OOPs, Object, Class

### C++ OOPs Concepts

The major purpose of C++ programming is to introduce the concept of object orientation to the C programming language.

Object Oriented Programming is a paradigm that provides many concepts such as inheritance, data binding, polymorphism etc.

The programming paradigm where everything is represented as an object is known as truly object-oriented programming language. Smalltalk is considered as the first truly object-oriented programming language.

##### OOPs (Object Oriented Programming System)

Object means a real word entity such as pen, chair, table etc. Object-Oriented Programming is a methodology or paradigm to design a program using classes and objects. It simplifies the software development and maintenance by providing some concepts:

Cpp Oops concept 1  
Object  
Class  
Inheritance  
Polymorphism  
Abstraction  
Encapsulation

##### C++ Object

In C++, Object is a real world entity, for example, chair, car, pen, mobile, laptop etc.

In other words, object is an entity that has state and behavior. Here, state means data and behavior means functionality.

Object is a runtime entity, it is created at runtime.

Object is an instance of a class. All the members of the class can be accessed through object.

Let’s see an example to create object of student class using s1 as the reference variable.

Student s1; //creating an object of Student  
In this example, Student is the type and s1 is the reference variable that refers to the instance of Student class.

##### C++ Class

In C++, object is a group of similar objects. It is a template from which objects are created. It can have fields, methods, constructors etc.

Let’s see an example of C++ class that has three fields only.

class Student   
 {   
 public:   
 int id; //field or data member   
 float salary; //field or data member   
 String name;//field or data member   
 }

##### C++ Object and Class Example

#include <iostream>   
using namespace std;   
class Student {   
 public:   
 int id;//data member (also instance variable)   
 string name;//data member(also instance variable)   
};   
int main() {   
 Student s1; //creating an object of Student   
 s1.id = 201;   
 s1.name = "Sonoo Jaiswal";   
 cout<<s1.id<<endl;   
 cout<<s1.name<<endl;   
 return 0;   
}

Output

201  
Sonoo Jaiswal

##### C++ Class Example: Initialize and Display data through method

#include <iostream>   
using namespace std;   
class Student {   
 public:   
 int id;//data member (also instance variable)   
 string name;//data member(also instance variable)   
 void insert(int i, string n)   
 {   
 id = i;   
 name = n;   
 }   
 void display()   
 {   
 cout<<id<<" "<<name<<endl;   
 }   
};   
int main(void) {   
 Student s1; //creating an object of Student   
 Student s2; //creating an object of Student   
 s1.insert(201, "Sonoo");   
 s2.insert(202, "Nakul");   
 s1.display();   
 s2.display();   
 return 0;   
}

Output

201 Sonoo  
202 Nakul