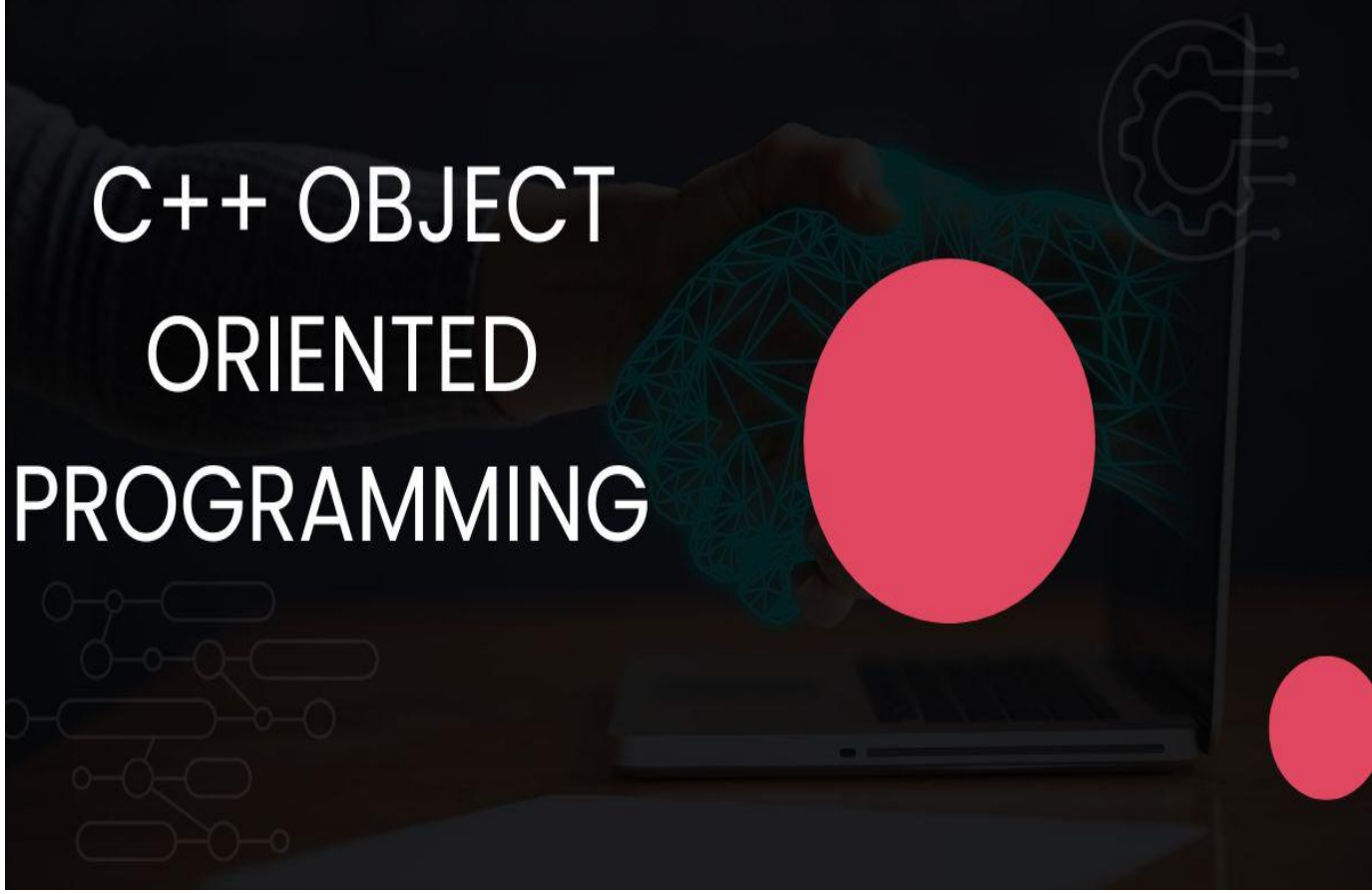


OBJECT ORIENTED PROGRAMMING

- C++ is the most used and most popular programming language developed by Bjarne Stroustrup. C++ is a high-level and object-oriented programming language

- A high-level programming language is a programming language designed to be easily understood and used by humans. It is characterized by its abstraction from the details of the computer hardware and its focus on expressing algorithms and logic in a more natural and intuitive way.

- Object-oriented programming (OOP) is a programming paradigm that organizes software design around objects and data, rather than functions and logic. In an object-oriented programming language, such as Java, Python, or C++, everything is treated as an object, which can contain data (attributes or properties) and code (methods or functions) that operate on the data.

