**1. Aim:**  
To implement different types of inheritance in TypeScript to demonstrate the concept of object-oriented programming (OOP).

**2. Theory:**  
Inheritance is a fundamental principle of Object-Oriented Programming (OOP) that allows a class to derive properties and methods from another class. In TypeScript, inheritance is implemented using the extends keyword. The class that is inherited from is called the **parent (superclass)**, and the class that inherits is called the **child (subclass)**.

**Types of Inheritance in TypeScript:**

1. **Single Inheritance:** A class inherits from only one parent class.
2. **Multilevel Inheritance:** A class inherits from another derived class, forming a chain.
3. **Hierarchical Inheritance:** Multiple child classes inherit from a single parent class.
4. **Multiple Inheritance (Through Interfaces):** TypeScript does not support multiple inheritance directly, but interfaces can be used to implement this behavior.

**3. Code:**

// Single Inheritance

class Animal {

name: string;

constructor(name: string) {

this.name = name;

}

makeSound(): void {

console.log("Some generic sound");

}

}

class Dog extends Animal {

breed: string;

constructor(name: string, breed: string) {

super(name);

this.breed = breed;

}

makeSound(): void {

console.log("Bark!");

}

}

// Multilevel Inheritance

class Puppy extends Dog {

age: number;

constructor(name: string, breed: string, age: number) {

super(name, breed);

this.age = age;

}

play(): void {

console.log(`${this.name} is playing!`);

}

}

// Hierarchical Inheritance

class Cat extends Animal {

constructor(name: string) {

super(name);

}

makeSound(): void {

console.log("Meow!");

}

}

// Multiple Inheritance using Interfaces

interface Flyable {

fly(): void;

}

class Bird extends Animal implements Flyable {

constructor(name: string) {

super(name);

}

makeSound(): void {

console.log("Chirp!");

}

fly(): void {

console.log(`${this.name} is flying!`);

}

}

// Creating instances and testing

let myDog = new Dog("Buddy", "Golden Retriever");

myDog.makeSound();

let myPuppy = new Puppy("Charlie", "Labrador", 1);

myPuppy.makeSound();

myPuppy.play();

let myCat = new Cat("Whiskers");

myCat.makeSound();

let myBird = new Bird("Tweety");

myBird.makeSound();

myBird.fly();

**4. Output:**

Bark!

Bark!

Charlie is playing!

Meow!

Chirp!

Tweety is flying!

**5. Conclusion:**  
In this practical, we implemented different types of inheritance in TypeScript, including single, multilevel, hierarchical, and multiple inheritance using interfaces. This helps in achieving code reusability and better organization in object-oriented programming.