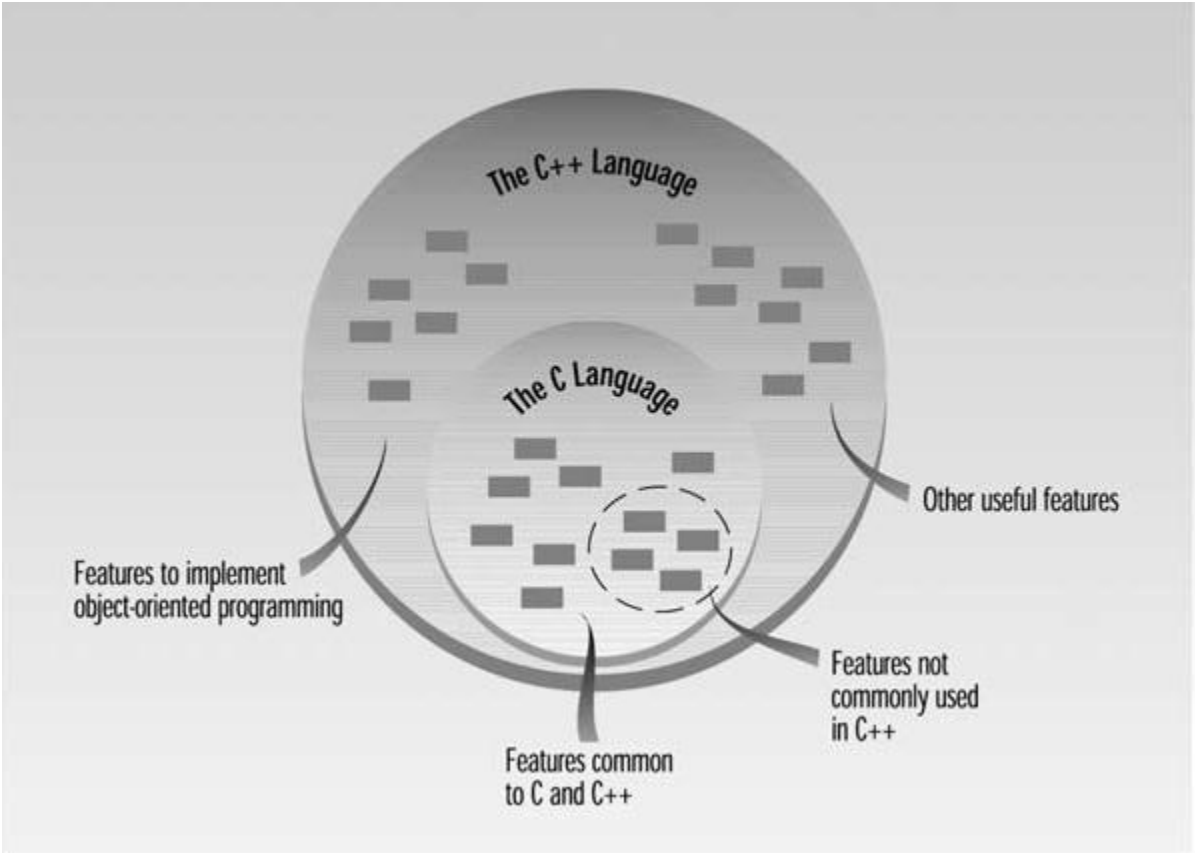
Difference between C and C++

DIFFERENCE BETWEEN C AND C++

- **C**
  - designed by **Dennies Ritchie** in the early 1970s at Bell Laboratories.
  - was the evolution of B and BCPL by incorporating type checking.
  - intended for use in writing compilers for other languages.
  - **Function or Procedure-Oriented**

- **C++**
  - devised by **Bjarne Stroustrup** in 1983 at Bell Laboratories.
  - is an extension of C by adding some enhancements to C language.
  - combines features of object oriented and the efficiency of C.
    - So, it is an extension of C language.
  - **Object-Oriented**

DIFFERENCE BETWEEN C AND C++



DIFFERENCE BETWEEN C AND C++

- **Features of C++**
  - C++ can declare variable **declarations anywhere**
  - C++ allow **function overloading**
  - New I/O library (e.g **iostream**)
  - **const -** Constant Data Types
  - **comments**
  - **inline** Functions
  - **reference** parameters
  - allocation and de-allocation of memory using **new** and **delete** operator

