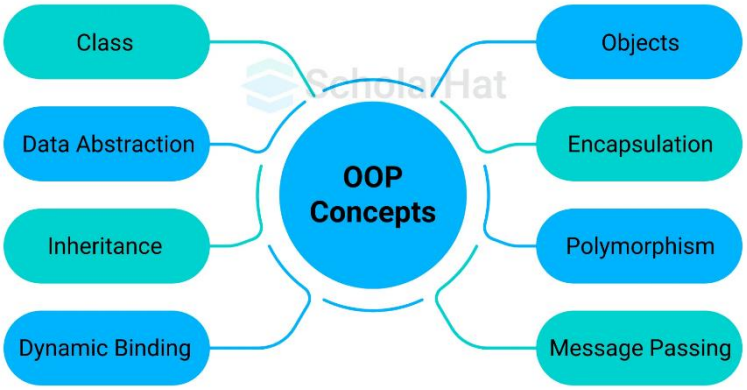
1.2 Object Oriented Programming Paradigm:

The **Object Oriented Programming Paradigm** is a programming approach in which software is designed around **objects**, which represent **real-world entities**. These objects contain **data (attributes)** and **functions (behaviors)** that operate on the data. OOP helps create modular, reusable, flexible, and easily maintainable programs. Object Oriented Programming Paradigm are

- Class
- Objects
- Data Abstraction
- Encapsulation
- Inheritance
- Polymorphism
- Dynamic Binding
- Message Passing



1. Class:

A class is a user-defined data type. It consists of data members and member functions, which can be accessed and used by creating an instance of that class. A class represents a set of properties or methods that are common to all objects of particular type. Thus, a class acts as blueprint for an object. The entire set of data and code of an object can be made a user-defined data type with the help of a **CLASS.** Therefore, a class is also known as an

Abstract Data Type (ADT). No memory is allocated when a class is defined. Memory is allocated only when an object is created, i.e., when an instance of a class is created.

For example, think about a class called **Car**. There are many different cars with different names and brands, but they all share common features such as **4 wheels, a speed limit, and a mileage range**. So, **Car** is the class, and **wheels, speed limit, and mileage** are its properties.

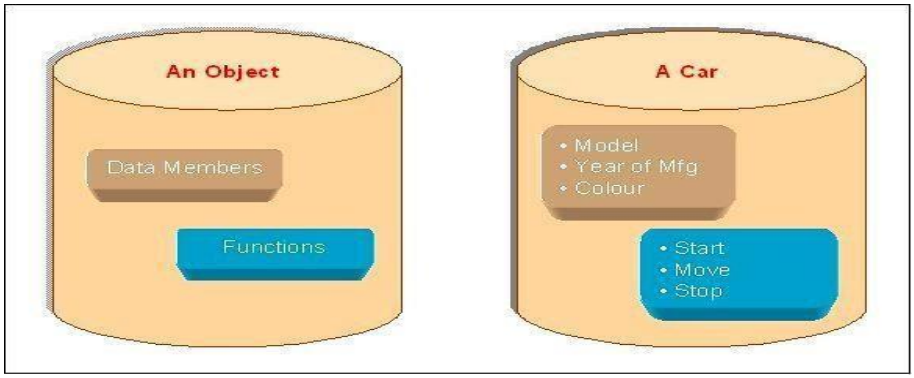
2. Object:

- ✓ It is a basic unit of Object-Oriented Programming and represents the real-life entities. 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.
- ✓ An object has three main characteristics: **identity, state, and behavior**. Each object contains data and code(methods/functions) to manipulate the data. Objects interact with one another by sending messages during program execution.
- ✓ Objects can interact with each other without knowing the internal details of each other's data or code.
- ✓ An object is considered to be a partitioned area of computer memory that stores data and set of operations that can access and modify that data.
- ✓ An Object is a collection of data members and associated member functions also known as methods.

For example "Dog" is a real-life Object, which has some characteristics like color, Breed, Bark, Sleep, and Eats.

