Lecture No 3: Concept of OOP

Object Oriented Programing / Paradigm

Object oriented paradigm is new style of programming in which data is not embedded with program. In this paradigm program is independent of data or the type of data.

How to Program in this Paradigm?

Some Assumptions are to be made i-e;

1. Everything in this world is an object like Man, Woman, Plants, animals, Sun, Moon, Earth or any obstacle Pen, Pencil, Rubber, Fan, Tv, Mobile etc

2. Classes implements these objects

Now: What is an Object? How do we say that it is an object? What is a Class?

Object:

General Definition: Any tangible thing for which we want to save Information

Technical Definition:

Object is a collection of Attributes + Behaviors.
Anything/object that has some Attributes (characteristics) and behaviors is called an object.

Object Name ------------- Attributes --------------- Behaviours

Object	Attributes	Behaviors
Man	Eyes Color, Ears, Nose, Hair Color	Walk, Talk, Sleep, Read, Eat, Help Others
Plants	Height, size of trunk, Shape of leaf	Provide Shelter, Provide Fruits, Cleans the environment, Provide Wood
Sun	Shape, Size, Color	Provide Light, Heat, Energy source for plants
Pen	Style, Shape, Color, Height	Write, Draw Shapes

Note: Object is also named as instance

Class:

General Definition:

Collection of Similar object.

The objects that share some common features.
It is a design of an object.
It is a detail of an object.
It tells us what an object contains in it.

Technical Definition:

Class is a blueprint of an object. It contains the detailed definition/implementation of an object.
That is, it defines the object in code or programing language

Example: Man, Women belongs to Human Class

Sun, Moon, Stars belongs to solar system
Pen, Pencil, Rubber, Ruler, Compass, Divider belongs to Stationary Class
Fan, Tv, Mobile belongs to Electronics class

If we again read the concept of OOP paradigm

We can say that here data is an Object and
Class is our Program Code that is independent of type of data

It means that we don't need to change the code for different objects (data) belongs to the same class or group.

Benefits of OOP

1. Security: Abstraction & Encapsulation
2. Re-usability: Inheritance
3. Manageability (Future Enhancement): Polymorphism
4. Easy to debug: Find error or bug easily
5. Scalability: Build large programs or application software with less piece of code, manageable code.

Mainly Four Pillars (features) of OOP

1. **Abstraction**
   Abstraction is the process by which a developer hides everything other than the relevant data about an object in order to simplify and increase efficiency.

2. **Encapsulation**
   Encapsulation is when a group of related methods, properties, and other members are treated as a single object.

3. **Inheritance**
   Inheritance is the ability to receive ("inherit") methods and properties from an existing class.

4. **Polymorphism**
   Polymorphism is when each class implements the same methods in varying ways, but you can still have several classes that can be utilized interchangeably.

Lecture No 3: Concept of OOP

Data Abstraction

This provides essential features without describing any background details. Abstraction is important because it can hide unnecessary details from reference objects to names. It is also necessary for the construction of programs. Instead of showing how an object is represented or how it works, it focuses on what an object does. Therefore, data abstraction is often used for managing large and complex programs.

Encapsulation

This binds the member function and data member into a single class. This also allows for abstraction. Within OOP, encapsulation can be achieved through creating classes. Those classes then display public methods and properties. The name encapsulation comes from the fact that this class encapsulates the set of methods, properties, and attributes of its functionalities to other classes.

Inheritance

Inheritance is a technique that avoids the repetition of the same code and reutilize the existing piece of code again and again. It made code clean and easy to write.
A process of creating new classes from existing classes. New classes inherit some of the properties and behavior of the existing classes. An existing class that is "parent" of a new class is called a base class.

Polymorphism

This is the ability of an object to perform in a wide variety of ways. There are two types:

1. Dynamic polymorphism (runtime time): You can obtain this type through executing function overriding.

2. Static polymorphism (compile time): You can achieve static polymorphism through function overloading and operator overloading.

Within OOP, polymorphism can be achieved using many techniques including:

- ***Method overloading*** (defining several methods at the same time)
- ***Method overriding*** (this allows a subclass to override a specific implementation of a method already issued by one of its super-classes)
- ***Operator overloading*** (some or all of the operators are handled has polymorphic functions with different behaviors depending on the types of its arguments)