

# Introduction

Md. Kamrul Islam

Lecturer, Dept. of CSE, RMU

# How comes OOP?

- ❑ Programming Language
- ❑ Procedural Oriented Programming Language
- ❑ Object Oriented Programming Language

# Procedure Oriented Programming

- Emphasis in on doing things (algorithms)
- Large programs are divided into smaller programs known as function.
- Most of the functions share global data.
- Data move openly around the system from function to function.
- Function transform data from one form to another.

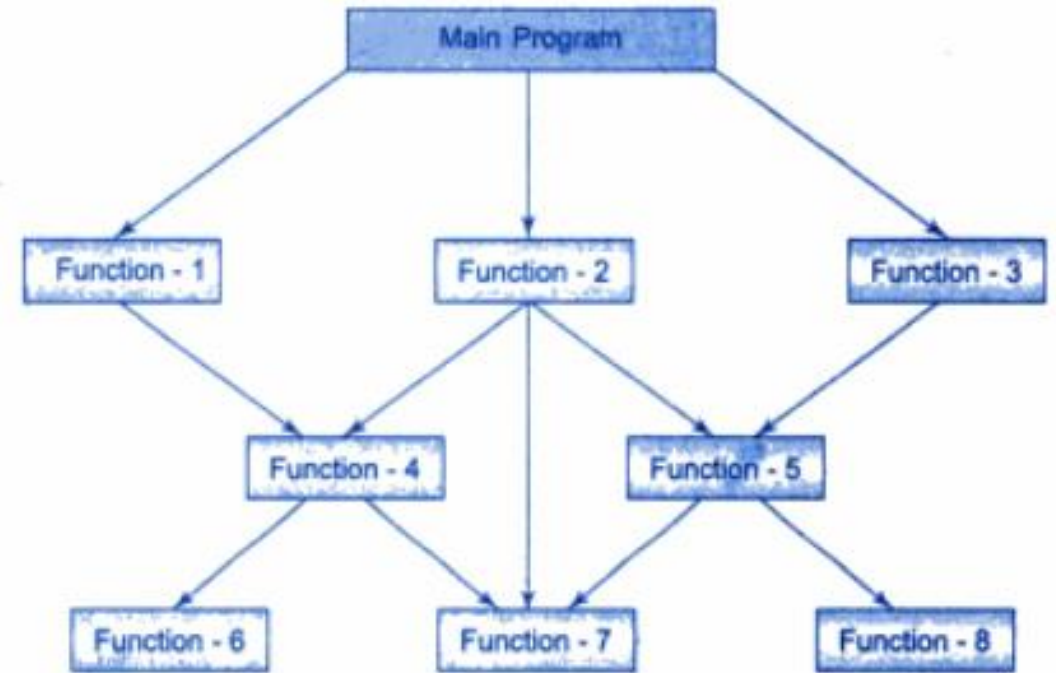


Fig. 1.4 ⇒ Typical structure of procedure-oriented programs

# OOP

- Emphasis is on data rather than procedure
- Programs are divided into what are known as objects.
- Data structures are designed such that they characterize the objects.
- Functions that operate on the data of an object are tied together in the data structure.
- Data is hidden and cannot be accessed by external functions.
- Objects may communicate with each other through functions.
- New data and functions can be easily added whenever necessary.