3. Data Abstraction:

Data abstraction is one of the most essential and important features of object-oriented programming. Data abstraction refers to providing only essential information about the data to the outside world, hiding



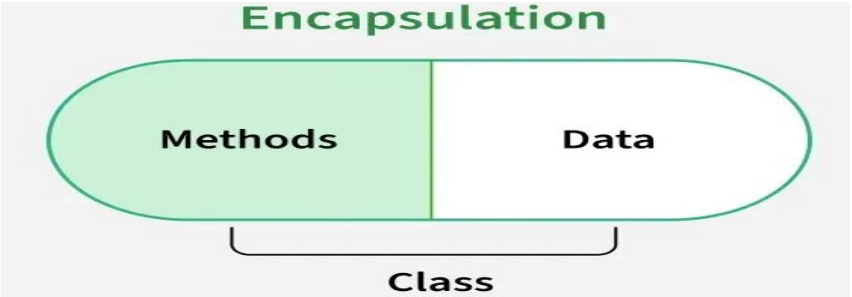
the background details or implementation. Since the classes use the concept of data abstraction, they are known as Abstract Data Types (ADTs).

Consider a **real-life example of a man driving a car**. The man only knows that pressing the accelerators will increase the speed of the car or applying brakes will stop the car, but he does not know about how on pressing the accelerator the speed is increasing, he does not know about the inner mechanism of the car or the implementation of the accelerator, brakes, etc in the car. This is what abstraction is.

4. Encapsulation:

The wrapping up of data and functions into a single unit is known as Encapsulation. It is the mechanism that binds together code and the data it manipulates. In encapsulation, the data of a class is hidden from outside access, and only the member functions of the class can access or modify it. Thus, the variables or data of a class are not directly accessible by other classes and can be accessed only through the member functions of the class in which they are declared. Since encapsulation hides data from unauthorized access, it is also known as **data hiding**.

Encapsulation means keeping data safe and allowing access only through proper channels.



For example, in a company, the sales department keeps its own data, and the finance department cannot access it directly. If finance needs sales information, they must request it from an authorized sales officer. The data and the people who manage it are grouped together inside the "sales section." This controlled access is exactly what encapsulation is.

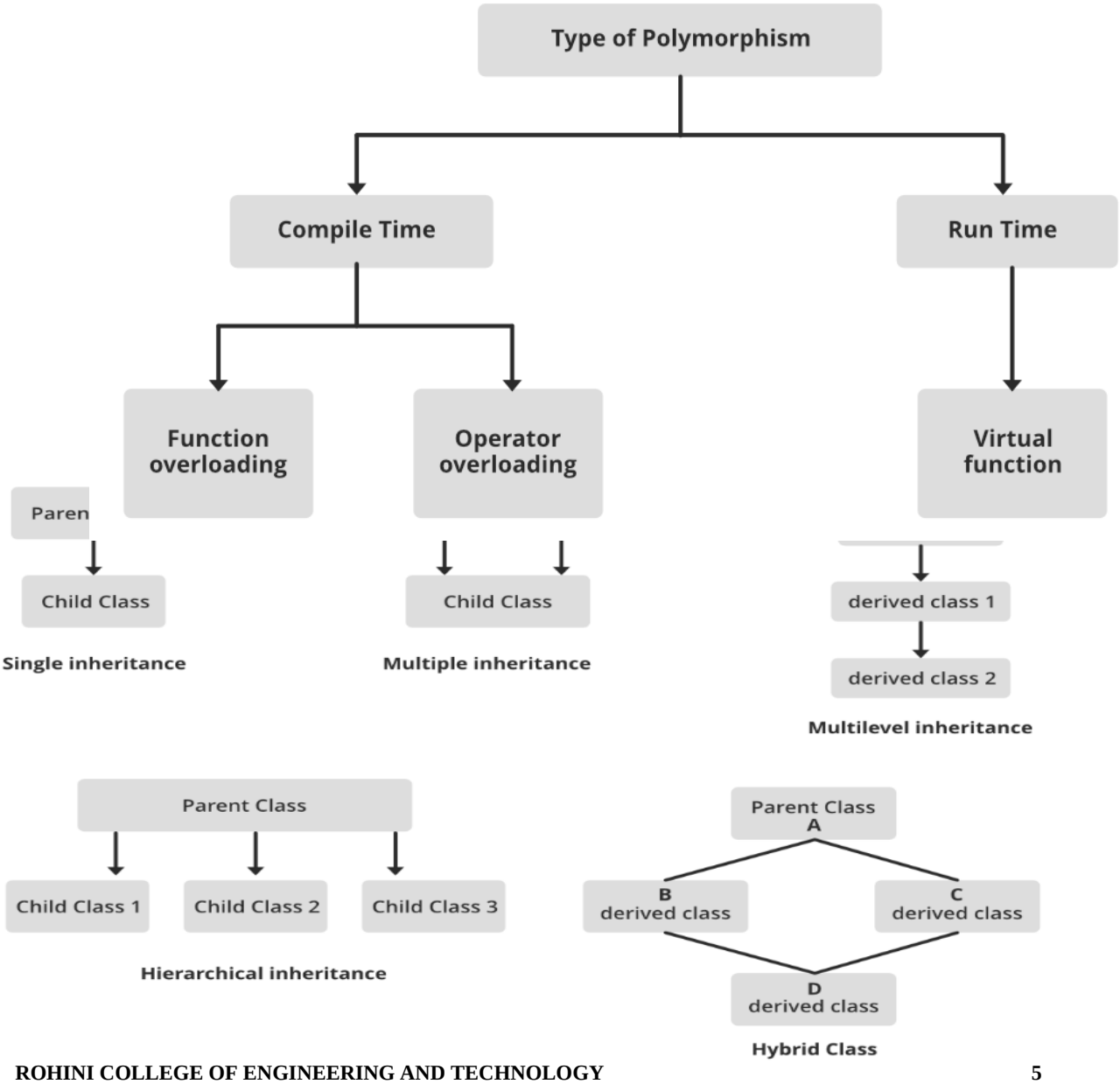
5. Inheritance:

