Written By -

Kamal Kishor

C++ Programming

What is C++?

C++ is a statically typed, compiled, general-purpose, free-form programming language that supports Procedural, Object-Oriented, and Generic programming.

Regarded as a middle-level language as it comprises a combination of both high-level & low-level language features.

→ Developed by Bjarne Stroustrup starting in 1979 at Bell Labs.

C++ is a superset of C; virtually any C program is a legal C++ program.

Object-Oriented Pillars

- Encapsulation
- Data Hiding
- Inheritance
- Polymorphism

Standard Libraries: [Core Language, Standard Library, STL (Standard Template Library)]

ANSI Standard: Portable code (Microsoft compiler works on

Usage of C++

- Window Application
- Client - Server Application
- Device Drivers
- Embedded Firmware

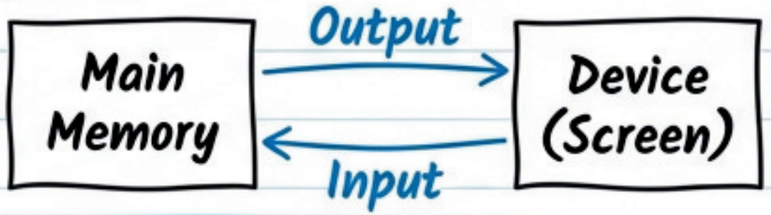
```cpp
#include <iostream>
using namespace std;
int main() {
    cout << "Hello C++ Programming";
    return 0;
}
```

Turbo C++ Download & Installation steps:

1. Download Turbo C++
2. Create turboc dictionary inside c drive & extract tc3.zip inside c:\turboc
3. Double click on install.exe file
4. Click on the tc application file located inside C:\TC\BIN

C	C++
C follows procedural style programming.	It follows both procedural & object oriented programming.
Data is less secured in C.	In C++ you can use modifiers for class members to make it inaccessible.
It follows top-down approach.	It follows bottom-up approach.
C doesn't support reference variable.	C++ supports reference variables.
scanf() & printf() mainly used for input/output.	cin & cout are used to perform input/output operations.
C does not support Inheritance.	C++ supports the term inheritance.

C++ Basic Input / Output



C++ I/O operation is using the stream concept.
Stream is the sequence of bytes or flow of data.

Header File	Function & Description
<iostream>	Used to define cout, cin & cerr objects.
<iomanip>	Used to declare services for formatted I/O such as setprecision.
<fstream>	Used to declare services for user-controlled file processing.

Standard Output Stream (cout)

The cout is predefined object of ostream class. Connected with standard output device (screen).

```
cout << "Value of ary is: " << ary;
```

Uses insertion operator (<<)

Standard Input Stream (cin)

The cin is predefined object of istream class. Connected with keyboard.

```
cin >> age;
```

Uses extraction operator (>>)

Standard end line (endl)

Object of ostream class. Inserts new line & flushes stream.

```
cout << "End of line" << endl;
```

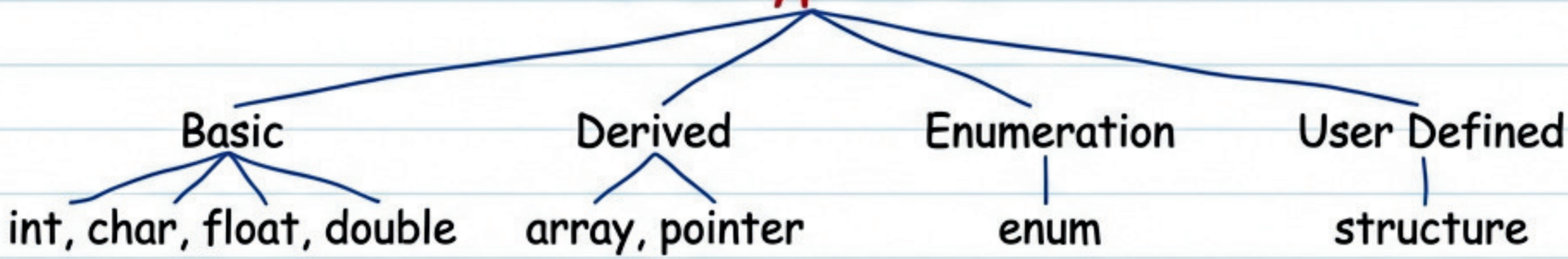
C++ Variables

A Variable is name of memory location used to store data. Its value can be changed.

```
int x;
float y;
char z;
```

Rules: Alphabets, digits, underscore. No white spaces.