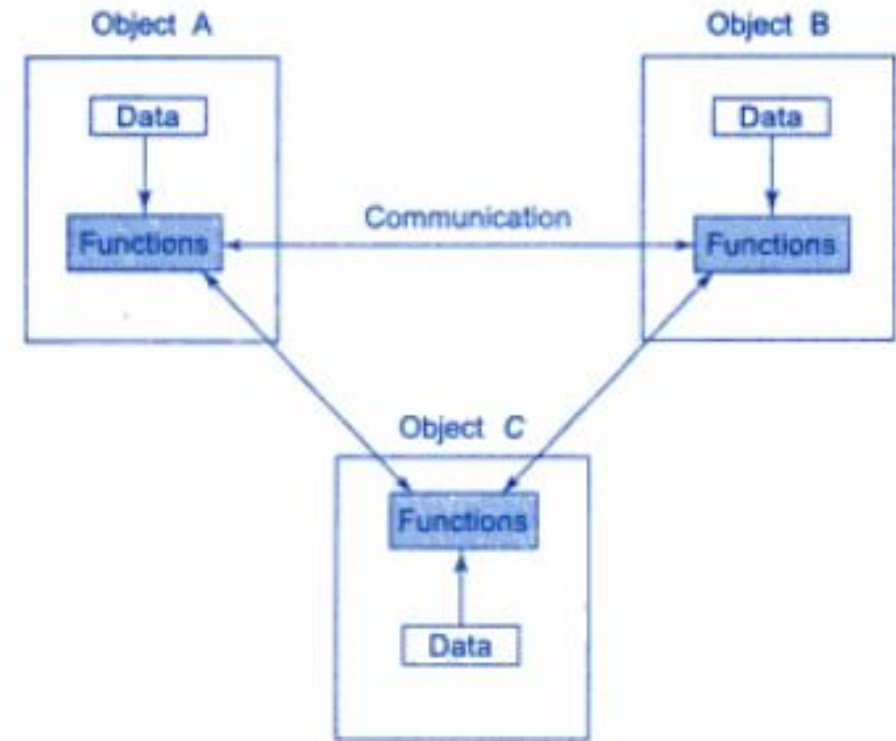


Fig. 1.6 ⇔ Organization of data and functions in OOP

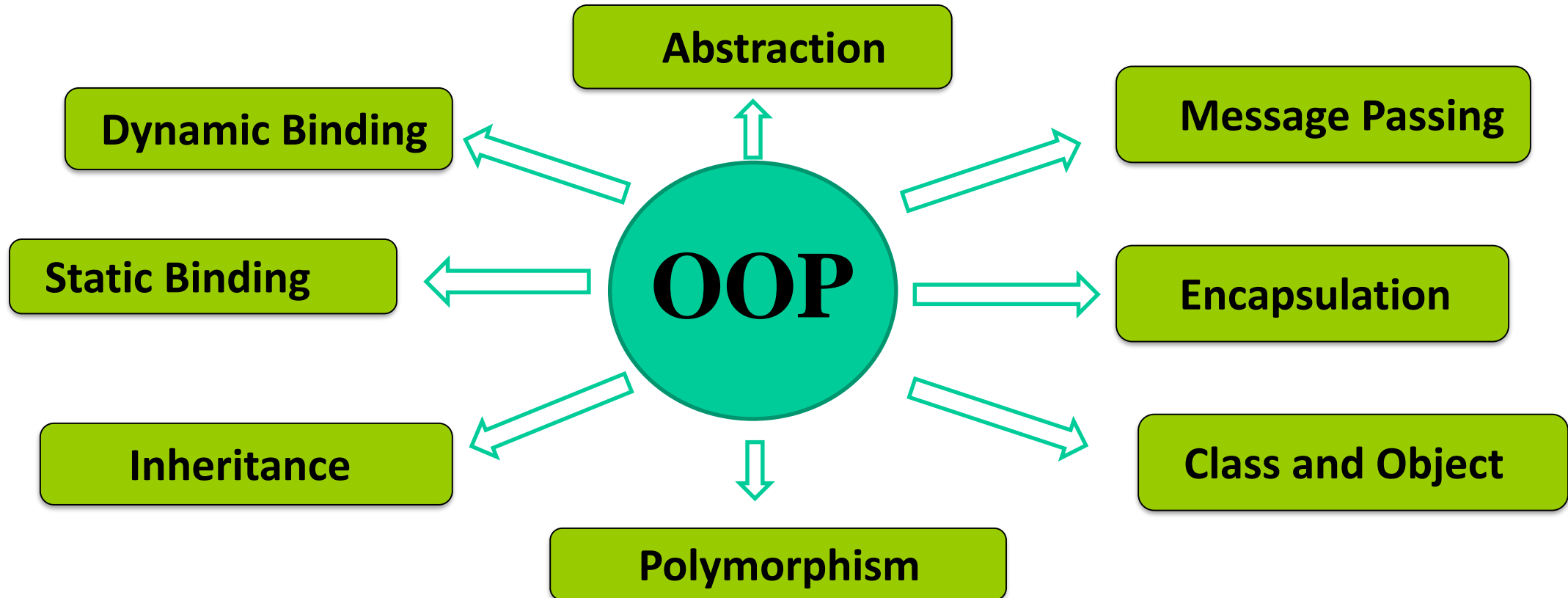
# POP vs OOP

- Emphasis is on doing things (algorithms)
- Large programs are divided into smaller programs known as function.
- Most of the functions share global data.
- Data move openly around the system from function to function.
- Function transform data from one form to another.
- Employs top-down approach in program design.