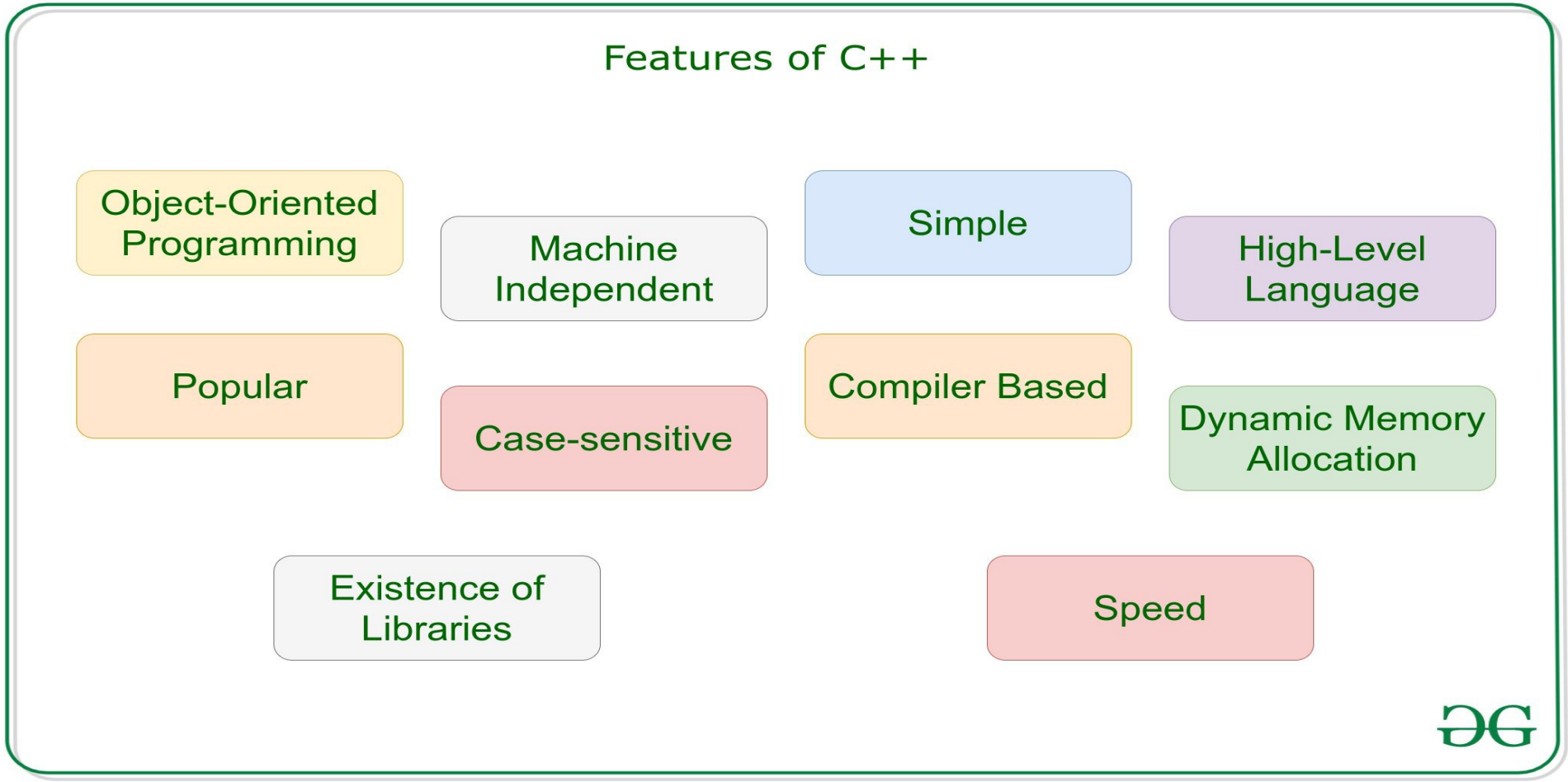
Procedural Oriented Programming	Object-Oriented Programming
In procedural programming, the program is divided into small parts called **functions**.	In object-oriented programming, the program is divided into small parts called **objects**.
Procedural programming follows a **top-down approach**.	Object-oriented programming follows a **bottom-up approach**.
There is no access specifier in procedural programming.	Object-oriented programming has access specifiers like private, public, protected, etc.
Adding new data and functions is not easy.	Adding new data and function is easy.
Procedural programming does not have any proper way of hiding data so it is **less secure**.	Object-oriented programming provides data hiding so it is **more secure**.
In procedural programming, overloading is not possible.	Overloading is possible in object-oriented programming.

What is C++?

- C++ is a most popular cross-platform programming language which is used to create high-performance applications and software like OS, Games, E-commerce software, etc. It was developed by **Bjarne Stroustrup**, as an extension of C language. C++ give a high level of control over system resources and memory.
- It was developed by Bjarne Stroustrup at Bell Labs in 1983 as an extension of the C programming language

Why Learn C++?