Type of Programming Techniques

TYPE OF PROGRAMMING TECHNIQUES

- Unstructured Programming
- Procedural Programming
- Object-Oriented Programming

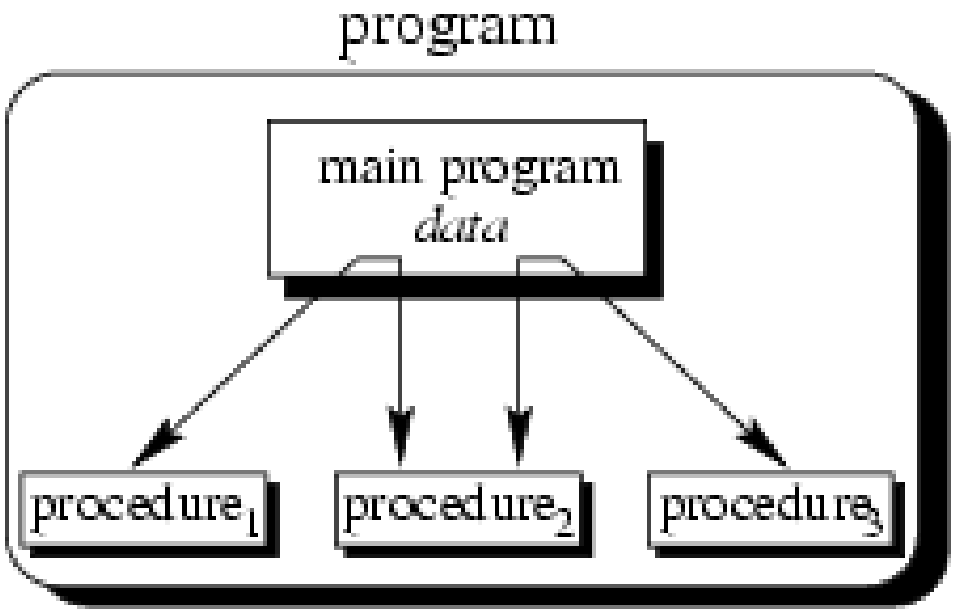
UNSTRUCTURED PROGRAMMING

- The historically earliest programming paradigm capable of creating Turing-complete algorithms.

- There are both high and low level programming languages that use non-structured programming.

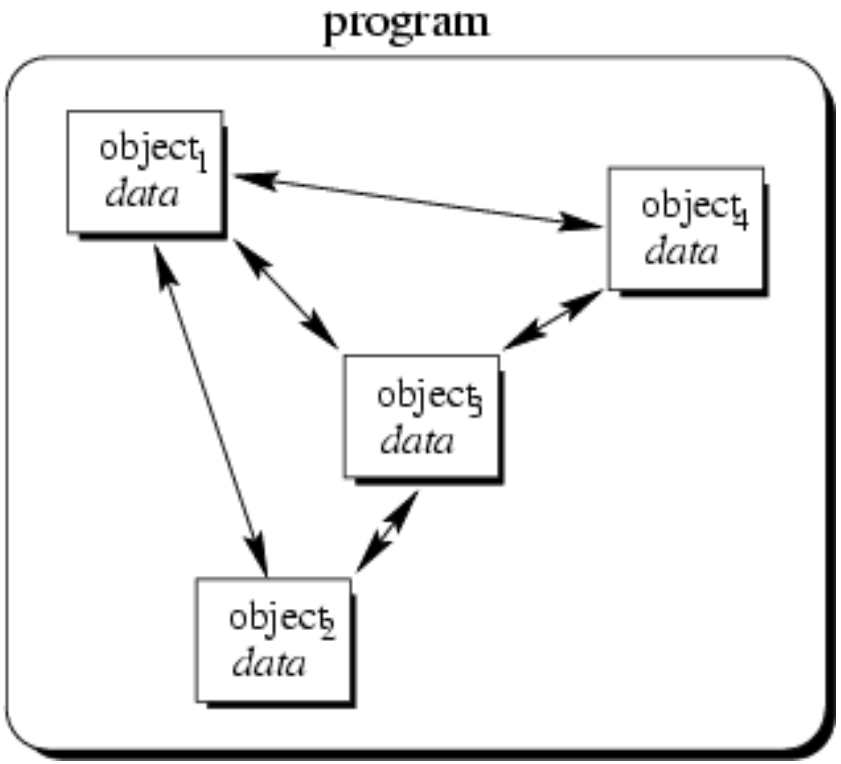
- It includes early versions of BASIC older dialects of COBOL, machine level code, early assembler systems (without procedural meta operators), assembler debuggers and some scripting languages such as MS-DOS batch file language.