- Emphasis is on data rather than procedure
- Programs are divided into what are known as objects.
- Data structures are designed such that they characterize the objects.
- Function that operate on the data of an object are tied together in the data structure.
- Data is hidden and cannot be accessed by external functions.
- Objects may communicate with each other through functions.
- New data and functions can be easily added whenever necessary.
- Follows bottom-up approach in program design.

# OOP

Object oriented programming encourages programmers to decompose a problem into its constituent parts.



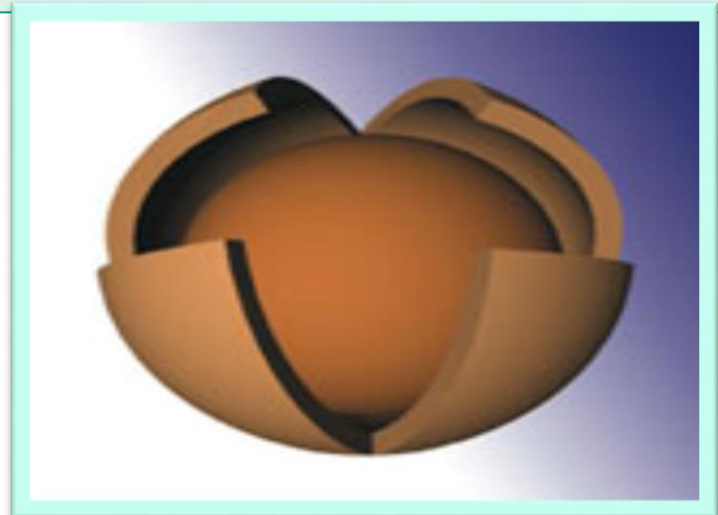
# OOP: Encapsulation

## Definition

Encapsulation is the mechanism that binds together code and the data it manipulates, and keeps both safe from outside interference and misuse.

In object oriented language, code and data can be combined in such way that a self-contained “black box” is created.

- Private
- Public



# OOP: Object

## Definition

Objects are basic run time entities in an object oriented system. They may represent a person, a place, a bank account, a table of data or any item that the program has to handle.

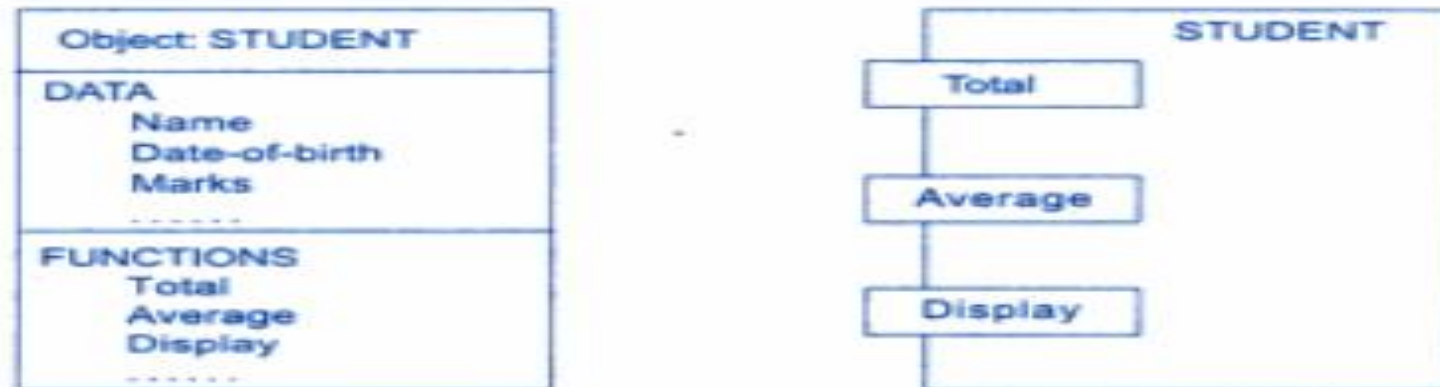


Fig. 1.7 ⇒ Two ways of representing an object



# OOP: Class

## Definition

A class is used to specify the form of an object and it combines data representation and methods for manipulating that data into one neat package.

In fact, objects are variables of the type class.