- C++ is one of the most used and popular programming languages.
- C++ is used in making operating systems, embedded systems, and Graphical User Interfaces.
- It is an object-oriented programming language that implements all the OOPs concepts such as Abstraction, Encapsulation, and Inheritance, which gives a clear structure to programs and allows code to be reused, lowering development costs and providing security.
- It is portable and can be used to create applications that can be adapted to multiple platforms.
- C++ is easy to learn so that you can choose it as your first programming language.
- It makes programming easy for programmers to switch to C++ because its syntax is similar to C, Java, and C#



One of the key features of C++ is its ability to support low-level, system-level programming, making it suitable for developing operating systems, device drivers, and other system software.

1. Object-oriented programming

- Object-oriented programming can be defined as a programming model which is based upon the concept of objects. Objects contain data in the form of attributes and code in the form of methods. In object-oriented programming, computer programs are designed using the concept of objects that interact with the real world. Object-oriented programming languages are various but the most popular ones are class-based, meaning that objects are instances of classes, which also determine their types.

- Eg-*Java, C++, C#, Python,*
- *PHP, JavaScript, Ruby, Perl,*
- *Objective-C, Dart, Swift, Scala.*
- Class
- Objects
- Encapsulation
- Polymorphism
- Inheritance
- Abstraction

2. Machine Independent

- A C++ executable is not platform-independent (**compiled programs on Linux won't run on Windows),** however, they are machine-independent. Let us understand this feature of C++ with the help of an example. Suppose you have written a piece of code that can run on Linux/Windows/Mac OSx which makes the C++ Machine Independent but the executable file of the C++ cannot run on different operating systems.

3. Simple

- It is a simple language in the sense that programs can be broken down into logical units and parts, has rich library support and has a variety of data types. Also, the Auto Keyword of C++ makes life easier.

Auto Keyword

- The idea of the auto keyword was to form the C++ compiler to deduce the data type while compiling instead of making you declare the data type every freaking time. Do keep in mind that you cannot declare something without an initializer. There must be some way for the compiler to deduce your type.

Case-sensitive

- It is clear that C++ is a case-sensitive programming language. For example, **cin** is used to take input from the input stream. But if the **“Cin”** won’t work. Other languages like HTML and MySQL are not case-sensitive languages.

- **Compiler Based**

- C++ is a compiler-based language, unlike Python. That is C++ programs used to be compiled and their executable file is used to run them. C++ is a relatively faster language than Java and Python.

- **Dynamic Memory Allocation**

- When the program executes in C++ then the variables are allocated the dynamical heap space. Inside the functions, the variables are allocated in the stack space. Many times, We are not aware in advance how much memory is needed to store particular information in a defined variable and the size of required memory can be determined at run time.

	C	C++
1)	C follows the **procedural style programming.**	C++ is multi-paradigm. It supports both **procedural and object oriented.**
2)	Data is less secured in C.	In C++, you can use modifiers for class members to make it inaccessible for outside users.
3)	C follows the **top-down approach.**	C++ follows the **bottom-up approach.**
4)	C does not support function overloading.	C++ supports function overloading.
5)	In C, you can't use functions in structure.	In C++, you can use functions in structure.
6)	C does not support reference variables.	C++ supports reference variables.
7)	In C, **scanf() and printf()** are mainly used for input/output.	C++ mainly uses stream **cin and cout** to perform input and output operations.
8)	Operator overloading is not possible in C.	Operator overloading is possible in C++.
9)	C programs are divided into **procedures and modules**	C++ programs are divided into **functions and classes.**
10)	C does not provide the feature of namespace.	C++ supports the feature of namespace.
11)	Exception handling is not easy in C. It has to perform using other functions.	C++ provides exception handling using Try and Catch block.
12)	C does not support the inheritance.	C++ supports inheritance

History of C++

- **C++ was initially known as “C with classes, ” and was renamed C++ in 1983.**
- **++ is shorthand for adding one to variety in programming; therefore C++ roughly means that “one higher than C.”**

- **Applications of C++:**
  C++ finds varied usage in applications such as:
- Operating Systems & Systems Programming. e.g. *Linux-based OS (Ubuntu etc.)*
- Browsers *(Chrome & Firefox)*
- Graphics & Game engines *(Photoshop, Blender, Unreal-Engine)*
- Database Engines *(MySQL, MongoDB, Redis etc.)*
- Cloud/Distributed Systems