Data Types in C++



Memory Size & Range

Data Types	Memory Size	Range
char	1 byte	-128 to 127
unsigned char	1 byte	0 to 255
short	2 byte	-32,768 to 32,767
int	2 byte	-32,768 to 32,767
unsigned int	2 byte	0 to 65,535
float	4 byte	
double	8 byte	

C++ Keywords

A keyword is a reserved word. You cannot use it as a variable name.

auto	break	case	char	const	continue	default	do
double	else	enum	extern	float	for	goto	if
int	long	register	return	short	signed	sizeof	static
struct	switch	typedef	union	unsigned	void	volatile	while

C++ Operators

Arithmetic: +, -, *, /, %
Relational: <, <=, >, >=, ==
Logical: &&, ||, !
Bitwise: &, |, <<, >>, ^
Assignment: =, +=, -=, /=, %=
Unary: ++, --
Conditional: ? :

Identifiers	Keywords
Name defined by programmer	Reserved words known by compiler.
Identify name of variable	Specify type of entity.
Letters, digits, underscore	Letters only.
Lowercase and uppercase	Lowercase only.

Logic and Flowcharts

(Theory)

Expressions

- Constant, Integral, Float, Pointer, Relational, Logical, Bitwise

Control Statements

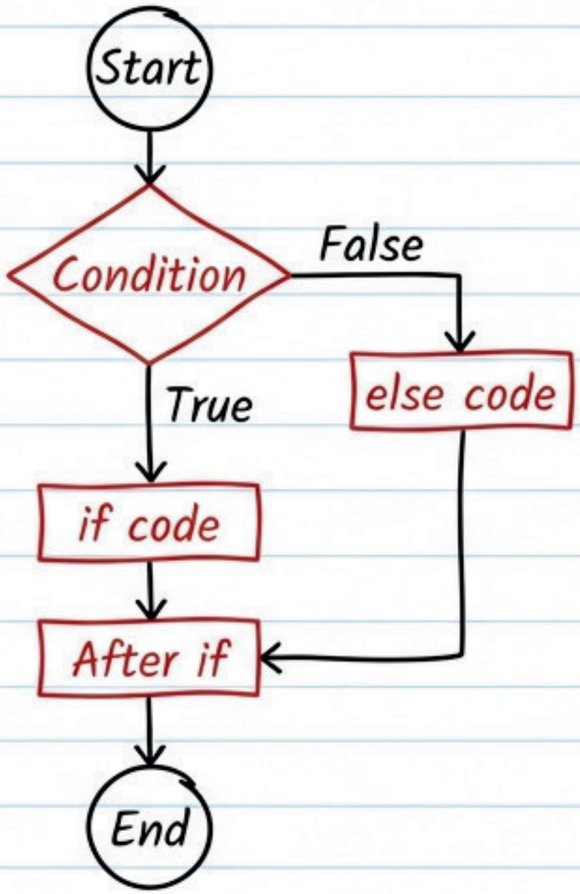
- if statement
- if-else statement
- nested-if statement
- if-else-if ladder

```
int a;
int A;
```

Case sensitive.

(Visual Logic)

If-Else Flowchart




```
#include <iostream>
using namespace std;
int main() {
   int num;
   cout << "Enter a number: ";
   cin >> num;
   if(num % 2 == 0) {
      cout << "It is even number";
   } else {
      cout << "It is odd number";
   }
   return 0;
}
```

Even/Odd logic example.

OOPs in C++

Main aim is to bind together the data and functions so that no other part of code can access this data.

Dynamic Binding: Code to be executed is decided at run time.

Class Anatomy

Constructors :

A constructor is a member function of a class which initializes objects. Automatically called when the object is created. It has the same name as the class itself. Constructor doesn't have a return type.

1. Default constructor (NO parameter passed)
2. Parametrized Constructor
3. Copy Constructor

Destructor in C++ :

Invoked when object destroyed. Derived class destructor will be invoked first, then the base class destructor will be invoked.

Access Modifier :

Public :
Can be accessed by any class.

Private :
Accessed only by function in a class. Inaccessible outside.

Protected :
Inaccessible outside class but accessible by Subclass.

Note :- If we do not specify any access modifier inside the class then by default the access modifier for the member will be private.