Fruit mango;

# OOP: Inheritance

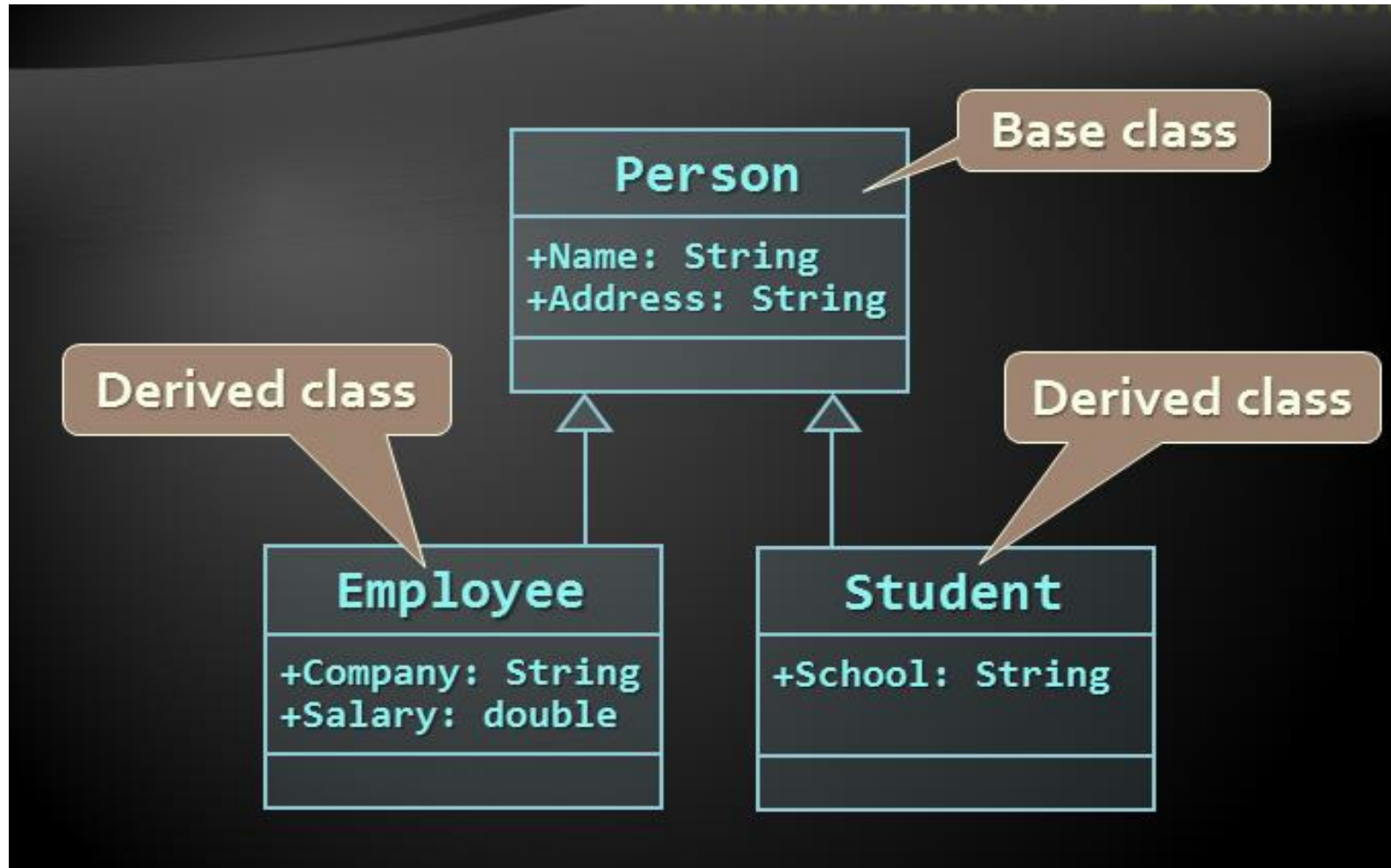
## Definition

Inheritance is the process by which one object can acquire the properties of another.

Hierarchical classification

Without the use of ordered classifications, each object has to define all characteristics related to it explicitly.