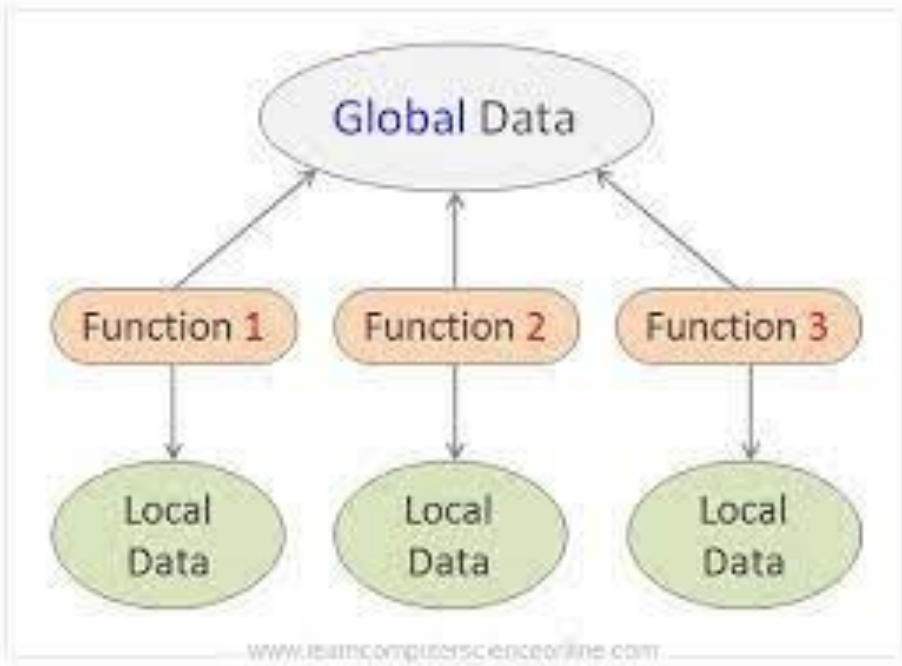
1st C++ program

```
1  #include<iostream>
2
3  int main(){
4      std::cout<<"Hello World";
5      return 0;
6  }
```

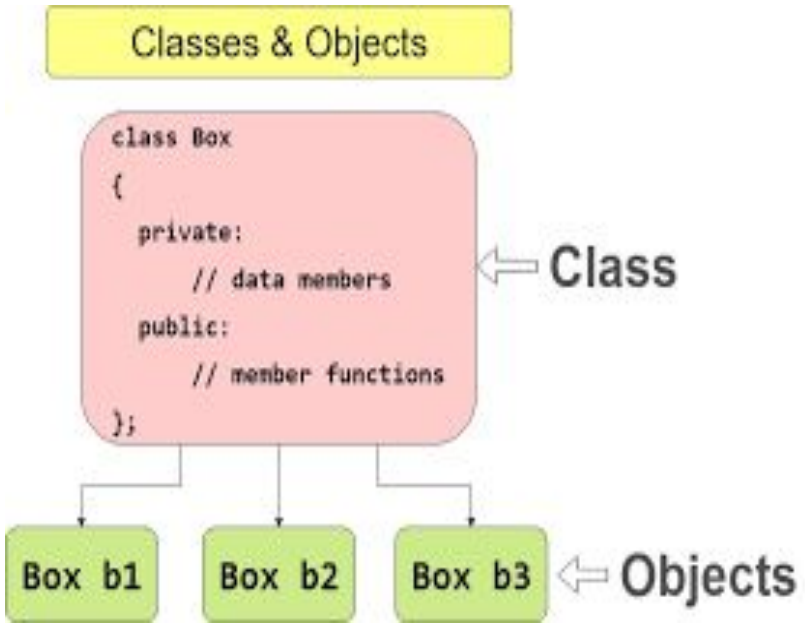
Line 1: "**#include<iostream>"** - this whole text is called a **header file**. **iostream"** is the name of a library, added to our program.
The **iostream** library helps us to get input data and show output data.
std::cout<<" hello world";' - In this line of code "**std"** is a namespace, "::" is the scope resolution operator and "**cout<<"** is a function which is used to output data, "**hello world"**
<<- Insertion operator or put to operator