Friend Class :

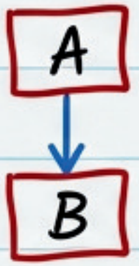
A Friend class can access private and protected members of other class in which it is declared as friend.

Ex:- friend class B;

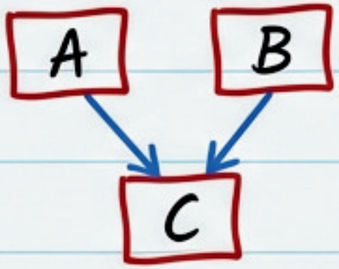
Inheritance

Class Subclass : accessmode . baseclass { }

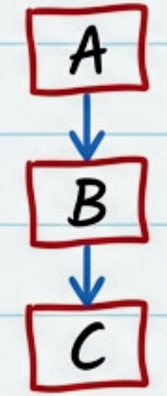
1. Single inheritance :



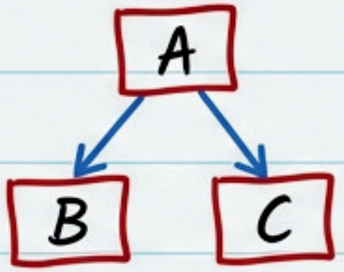
2. Multiple inheritance :



3. Multilevel :



4. Hierarchical inheritance :



5. Hybrid Combination of one or more type.

Polymorphism → Compile time Poly → Operator Overloading.
Compile time Poly → Function Overloading.
Polymorphism → Runtime

Runtime Poly & Exceptions

Runtime Poly: Function overriding occurs when derived class has definition of base class member.

Exception Handling:

try (code that throws), catch (code executed on error), throw (throws exception).

Comparisons

Structure vs Class:

Structure is not secure (cannot hide data). Class is secure (can hide data/functions).

C vs C++ (Advanced):

- C is function driven, C++ is object driven.
- C has 32 keywords, C++ has 52.
- C supports procedural, C++ supports hybrid.

Static Members in C++

In Function:

Allocated for lifetime of program. Default 0. Allocated once.

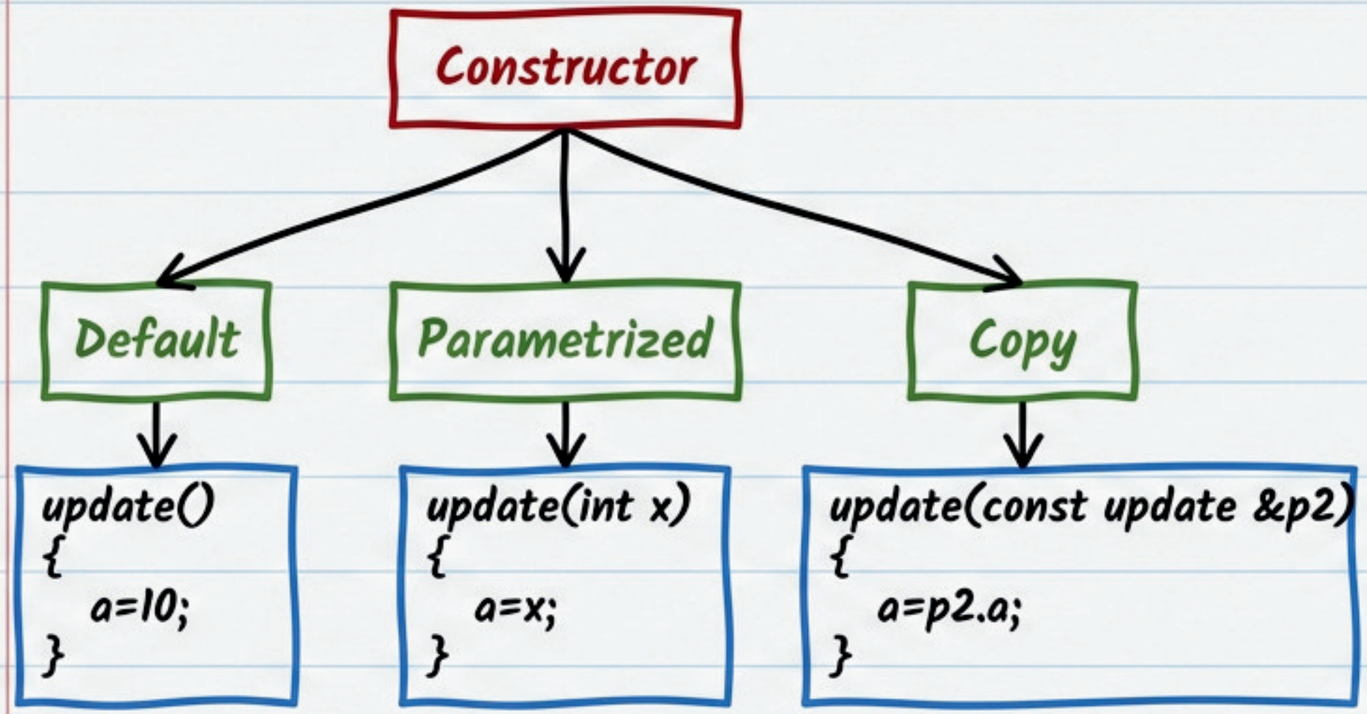
In Class:

