Date: 28-04-23

Experiment - 5.1

Aim:

Write a program to implement method overriding.

Code:

```
class Main{
   void speak(){
      System.out.println("Speaking");
   }

   public static class Human extends Main{
      void speak(){
         System.out.println("Speaking as Human");
      }
   }

   public static void main(String[] args) {
                  Human p1 = new Human();
                  p1.speak();
         }
}
```

Output:

```
Speaking as Human
```

Date: 28-04-23

Experiment - 5.2

Aim:

Write a program to illustrate simple Inheritance.

Code:

```
class Main{
   void speak(){
      System.out.println("Speaking");
   }

   public static class Human extends Main{
      void walk(){
         System.out.println("Walking as human");
      }
   }

   public static void main(String[] args) {
                  Human p1 = new Human();
                  p1.speak();
                  p1.walk();
         }
}
```

Output:

```
Speaking
Walking as human
```

Date: 28-04-23

Experiment - 5.3

Aim:

Write a program to illustrate multilevel Inheritance.

Code:

```
class Main{
   void speak(){
      System.out.println("Speaking");
   }

   public static class Mammals extends Main{
      void eat(){
         System.out.println("Eating");
      }
   }

   public static class Human extends Mammals{
      void walk(){
         System.out.println("Walking");
      }
   }

   public static void main(String[] args) {
                  Human p1 = new Human();
                  p1.speak();
                  p1.eat();
                  p1.walk();
         }
}
```

Output:

```
Speaking
Eating
Walking
```

Date: 28-04-23

Experiment - 5.4

Aim:

Write a program illustrating all uses of super keywords.

Code:

```
public class Main {
    protected int number, a = 5;
    public Main(int number) {
      this.number = number;
    }
    public void printNumber() {
      System.out.println("Number in ParentClass: " + number);
    }
    public static class ChildClass extends Main {
      private int anotherNumber, a = 10;
      public ChildClass(int number, int anotherNumber) {
        super(number);  // Calling the superclass constructor using 'super'
        this.anotherNumber = anotherNumber;
      }
      public void printNumbers() {
        super.printNumber();  // Calling the superclass method using 'super'
        System.out.println("Number in ChildClass: " + number);
        System.out.println("Another Number in ChildClass: " + anotherNumber);
        System.out.println("a = " + super.a);  // 'super' used here to reference base class's 'a'
      }
    }
  public static void main(String[] args) {
    ChildClass child = new ChildClass(10, 20);
    child.printNumbers();
  }
}
```

Output:

```
Number in ParentClass: 10
Number in ChildClass: 10
Another Number in ChildClass: 20
a = 5
```

Date: 28-04-23

Experiment - 5.5

Aim:

Write a program to show dynamic polymorphism and interface.

Code:

```
public class Main {
   public static void main(String[] args) {
      Animal animal1 = new Dog();
      Animal animal2 = new Cat();
      animal1.sound();   // Dog barks
      animal2.sound();   // Cat meows
   }
}

interface Animal {
   void sound();
}

class Dog implements Animal {
   public void sound() {
      System.out.println("Dog barks");
   }
}

class Cat implements Animal {
   public void sound() {
      System.out.println("Cat meows");
   }
}
```

Output:

```
Dog barks
Cat meows
```