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

Object

Any entity that has state and behavior is known as an object. For example: chair, pen, table, keyboard, bike etc. It can be physical and logical.

### Class

**Collection of objects** is called class. It is a logical entity.

An object's class acts as its blueprint.

### Inheritance

**When one object acquires all the properties and behaviours of parent object** i.e. known as inheritance.

### Polymorphism

When **one task is performed by different ways** i.e. known as polymorphism. For example: to convince the customer differently, to draw something e.g. shape or rectangle etc.

### Abstraction

**Hiding internal details and showing functionality** is known as abstraction.

Data abstraction is the process of exposing to the outside world only the information that is absolutely necessary while concealing implementation or background information. For example: phone call, we don't know the internal processing.

### Encapsulation

**Binding (or wrapping) code and data together into a single unit is known as encapsulation.**

**Dynamic Binding -** In dynamic binding, a decision is made at runtime regarding the code that will be run in response to a function call. For this, C++ supports virtual functions.

Why did we require OOPs in C++?

It was not easy to maintain the complex code and plus it made complications with the POPs language. It was not easy to manage if code grows as project size grows.

It was not easy to hide the data whereas in POPs language a global data can be accessed from anywhere. (Security).

POPs was not efficient enough when compared to OOPs and also it help to deal with real world problem if we are using the OOPs language.

First program Of OOPs in C++:

#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;

}

## **Store and Display Employee Information**

#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;

}

#include <iostream>

**using** **namespace** std;

**class** Account {

**public**:

**int** accno; //data member (also instance variable)

       string name;

**static** **int** count;

       Account(**int** accno, string name)

        {

**this**->accno = accno;

**this**->name = name;

            count++;

        }

**void** display()

        {

            cout<<accno<<" "<<name<<endl;

        }

};

**int** Account::count=0;

**int** main(**void**) {

    Account a1 =Account(201, "Sanjay"); //creating an object of Account

    Account a2=Account(202, "Nakul");

     Account a3=Account(203, "Ranjana");

    a1.display();

    a2.display();

    a3.display();

    cout<<"Total Objects are: "<<Account::count;

**return** 0;

}

#include <iostream>

**using** **namespace** std;

**struct** Rectangle

{

**int** width, height;

 };

**int** main(**void**) {

**struct** Rectangle rec;

    rec.width=8;

    rec.height=5;

   cout<<"Area of Rectangle is: "<<(rec.width \* rec.height)<<endl;

**return** 0;

}