- ✓ Inheritance is the process by which objects of one class acquire the properties of objects of another class.
- ✓ That is, deriving a new class from existing class. The existing class is called the **base class**, and the newly derived class is called the **derived class**.
- ✓ In Object-Oriented Programming, inheritance provides the concept(idea) of **reusability**. This means that new features can be added to an existing class without modifying it. This is possible by deriving a new class from the existing one, so the new(derived) class will have the combined features of both the classes.
- ✓ The capability of a class to derive properties and characteristics from another class is called Inheritance.
- ✓ Inheritance is one of the important pillars of Object-Oriented Programming.

- ✓ It allows programmers to reuse existing code, thereby reducing redundancy and improving code maintainability. Using inheritance, we do not need to write the same properties and functions repeatedly, as they can be inherited from another class.

6. Polymorphism:

The word polymorphism means having many forms. In Object-Oriented Programming, polymorphism is the ability of a message or function to be represented in more than one form. In simple terms, the same entity can behave differently in different situations. For example, A person at the same time can have different characteristics. A person can



play multiple roles at the same time, such as a father, a husband, and an employee. Although the person is the same, the behavior varies according to the role and situation. This concept is known as polymorphism. In programming, an operation may exhibit different behaviors in different instances, depending on the type of data used. **Function overloading** and **operator overloading** are the examples of polymorphism.

- ✓ For example, an operation may exhibit different behavior in different instances. The behavior depends upon the types of data used in the operation.

7. Dynamic Binding:

In dynamic binding, is the process in which the code to be executed in response to the function call is decided at runtime. In Dynamic binding, the code associated between a function call and the corresponding function definition is not known until the program is executed at the run time. This is also referred to as **dynamic method binding**. One of the main advantages of inheritance is that a derived class inherits all the public members of its base class.

If the derived class does not hide any of the public members of the base class, an object of the derived class can be used wherever an object of the base class is expected. This feature is known as **subtype polymorphism** and is achieved through dynamic binding.

8. Message Passing:

- ✓ It is a form of communication used in object-oriented programming as well as parallel programming.
- ✓ Objects communicate with one another by sending and receiving information to each other.
- ✓ A message for an object is a request for execution of a procedure and therefore will invoke a function in the receiving object that generates the desired results.

- ✓ Message passing involves specifying the name of the object, the name of the function, and the information to be sent.
- ✓ Objects have a life cycle; they can be created, used, and destroyed during program execution.

Need for object-oriented programming

- To make the development and maintenance of projects more effortless.
- To provide the feature of data hiding that is good for security concerns.
- We can solve real-world problems if we are using object-oriented programming.
- It ensures code reusability.
- It lets us write generic code: which will work with a range of data, so we don't have to write basic stuff over and over again.