PROCEDURAL PROGRAMMING



- Tell the computer to do something
- A program in a procedural language is a list of instructions
- The programmer creates the list of instructions, and the computer carries them out
- **Examples**: C, Pascal, Basic, Assembly language…

OBJECT-ORIENTED PROGRAMMING



- Objects of the program interact by sending messages to each other

OBJECT-ORIENTED PROGRAMMING

- **An object is an encapsulation of both functions and data.**
- **Objects are the abstractions of real world entities.**
  - Classes are data structures that define common properties or attributes
  - Objects are instances of a class
- **Objects have States.**
  - Each property or attribute has a value at a particular time
- **Objects have Operations.**
  - associated set of operations called methods describe how to carry out operations
- **Objects have Messages.**
  - request an object to carry out one of its operations by sending it a message
  - messages are the means by which we exchange data between objects

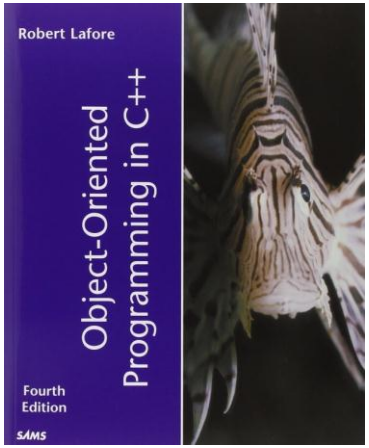
Object-Oriented Programming in C++

COURSE OUTLINE

- Week 1 Introduction
- Week 2 C++ Programming Basics
- Week 3 Loops and Decisions
- Week 4 Structures
- Week 5 Functions
- Week 6 Classes
- Week 7 Arrays and Strings
- Week 8 Operator Overloading
- Week 9 Single Inheritance
- Week 10 Multiple Inheritance
- Week 11 Pointer
- Week 12 Streams and Files

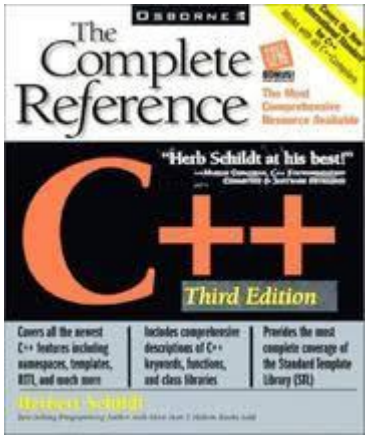
TEXT BOOK & REFERENCES

[1] Object-oriented Programming in C++ Fourth Edition, 2002



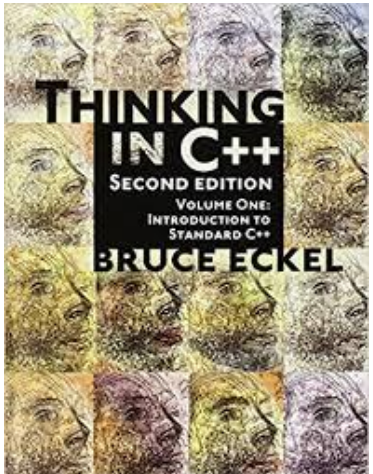
---

[2] The Complete Reference C++, Herbert Schildt, 3rd Edition



---

[3] Thinking in C++ Volume 1 and 2, Bruce Eckel, 2nd Edition



TEACHING AID

- SOLOLEARN (Everyone Can Code)

https://www.sololearn.com/Course/CPlusPlus/

- Learn and Understand C++

https://www.udemy.com/learn-c-plus-plus-from-beginner-to-advanced/

LEARNING OUTCOMES

- To understand the concept of OOP
- To analyze and create the classes, objects and members of a class
- To develop C++ classes for simple applications

