Q1: class Animal {

    private String name;

    private String food;

    private String sound;

    private String movement;

    public Animal(String name, String food, String sound, String movement) {

        this.name = name;

        this.food = food;

        this.sound = sound;

        this.movement = movement;

    }

    public void makeSound() {

        System.out.println(name + " makes a " + sound + " sound.");

    }

    public void move() {

        System.out.println(name + " moves " + movement + ".");

    }

    public void food() {

        System.out.println(name + " eats " + food + ".");

    }

}

class Lion extends Animal {

    public Lion() {

        super("Lion", "meat", "roar", "gracefully");

    }

}

class Hippo extends Animal {

    public Hippo() {

        super("Hippo", "plants", "grunt", "heavily");

    }

}

class Tiger extends Animal {

    public Tiger() {

        super("Tiger", "meat", "roar", "stealthily");

    }

}

class Wolf extends Animal {

    public Wolf() {

        super("Wolf", "meat", "howl", "in a pack");

    }

}

class Dog extends Animal {

    public Dog() {

        super("Dog", "varied", "bark", "playfully");

    }

}

class Cat extends Animal {

    public Cat() {

        super("Cat", "varied", "meow", "gracefully");

    }

}

class Horse extends Animal {

    public Horse() {

        super("Horse", "plants", "neigh", "elegantly");

    }

}

class Pigeon extends Animal {

    public Pigeon() {

        super("Pigeon", "seeds", "coo", "in flight");

    }

}

public class AnimalSimulation {

    public static void main(String[] args) {

        Lion lion = new Lion();

        Hippo hippo = new Hippo();

        Tiger tiger = new Tiger();

        lion.makeSound();

        hippo.move();

        tiger.food();

    }

}

Q2: import java.lang.Math.\*;

class Shape {

    public double calculateArea(){

        return 0.0;

    }

    public double calculatePerimeter(){

        return 0.0;

    }

}

class Circle extends Shape{

    private double radius;

    Circle(double rad){

        radius = rad;

    }

    @Override

    public double calculateArea(){

        return Math.PI\*radius\*radius;

    }

    @Override

    public double calculatePerimeter(){

        return 2\*Math.PI\*radius;

    }

}

class Rectangle extends Shape{

    private double length;

    private double breadth;

    Rectangle(double len, double bread){

        length= len;

        breadth= bread;

    }

    @Override

    public double calculateArea(){

        return length\*breadth;

    }

    @Override

    public double calculatePerimeter(){

        return 2\*(length+breadth);

    }

}

class ShapeTest{

    public static void main(String[] args) {

    Circle c1 = new Circle(5.9);

    Rectangle r1 = new Rectangle(5, 6);

    System.out.println("Area of Circle: "+c1.calculateArea());

    System.out.println("Perimeter of Circle: "+c1.calculatePerimeter());

    System.out.println("Area of Rectangle: "+r1.calculateArea());

    System.out.println("Perimeter of Rectangle: "+r1.calculatePerimeter());

    }

}

Q3: class Person {

    public String name;

    public String gender;

    public int age;

    public void displayBasic(){

        System.out.println("Name: " + name);

        System.out.println("Age: " + age);

        System.out.println("Gender: "+gender);

    }

}

class Student extends Person {

    private int studentID;

    private double grade;

    Student(int id, String nam, String gen, int ag, double grad) {

        studentID = id;

        grade = grad;

        name = nam;

        gender = gen;

        age = ag;

    }

    public void displayInfo() {

        System.out.println("Student ID: " + studentID);

        System.out.println("Grade: " + grade);

        this.displayBasic();

    }

    public void calculateGradeLevel() {

        if (grade >= 9.5) {

            System.out.println("Grade level: A+");

        } else if (grade >= 8.5) {

            System.out.println("Grade level: A");

        } else if (grade >= 7.5) {

            System.out.println("Grade level: B+");

        } else if (grade >= 6.5) {

            System.out.println("Grade level: B");

        } else {

            System.out.println("Grade level: C");

        }

    }

}

class Teacher extends Person {

    private int teacherID;

    private String subject;

    Teacher(int id, String nam, String gen, int ag, String sub) {

        teacherID = id;

        name = nam;

        gender = gen;

        age = ag;

        subject = sub;

    }

    public void displayInfo() {

        System.out.println("Teacher ID: " + teacherID);

        System.out.println("Subject: " + subject);

        this.displayBasic();

    }

}

class AdministrativeStaff extends Person {

    private int staffID;

    private String role;