# OOP: Inheritance



# OOP: Polymorphism

## Definition

Polymorphism is the quality that allows one name to be used for two or more related but technically different purposes.

polymorphism  many forms

One interface, multiple methods

- Function overloading
- Operator overloading

# OOP: Polymorphism



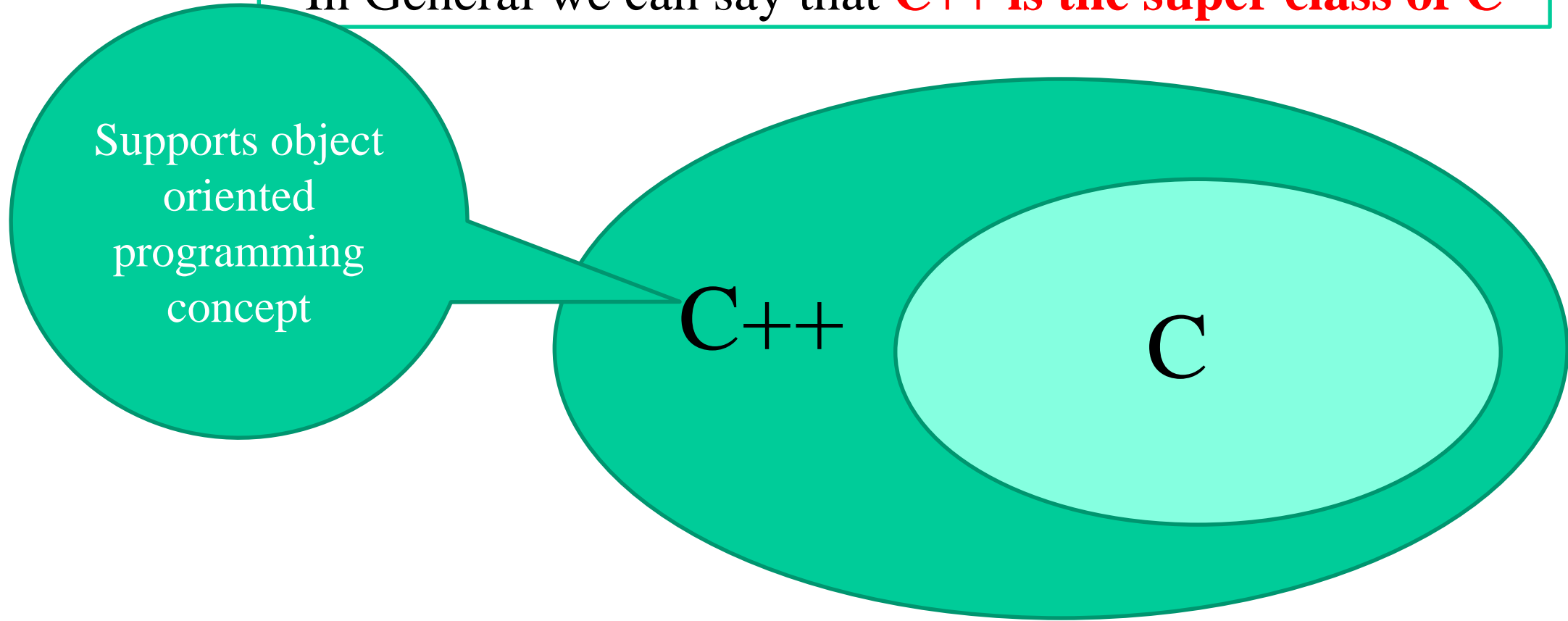
# C++?