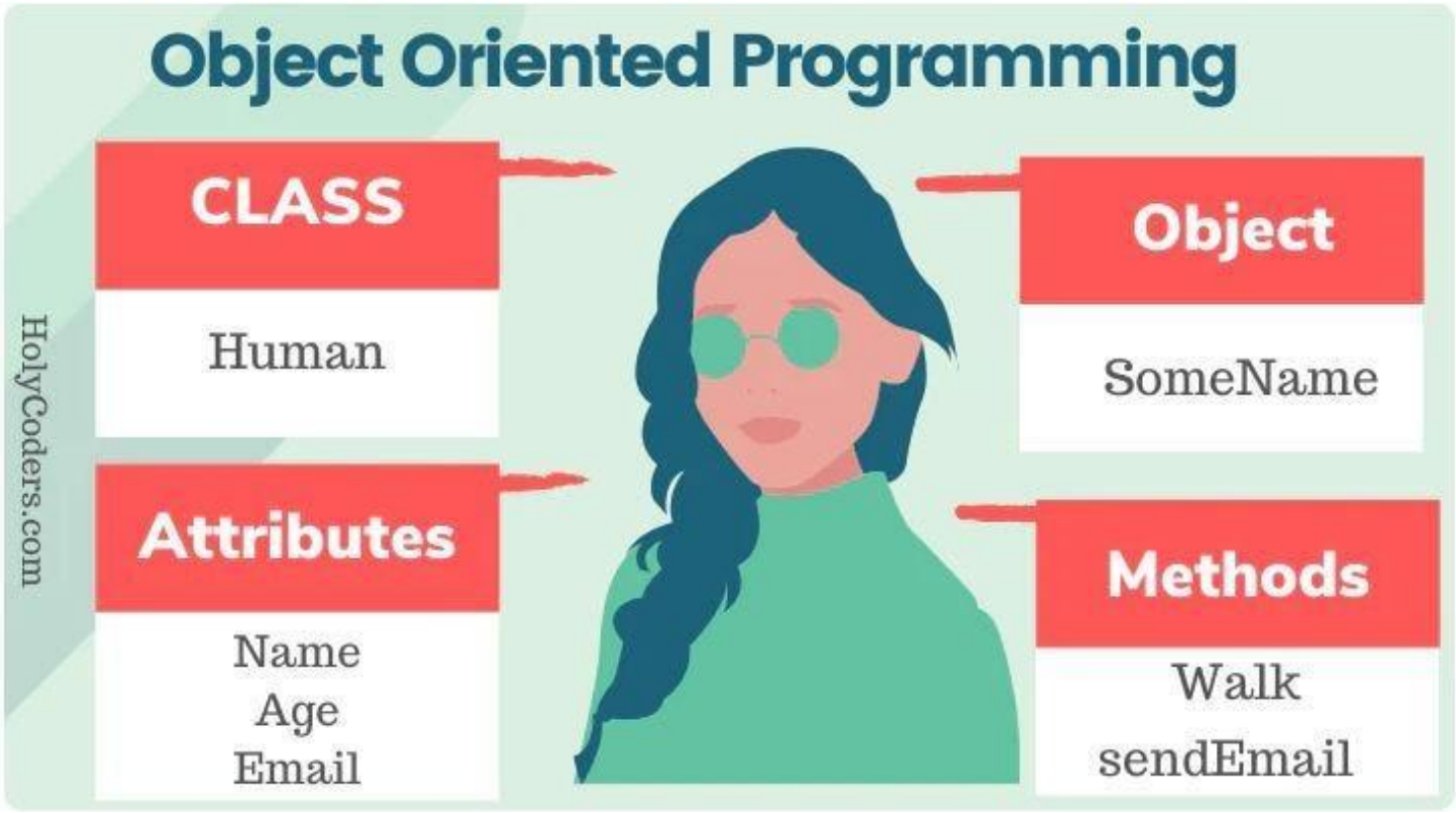
Procedural oriented programming

Problem: As we have global data in POP , the transfer of data is more in POP. Due to transfer of data the data becomes less secure. Any function can update global data



OBJECT ORIENTED PROGRAMMING





CLASS

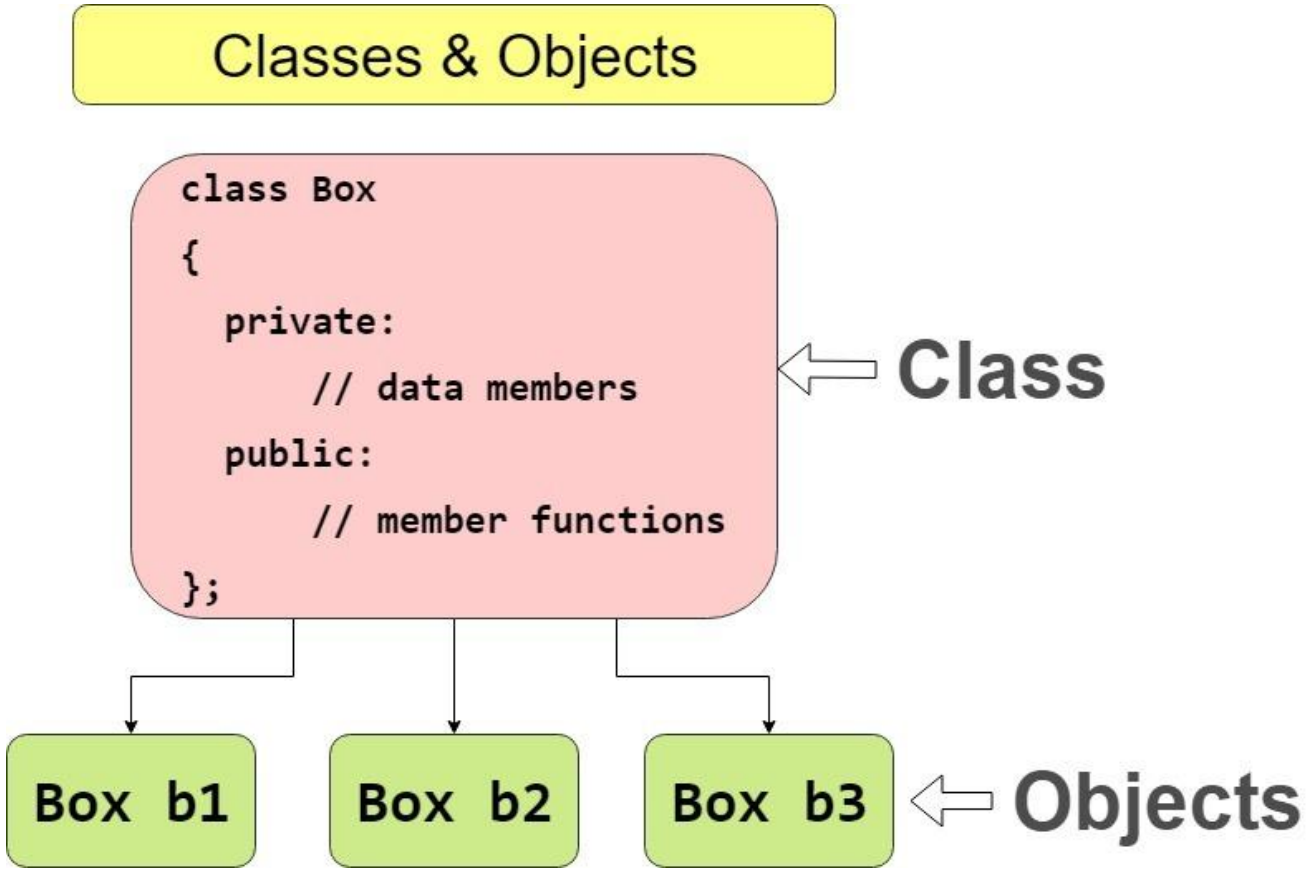
The building block of C++ that leads to Object-Oriented programming is a Class. It is a user-defined data type, which holds its own data members and member functions, which can be accessed and used by creating an instance of that class