    AdministrativeStaff(int id, String nam, String gen, int ag, String rol) {

        staffID = id;

        role = rol;

        name = nam;

        gender = gen;

        age = ag;

    }

    public void displayInfo() {

        System.out.println("Staff ID: " + staffID);

        System.out.println("Role: " + role);

        this.displayBasic();

    }

}

class Principal extends Person {

    private int principalID;

    private int yearsOfExperience;

    Principal(int id, String nam, String gen, int ag, int exp) {

        principalID = id;

        yearsOfExperience = exp;

        name = nam;

        gender = gen;

        age = ag;

    }

    public void displayInfo() {

        System.out.println("Principal ID: " + principalID);

        System.out.println("Years of experience: " + yearsOfExperience);

        this.displayBasic();

    }

}

public class SchoolManage{

    public static void main(String[] args){

        Student stu = new Student(1, "Stu1", "Male", 19, 7.5);

        stu.displayInfo();

        stu.calculateGradeLevel();

        Teacher teacher = new Teacher(2, "Teacher1", "Male", 45, "Math");

        teacher.displayInfo();

        AdministrativeStaff staff = new AdministrativeStaff(3, "Staff1", "Female", 55, "Principal");

        staff.displayInfo();

        Principal pri = new Principal(3, "Principal", "Female", 55, 12);

        pri.displayInfo();

    }

}

Q4: class Person{

    String name;

    int age;

}

interface Course{

    public int getcourseCode();

    public String getcourseName();

}

class Student extends Person{

    int studentID;

    int batch;

    Student(String name,int age,int studentID,int batch){

        this.name = name;

        this.age = age;

        this.studentID=studentID;

        this.batch=batch;

    }

    void display(){

        System.out.println("Name: "+name);

        System.out.println("Age: "+age);

        System.out.println("Student ID: "+studentID);

        System.out.println("Batch: "+batch);

    }

    int calculateAge(){

        return age;

    }

}

class Faculty extends Person{

    int employeeID;

    String department;

    Faculty(String name,int age,int employeeID,String department){

        this.name = name;

        this.age = age;

        this.employeeID=employeeID;

        this.department=department;

    }

    void display(){

        System.out.println("Name: "+name);

        System.out.println("Age: "+age);

        System.out.println("Employee ID: "+employeeID);

        System.out.println("Department: "+department);

    }

    int calculateAge(){

        return age;

    }

}

class CourseEnrollment extends Student implements Course{

    int enrollmentID;

    String semester;

    int courseCode;

    String courseName;

    CourseEnrollment(String name,int age,int studentID,int batch,int enrollmentID,String semester, int courseCode, String courseName){

        super(name,age,studentID,batch);

        this.enrollmentID=enrollmentID;

        this.semester=semester;

        this.courseCode=courseCode;

        this.courseName=courseName;

    }

    public int getcourseCode(){

        return courseCode;

    }

    public String getcourseName(){

        return courseName;

    }

    void display(){

        System.out.println("Student Name: "+name);

        System.out.println("Age: "+age);

        System.out.println("Enrollment ID: "+enrollmentID);

        System.out.println("Semester: "+semester);

        System.out.println(courseName+": "+courseCode);

    }

    int calculateDuration(){

        return 4;

    }

}

class TeachingAssignment extends Faculty{

    int assignmentID;

    String academicYear;

    TeachingAssignment(String name,int age,int employeeID,String department,int assignmentID,String academicYear){

        super(name,age,employeeID,department);

        this.assignmentID=assignmentID;

        this.academicYear=academicYear;

    }

    void display(){

        super.display();

        System.out.println("Assignment ID: "+assignmentID);

        System.out.println("Academic Year: "+academicYear);

    }

    int calculateDuration(){

        return 4;

    }

}

public class ExecInstitute{

    public static void main(String[] args) {

        Student s = new Student("Rahul", 20, 123, 2020);

        s.display();

        System.out.println("Age: "+s.calculateAge());

        System.out.println();

        Faculty f = new Faculty("Raj", 40, 456, "CSE");

        f.display();

        System.out.println("Age: "+f.calculateAge());

        System.out.println();

        CourseEnrollment ce = new CourseEnrollment("Rahul", 20, 123, 2020, 789, "Spring", 101, "Java");

        ce.display();

        System.out.println("Duration: "+ce.calculateDuration()+" years");

        System.out.println();

        TeachingAssignment ta = new TeachingAssignment("Raj", 40, 456, "CSE", 987, "2020-21");

        ta.display();

        System.out.println("Duration: "+ta.calculateDuration()+" years");

    }

}