Library Management System

Student Database Management System

TEACHING SCHEDULE

	Date	Topic
1	17.9.2018	• Introduction
2	24.9.2018	• C++ Programming Basics
3	1.9.2018	• Loops and Decisions
4	8.9.2018	• Structures
5	15.9.2018	• Functions
6	22.9.2018	• Objects and Classes

TEACHING SCHEDULE

	Date	Topic
7	29.9.2018	• Arrays and Strings
8	5.10.2018	• Operator Overloading
9	12.10.2018 ~ 19.10.2018	• Inheritance •- Single Inheritance • - Multi Inheritance
10	26.10.2018	• Pointer
11	3.10.2018	• Streams and Files

EXAM AND GRADING SYSTEM

- Grading:
  - Final exam (80% out of 100%)
  - Lab Report (10%)
  - Project (10%)

- Assignment in
  - C-Free 5, Bloodshed Dev-C++

EXAM AND GRADING SYSTEM

	Task	Marks
1	Assignment I	10
2	Assignment II	10
3	Assignment III	10
4	Assignment IV	10
5	Quiz I	10

	Task	Marks
6	Assignment V	10
7	Assignment VI	10
8	Assignment VII	10
9	Assignment VIII	10
10	Quiz II	10

WEEK 1
INTRODUCTION

INTRODUCTION

- What is C++?

  C++ is a powerful, efficient and fast language

  - is a better C,
  - has a rich function library
  - supports object-oriented programming
    - classes
    - inheritance
    - polymorphism
    - data abstraction and encapsulation

REAL-WORLD APPLICATIONS OF C++

- Games
- Graphic User Interface (GUI) based applications
- Web Browsers
- Advance Computations and Graphics
- Database Software
- Operating Systems
- Enterprise Software
- Medical and Engineering Applications

EXAMPLE C AND C++ PROGRAM

```c
#include <stdio.h>
#include <stdlib.h>

main()
{
    printf("Hello World!\n");
    system("pause");
}
```