In General we can say that **C++ is the super class of C**

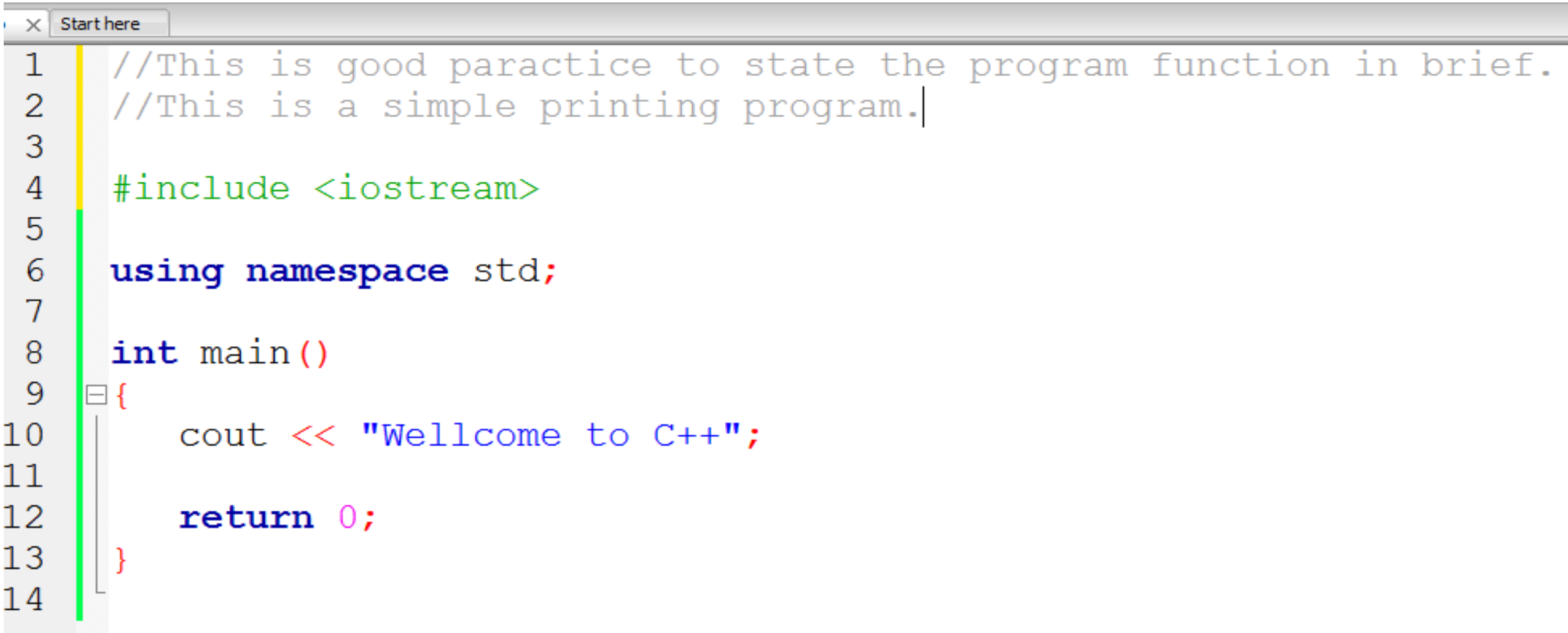
Supports object  
oriented  
programming  
concept

C++

C



# Basic Structure of C++ Program

A screenshot of a code editor window titled 'Start here'. The editor displays a C++ program with line numbers 1 through 14 on the left margin. The code is as follows:

```
1 //This is good paractice to state the program function in brief.  
2 //This is a simple printing program.  
3  
4 #include <iostream>  
5  
6 using namespace std;  
7  
8 int main()  
9 {  
10     cout << "Wellcome to C++";  
11  
12     return 0;  
13 }  
14
```

The code is color-coded: comments are grey, preprocessor directives are green, namespace declarations are blue, and function declarations and control flow keywords are red. A vertical green line is positioned at the start of line 9, and a yellow line is at the start of line 4. A small icon is visible in the left margin next to line 9.

# #include <iostream>

- a preprocessor directive
- Lines that begin with **#** are processed by the preprocessor *before the program is compiled.*
- include in the program the contents of the **input/output stream header <iostream>.**

- We also can use C header files!!



# #include <iostream>

- We also can use C header files!!

- <math.h>
- <string.h>

- <cmath>
- <cstring>

# using namespace std;

- The **namespace** is simply a declarative region.
  - The **namespace** defines scope of an identifier.
  - Purpose is to localize the names of identifiers to avoid name collisions.
- 
- **std** is the namespace where C++ standard class libraries are defined.

# cout <<

- **cout** represents the standard output stream in C++
- The << operator is referred to as the stream insertion operator.
- When this program executes, the value to the operator's right, the right operand, is inserted in the output stream.

# return 0;

- Use to **EXIT** function

- Returns a value 0 to the calling procedure

# C++ Console I/O

In C++ I/O is performed using I/O operators instead of I/O functions.

Output Operator is <<

```
cout<<expression;
```

Input Operator is >>

```
Cin>>variable;
```

# Example:1

```
#include<iostream>
using namespace std;
int main(){
int a,b;
a=10;
b=20;
cout<<"Value of a and b";
cout<< a;
cout<<" ";
cout<<b;
return 0;
}
```

# Example:2

```
#include<iostream>
using namespace std;
int main(){
int a,b;
cout<<"Enter value of a and b";
cin>>a>>b;
cout<< a<<endl;
cout<<b;
return 0;
}
```

# Comments in C++

Program comments are explanatory statements that you can include in the C++ code that you write and helps anyone reading it's source code.

C++ supports single-line and multi-line comments

Single Line Comment can be written using //

Multi-line comments are written using /\*.....\*/