

Encapsulation Assignment

By

Narala Praveen

28-jan-2022

Amazon Employee Class:

Program: `class Employee`

```
{  
    private int Empid;  
    private string Name;  
    private string Email;  
    private string Designation;  
    private long Phone;  
    public void AddEmployeeData()  
    {  
        //To do  
    }  
    public void PrintEmployeeData()  
    {  
        //To do  
    }  
    public void EditEmployeeData()  
    {  
        //To do  
    }  
    public void DeleteEmployeeData()  
    {  
        //To do  
    }  
}
```

UML :

Class Employee

-name: String
-EmpId: String
-Email: String
-phno : long
-Role: String

+ Create() : Void
+ Delete() : void
+ Add () : Void
+ Display() : Void
+ Edit () : Void

Amazon Customer Class:

Program class: `class Customer`

```
{
    private int CustomerID;
    private string Name;
    private string Email;
    private string Address;
    private int Phone;
    public void AddCustomerdata()
    {
        //To do
    }
    public void PrintCustomerData()
    {
        //To do
    }
    public void EditCustomerData()
    {
        //To do
    }
    public void DeleteCustomerData()
    {
        //To do
    }
}
```

UML:

Class Customer

-name: String
-Mobile no: Int
-CustomerID: String
-Address: String
-Email: String

+ Create() : Void
+ Delete() : void
+ Add () : Void
+ Display() : Void
+ Edit () : Void

Amazon Product Class Program:

Program: `class Product`

```
{
    private int price;
    private string Name;
    private string Colour;
    private string Brand;
    private string IMEI;
    public void Adddata()
    {
        //To do
    }
    public void PrintData()
    {
        //To do
    }
    public void EditData()
    {
        //To do
    }
    public void DeleteData()
    {
        //To do
    }
}
```

UML:

Class Product

-name: String
-Price: Int
-Brand: String
-IMEI : String
-Colour: String

+ Create() : Void
+ Delete() : void
+ Add () :Void
+ Display():Void
+ Edit () :Void

Amazon Class Order:

Program: class Order

```
{
    private int orderId;
    private int Price;
    private string Track;
    private string Specifications;
    private string DeliveryDate;
    public void Addorderdate()
    {
        //To do
    }
    public void PrintOrderData()
    {
        //To do
    }
    public void EditOrderData()
    {
        //To do
    }
    public void DeleteOrderData()
    {
        //To do
    }
}
```

UML:

Class Order

-OrderID: String
-Price: Int
-Track: String
-Specification: String
-Deliverydate: int

+ Create(): Void
+ Delete(): void
+ Add ():Void
+ Display(): Void
+ Edit ():Void

Amazon Class Seller:

Program: `class Seller`

```
{  
    private string SellerId;  
    private int Phoneno;  
    private string name;  
    private string Adress;  
    private string Product;  
    public void Adddata()  
    {  
        //To do  
    }  
    public void PrintData()  
    {  
        //To do  
    }  
    public void EditData()  
    {  
        //To do  
    }  
    public void DeleteData()  
    {  
        //To do  
    }  
}
```

UMD:

Class Seller

-SellerID: String
-Phoneno: Int
-Product: String
-Address: String
-Name: String

+ Create(): Void
+ Delete(): void
+ Add ():Void
+ Display():Void
+ Edit ():Void

Apollo class Doctors:

Program: class Doctors

```
{  
    private string name;  
    private int Phoneno;  
    private string Email;  
    private string Patientcount;  
    private string Specialization;  
    public void AddDoctorsdata()  
    {  
        //To do  
    }  
    public void PrintDoctorsData()  
    {  
        //To do  
    }  
    public void EditDoctorsData()  
    {  
        //To do  
    }  
    public void DeleteDoctorsData()  
    {  
        //To do  
    }  
}
```

UML:

Class Doctor

-NAME: String
-Phoneno: Int
-Patientcount: String
-Email: String
-Specialization:String

+ Create() : Void
+ Delete() : void
+ Add ():Void
+ Display():Void
+ Edit ():Void

Apollo Class Patient:

Program: `class Patients`

```
{  
    private string name;  
    private int Phoneno;  
    private string Email;  
    private string Disease;  
    private string Address;  
    public void Addpatientdata()  
    {  
        //To do  
    }  
    public void PrintPatientData()  
    {  
        //To do  
    }  
    public void EditPatientData()  
    {  
        //To do  
    }  
    public void DeletePatientData()  
    {  
        //To do  
    }  
}
```

UML:

Class Patient

-NAME: String
-Phoneno: Int
-Address: String
-Disease: String
-EmailID:String

+ Create() : Void
+ Delete() : void
+ Add () :Void
+ Display():Void
+ Edit () :Void

Apollo Non-Medical Staff:

Program: `class Nonmedicalstaff`

```
{
    private string name;
    private int Phoneno;
    private string Email;
    private string Designation;
    private string Address;
    public void Addnonmedicalstaffdata()
    {
        //To do
    }
    public void PrintnonmedicalstaffData()
    {
        //To do
    }
    public void EditnonmedicalData()
    {
        //To do
    }
    public void DeletenonmedicalstaffData()
    {
        //To do
    }
}
```