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

main()
{
    cout << “Hello World!” << “\n
    system(“pause”);
 }
```

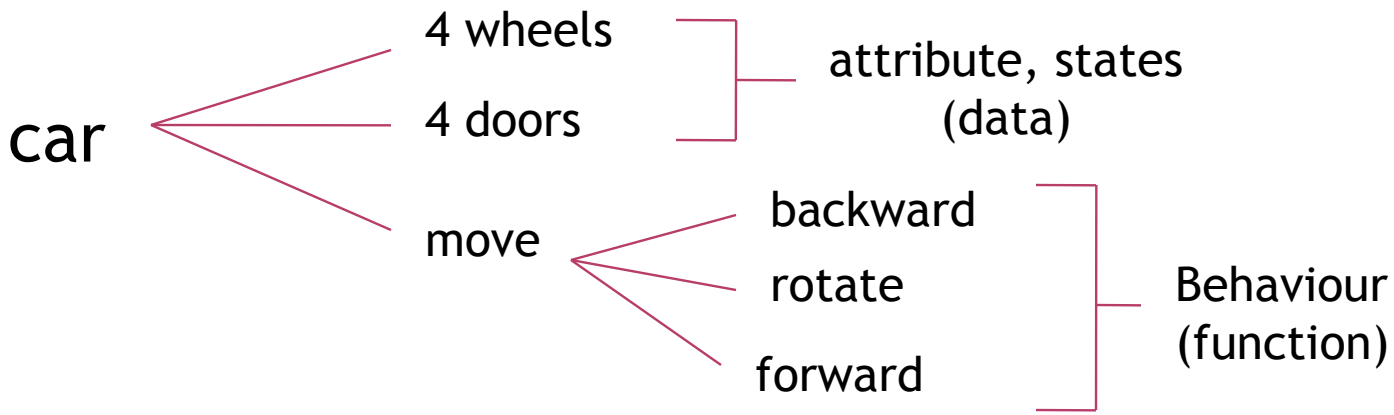
FEATURES OF OOP

- Object
- Class
- Message and Method
- Inheritance
- Encapsulation
- Polymorphism

OBJECT

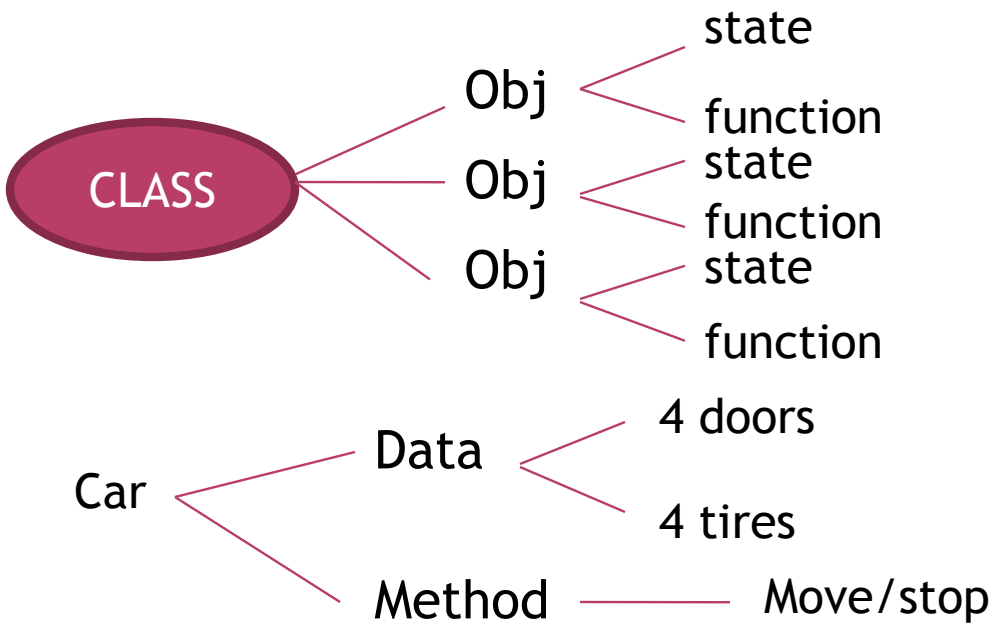
- Object are everything in the world. An object is said to an instance of a class

Eg.



CLASS

- A description of one or more objects with a uniform set of attributes and services including a description of how to create new objects in the class
- So, a class is to be a kind of template



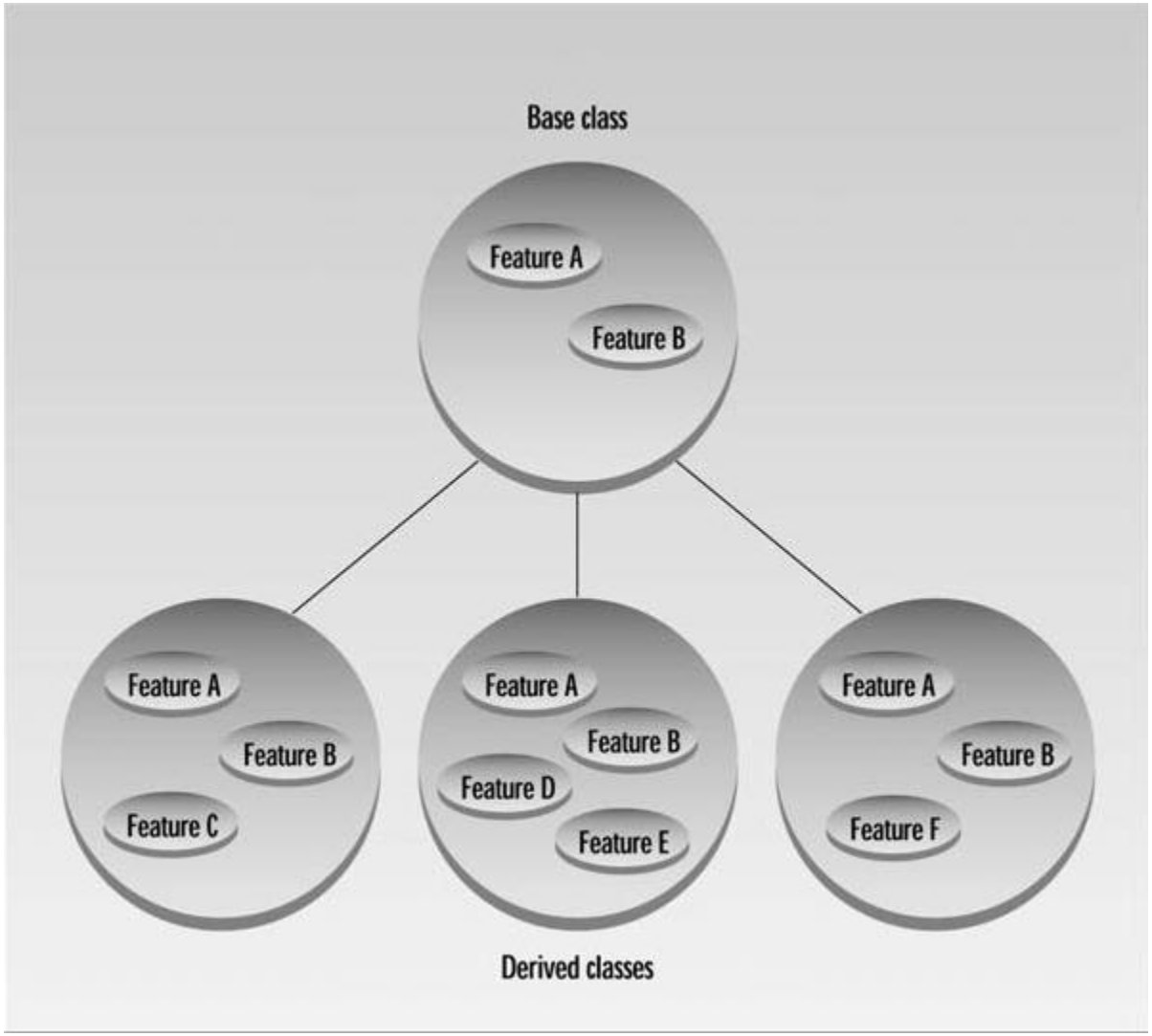
MESSAGE AND METHOD

- The message is an only means available for giving an instruction to an object
- A behaviour specific to an object is called a method

INHERITANCE

- Inheritance is the process of *creating new classes*, called *derived classes*, from existing or base classes
- The original class is called *the base class*; other classes can be defined that *share* its characteristics, called *derived classes*
- The derived classes *inherit* some characteristics from *their base class*, but can add refinements of its own
- It permits code *reusability*

INHERITANCE



ENCAPSULATION

- **Data encapsulation** is a mechanism of bundling the data, and the functions that use them
- **Data abstraction** is a mechanism of exposing only the interfaces and hiding the implementation details from the user
- C++ supports the properties of encapsulation and data hiding through the creation of user-defined types, called classes
- A class can contain *private*, *protected* and *public* members

ENCAPSULATION EXAMPLE

```
class Rectangle
{  private:
        int x, y, width, length;
   public:
       void set(int a, int b, int w, int l)
      { x=a;
        y=b;
        width=w;
        length=l;
      }
};
```

ACCESS CONTROL SPECIFIERS

- **Public**
  - Can be accessible from anywhere in the program
- **Private (default)**
  - Can be accessible only from member functions of its class and friends
- **Protected**
  - Acts as public for objects of its own class and derived classes
  - Acts as private to rest of the program
- ***Public functions*** – Class Interface
- ***Private functions*** – helper functions (can be accessed by class objects and friends)

POLYMORPHISM

- *polymorphism* means that some code or operations or objects behave differently in different contexts.

For example, the + (plus) operator in C++:

```
4 + 5          <--    integer addition
3.14 + 2.0     <--    floating point addition
s1 + "bar"     <--    string concatenation!
```

- In C++, that type of polymorphism is called *overloading*.

UNIFIED MODELING LANGUAGE (UML)

- UML is a standardized way to visualize a program's structure and operation using diagrams
- UML is useful in developing software systems, testing and maintenance