1. create multilevel inheritance for

//Vehicle

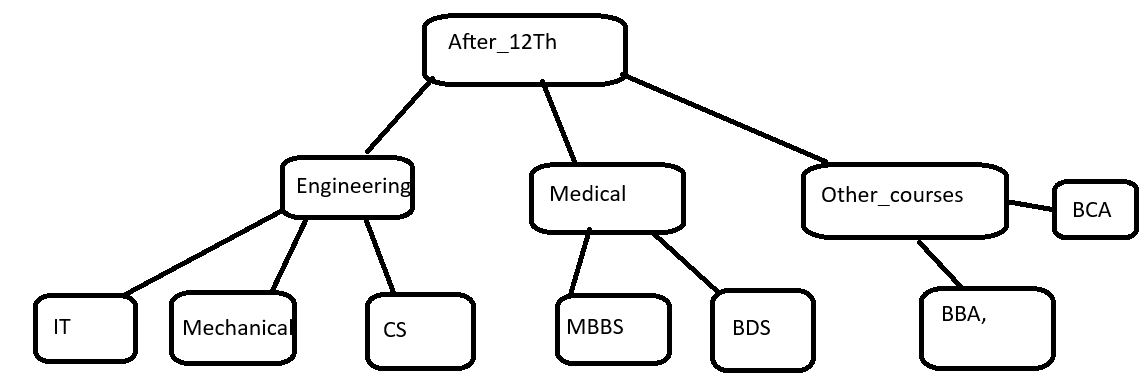
//Four\_wheeler

//Petrol\_Four\_Wheeler

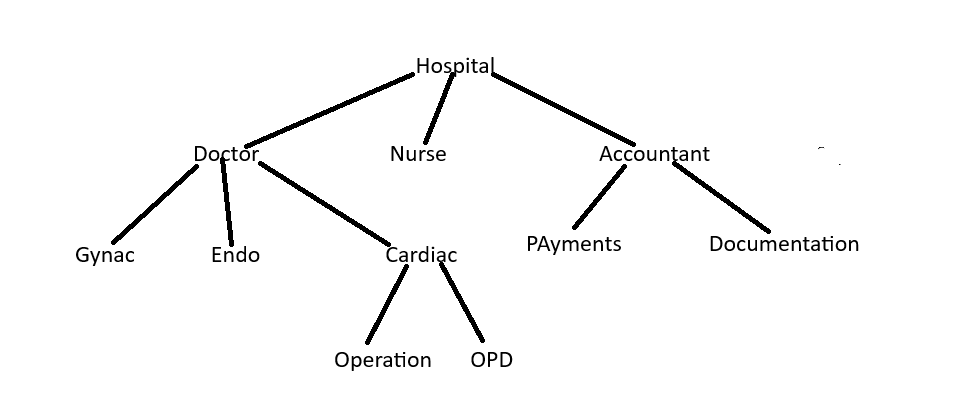
//FiveSeater\_Petrol\_Four\_Wheeler

//Baleno\_FiveSeater\_Petrol\_Four\_Wheeler

1. Demonstrate the use of the super keyword
2. Create Hospital super class and access this class inside the patient child class and access properties from Hospital class.
3. Create Hierarchical inheritance



1. Create practice on this



Polymorphism

1. Create a class Calculator with the following overloaded add()

1.add(int a, int b)

2.add(int a, int b, int c)

3.add(double a, double b)

1. Create a base class Shape with a method area() that prints a message. Then create two subclasses Circle🡪override area() to calculator and print area of circle Rectangle🡪 override area() to calculate and print area of a rectangle
2. Create a Bank class with a method getInterestRate() create subclasses: SBI🡪return 6.7% ICICI🡪return 7.0% HDFC🡪return 7.5%
3. Runtime Polymorphism with constructor Chaining create a class vehicle with a constructor that prints “Vehicle Created”

Create a subclass Bike that override a method and uses super() in constructor

Combined question

Create an abstract class SmartDevice with methods like turnOn(), turnOff(), and performFunction().  
Create child classes:

* SmartPhone: performs calling and browsing.
* SmartWatch: tracks fitness and time.
* SmartSpeaker: plays music and responds to voice commands.
* Write code to store all objects in an array and use polymorphism to invoke their performFunction().

**2.**Design an interface Bank with methods deposit(), withdraw(), and getBalance().  
Implement this in SavingsAccount and CurrentAccount classes.

* Use inheritance to create a base Account class.
* Demonstrate method overriding with customized logic for withdrawal (e.g., minimum balance in SavingsAccount).

**3**

Create a base class Vehicle with method start().  
Derive Car, Bike, and Truck from it and override the start() method.

* Create a static method that accepts Vehicle type and calls start().
* Pass different vehicle objects to test polymorphism.

**4.**

Design an abstract class Person with fields like name, age, and abstract method getRoleInfo().  
Create subclasses:

* Student: has course and roll number.
* Professor: has subject and salary.
* TeachingAssistant: extends Student and implements getRoleInfo() in a hybrid way.
* Create and print info for all roles using overridden getRoleInfo().

5.Create:

* Interface Drawable with method draw()
* Abstract class Shape with abstract method area()  
  Subclasses: Circle, Rectangle, and Triangle.
* Calculate area using appropriate formulas.
* Demonstrate how interface and abstract class work together.