One copy for whole class. Shared.

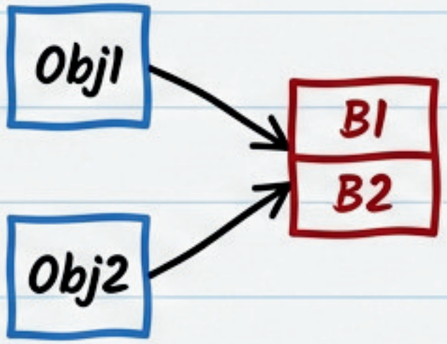
```
class Account {
    static float roi;
float Account :: roi = 3.5f; // Initialized outside class
```

Required outside class

Constructors & Copying:

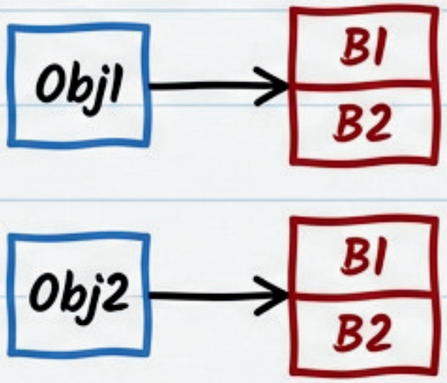


Shallow Copy:



Points to same memory location. Default copy constructor.

Deep Copy:



Points to new memory location. User defined constructor.

Q&A:

Can we make copy constructor private?

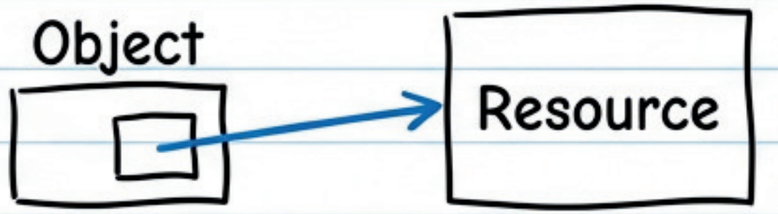
Yes.

Reference Note: Why argument to copy constructor must be passed as a reference?

To avoid non-terminating recursion.

Destructor

- No argument, no return type.
- Cannot be static.
- Last function invoked.



Destructor frees space allocated for resource before object deletion.

Operator Overloading

Give special meaning to operator.

```
Class Complex {
   Complex operator + (Complex &c1) { Complex res; res.a = c1.a + c2.a;
   return res; }
} int main() { C = C1 + C2; }
```