- For Example: Consider the Class of Cars. There may be many cars with different names and brands but all of them will share some common properties like all of them will have 4 wheels, Speed Limit, Mileage range, etc. So here, the Car is the class, and wheels, speed limits, and mileage are their properties.

- example of class Car, the data member will be speed limit, mileage, etc and member functions can apply brakes, increase speed, etc.

Object

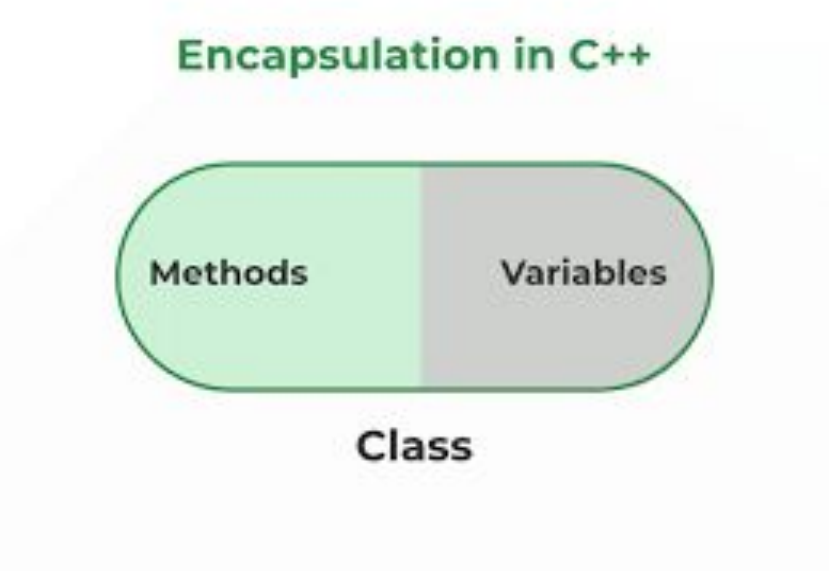
- An Object is an identifiable entity with some characteristics and behavior. An Object is an instance of a Class. When a class is defined, no memory is allocated but when it is instantiated (i.e. an object is created) memory is allocated.



Encapsulation

- In normal terms, Encapsulation is defined as wrapping up data and information under a single unit
- Encapsulation is defined as binding together the data and the functions that manipulate them. Consider a real-life example of encapsulation, in a company, there are different sections like the accounts section, finance section, sales section, etc.

Encapsulation also leads to *data abstraction or data hiding*. Using encapsulation also hides the data. In the above example, the data of any of the sections like sales, finance, or accounts are hidden from any other section.