UML:

Class Nonmedical

-NAME: String
-Phoneno: Int
-Address: String
-Designation: String
-EmailID:String

+ Create() : Void
+ Delete() : void
+ Add () :Void
+ Display():Void
+ Edit () :Void

Apollo Class Medical store:

Program: `class medicalstore`

```
{
    private string medicines;
    private int Phoneno;
    private bool Availability;
    private string incharge;
    private string Address;
    public void Addmedicalstoredata()
    {
        //To do
    }
    public void PrintmedicalstoreData()
    {
        //To do
    }
    public void EditmedicalstoreData()
    {
        //To do
    }
    public void DeletemedicalstoreData()
    {
        //To do
    }
}
```

UML:

Class medicalstore

-incharge: String
-Phoneno: Int
-Address: String
-Availability: Bool
-EmailID:String

+ Create() : Void
+ Delete() : void
+ Add () :Void
+ Display():Void
+ Edit () :Void

Apollo Class Diagnostics:

Program: `class Diagnostics`

```
{
    private string type;
    private int testsno;
    private string Equipment;
    private string incharge;
    private string location;
    public void AddDiagnosticsdata()
    {
        //To do
    }
    public void PrintDiagnosticsData()
    {
        //To do
    }
    public void EditDiagnosticsData()
    {
        //To do
    }
    public void DeleteDiagnosticsData()
    {
        //To do
    }
}
```

UML:

Class Diagnostics

-type: String
-testsno: Int
-location: String
-Equipment: Bool
-EmailID:String

+ Create() : Void
+ Delete() : void
+ Add () :Void
+ Display():Void
+ Edit () :Void

.

Police station Class

Program: `class Station`

```
{  
    private string name;  
    private int stationno;  
    private string type;  
    private string Address;  
    private string Commissionarate;  
    private string zone;  
    public void AddSationdata()  
    {  
        //To do  
    }  
    public void PrintSationData()  
    {  
        //To do  
    }  
    public void EditstaionData()  
    {  
        //To do  
    }  
    public void DeleteStationData()  
    {  
        //To do  
    }  
}
```

UML:

Class Station

-name: String
-depttype: String
-Address: String
-Zone: String
-Sationno: Int

+ Create() : Void
+ Delete() : void
+ Add () : Void
+ Display() : Void
+ Edit () : Void

Police Station Class Staff:

Program: class Staff

```
{
    private string name;
    private int staffcount;
    private string staffshift;
    private string Address;
    private string Designation;
    private int salary;
    public void AddStaffdata()
    {
        //To do
    }
    public void PrintStaffData()
    {
        //To do
    }
    public void EditStaffData()
    {
        //To do
    }
    public void DeleteStaffData()
    {
        //To do
    }
}
```

UML:

Class Staff

-name: String
-Staffcount: int
-Designation: String
-salary: int
-Shift: String

+ Create() : Void
+ Delete() : void
+ Add () :Void
+ Display():Void
+ Edit () :Void

Police Station Class case:

Program: `class Case`

```
{  
    private string name;  
    private int caseno;  
    private string casetype;  
    private string caseassigned;  
    private string casestatus;  
  
    public void AddCasedata()  
    {  
        //To do  
    }  
    public void PrintCaseData()  
    {  
        //To do  
    }  
    public void EditCaseData()  
    {  
        //To do  
    }  
    public void DeleteCaseData()  
    {  
        //To do  
    }  
}
```

UML:

Class case

-name: String
-caseno: int
-casetype: String
-caseassigned: int
-casestatus: String

+ Create() : Void
+ Delete() : void
+ Add () : Void
+ Display() : Void
+ Edit () : Void

Police Station Class Investigation:

Program: `class Investigation`

```
{
    private string name;
    private int number;
    private string type;
    private string under;
    private string status;

    public void Addinvestigationdata()
    {
        //To do
    }
    public void PrintinvestigationData()
    {
        //To do
    }
    public void EditinvestigationData()
    {
        //To do
    }
    public void DeleteinvestigationData()
    {
        //To do
    }
}
```

UML:

Class Investigation

-name: String
-number: int
-type: String
-under: String
-status: String

+ Create() : Void
+ Delete() : void
+ Add () : Void
+ Display() : Void
+ Edit () : Void

Police Station Class accused:

Program: `class Accused`

```
{
    private string name;
    private int number;
    private string type;
    private int section;
    private string status;

    public void AddAccuseddata()
    {
        //To do
    }
    public void PrintAccusedData()
    {
        //To do
    }
    public void EditAccusedData()
    {
        //To do
    }
    public void DeleteAccusedData()
    {
        //To do
    }
}
```

UML:

Class Accused

-name: String
-number: int
-type: String
-Section: Int
-status: String

+ Create() : Void
+ Delete() : void
+ Add () : Void
+ Display() : Void
+ Edit () : Void