Friend Class Example

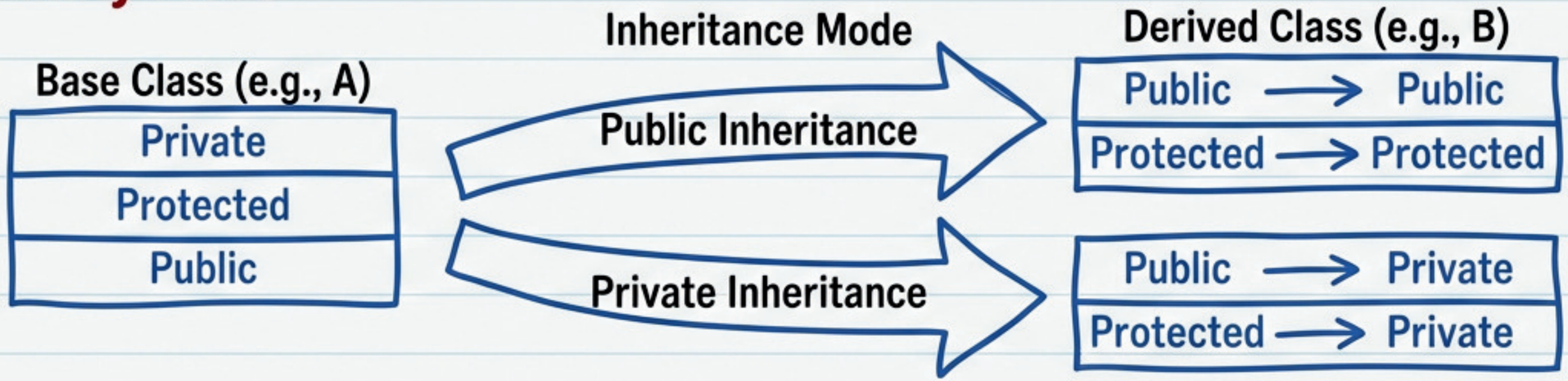
```
Class Box {
   private: double width;
   public: friend void printwidth(Box box);
};
void printwidth(Box box) { cout << box.width; // Accessing private data }
```

Inheritance Mechanics: Detailed Mechanics of Inheritance

Inheritance Syntax

```
Class Sports_Car : public Car { };
```

Visibility Mode



Is-A Relationship is always implemented as public inheritance.

Constructor & Destructor Order

Object Creation → Child Const → Parent Const → Complete Parent → Complete Child

Object Destruction → Child Desc → Parent Desc

In destructor, first child destructor executes, then parent.

This Pointer

Every object has access to its own address through "this" pointer.

```
void set(int l, int b, int h) {
    this->l = l;
    this->b = b;
    this->h = h;
}
```

Method Overriding

Redefinition of base class function in derived class with same name and parameters.

```
Class Car
 void change_gear(int gear) { gear++; }
```

```
Class SportsCar
 void change_gear(int gear) { if(gear>5) gear++; }
```

Note: Runtime Polymorphism.

Advanced Runtime & Templates

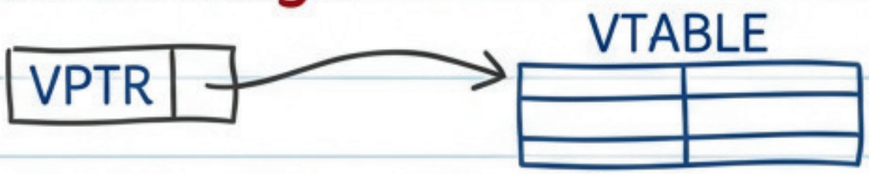
Virtual Function

Member function in base class with 'virtual' keyword. Enables Late Binding (Runtime).

Early Binding (Compile time)	vs	Late Binding (Runtime)

```
Base *bptr;
bptr = &derived;
bptr->print();    (Calls derived print if virtual)
```

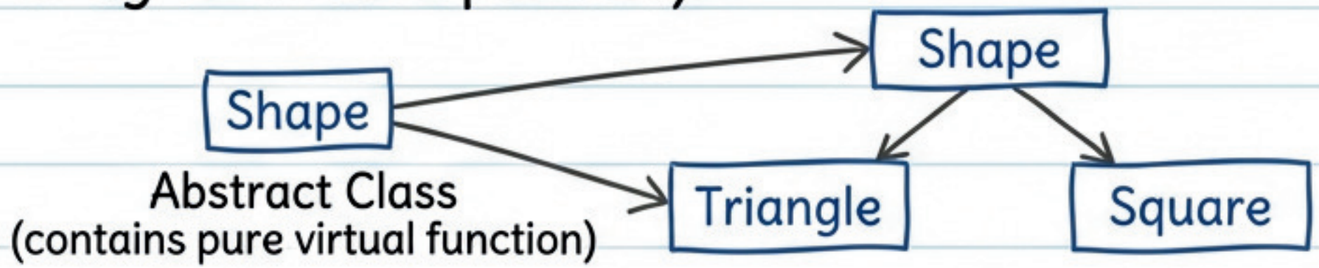
VTABLE Logic



VPTR points to VTABLE (static array of function pointers).

Pure Virtual & Abstract

```
virtual void fun() = 0;
```



Templates

Generic programming.

```
template <class T>
T add(T a, T b) {
    return a+b;
}
```