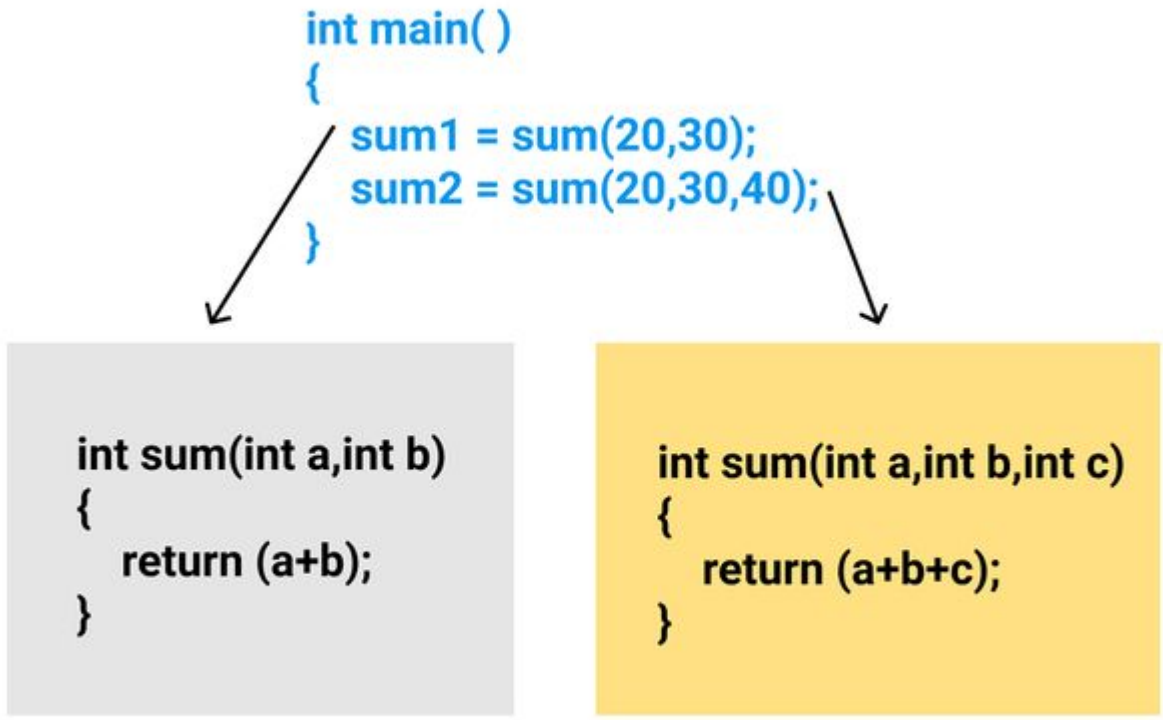
Abstraction

- Data abstraction is one of the most essential and important features of object-oriented programming in C++. Abstraction means displaying only essential information and hiding the details. Data abstraction refers to providing only essential information about the data to the outside world, hiding the background details or implementation. Consider a real-life example of a man driving a car. The man only knows that pressing the accelerator will increase the speed of the car or applying brakes will stop the car but he does not know how on pressing the accelerator the speed is actually increasing, he does not know about the inner mechanism of the car or the implementation of an accelerator, brakes, etc. in the car. This is what abstraction is.

POLYMORPHISM

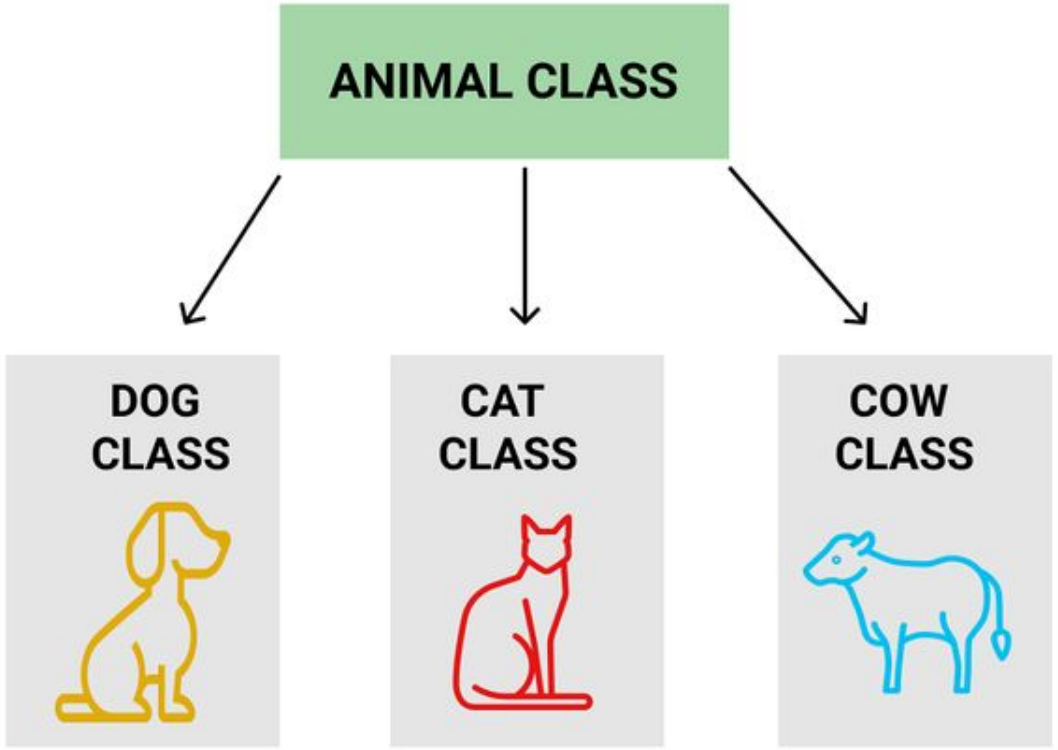
- The word “polymorphism” means having many forms. In simple words, we can define polymorphism as the ability of a message to be displayed in more than one form.

- **Types of Polymorphism:**
- **Compile-time Polymorphism**
- **Runtime Polymorphism**



Inheritance

- The capability of a class to derive properties and characteristics from another class is called Inheritance. Inheritance is one of the most important features of Object-Oriented Programming.
- **Example**: Dog, Cat, Cow can be Derived Class of Animal Base Class.



Data types

- **C++ supports the following data types:**

1. Primary or Built-in or Fundamental data type
2. Derived data types
3. User-defined data types

Data Types in C++ are Mainly Divided into 3 Types:

1. Primitive Data Types: These data types are built-in or predefined data types and can be used directly by the user to declare variables. example: int, char, float, bool, etc. Primitive data types available in C++ are:

- Integer
- Character
- Boolean
- Floating Point
- Double Floating Point
- Valueless or Void
- Wide Character

2. Derived Data Types: Derived data types that are derived from the primitive or built-in datatypes are referred to as Derived Data Types. These can be of four types namely:

- Function
- Array
- Pointer
- Reference

3. Abstract or User-Defined Data Types: Abstract or User-Defined data types are defined by the user itself. Like, defining a class in C++ or a structure. C++ provides the following user-defined datatypes:

- Class
- Structure
- Union
- Enumeration
- Typedef defined Datatype

void

- Uses of void:
- To specify the return type of function which is not returning anything
- to indicate empty list to a function . Eg-Void func19(void);
- Another intresting use of void is to create generic pointer eg- void *gp;

Cout

- The cout object in C++ is an object of class iostream. It is defined in iostream header file. It is used to display the output to the standard output device i.e. monitor. It is associated with the standard C output stream stdout. The data needed to be displayed on the screen is inserted in the standard output stream (cout) using the insertion operator(<<).

```
#include <iostream>
using namespace std;
// Driver Code
int main()
{
    // Print standard output
    // on the screen
    cout << "Welcome to GFG";
    return 0;
}
```

Cin

- The cin object in C++ is an object of class iostream. It is used to accept the input from the standard input device i.e. keyboard. It is associated with the standard C input stream stdin. The extraction operator(>>) is used along with the object cin for reading inputs. The extraction operator (>>)extracts the data from the object cin which is entered using the keyboard.

```
#include <iostream>
using namespace std;
// Driver Code
int main()
{
	string s;

	// Take input using cin
	cin >> s;
		// Print output
	cout << s;
	return 0;
}
```

CASCADING I/O opeartor

- The multiple use of << in one statement is called cascading.

Eg- cout<<“Sum=“<<sum<<“\n” ;

```
cout<<“Sum=“<< sum <<“\n”
    <<“ Average=“ << average <<“\n”;
```

Eg -cout<<“ Sum=“<< sum << “,”

```
    << “ Average=“<< average<< “\n”;
```

Output will be:

Sum=14, average = 7

CASCADING I/O opeartor

- Cin>> number1 >> number 2

Standard end line (endl)

The **endl** is a predefined object of **ostream** class. It is used to insert a new line characters and flushes the stream.

Let's see the simple example of standard end line (endl):

```
#include <iostream>
using namespace std;
int main( ) {
cout << "C++ Tutorial";
cout << " Javatpoint"<<endl;
cout << "End of line"<<endl;
}
```

C++ Identifiers

- C++ identifiers in a program are used to refer to the name of the variables, functions, arrays, or other user-defined data types created by the programmer. They are the basic requirement of any language. Every language has its own rules for naming the identifiers

- **Constants**
- **Variables**
- **Functions**
- **Labels**
- **Defined data types**

A **keyword** is a reserved word. You cannot use it as a variable name, constant name etc.
A list of 32 Keywords in C++ Language which are also available in C language are given below.

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

A list of 30 Keywords in C++ Language which are not available in C language are given below.

asm	dynamic_cast	namespace	reinterpret_cast	bool
explicit	new	static_cast	false	catch
operator	template	friend	private	class
this	inline	public	throw	const_cast
delete	mutable	protected	true	try
typeid	typename	using	virtual	wchar_t

Storage class in C++

C++ Storage Class

Storage Class	Keyword	Lifetime	Visibility	Initial Value
Automatic	auto	Function Block	Local	Garbage
External	extem	Whole Program	Global	Zero
Static	static	Whole Program	Local	Zero
Register	register	Function Block	Local	Garbage
Mutable	mutable	Class	Local	Garbage
Thread Local	thread_local	whole thread	Local or Global	Garbage

Symbolic constant

- There are two ways of carting symbolic constants:

1) Using the qualifier const

2) Defining a set of integer constant using enum keyword

Eg – const int size= 10;

Char name[size];

Declaration of variables