**Question 1)**

b) c3 and c6 objects can call any methods in both classes , but the other hand other objects can’t call Cylinder class (child class) methods. Because they are in Circle reference .

c) Constructor overloading

**Cylinder.java**

public class Cylinder extends Circle {  
 double height = 1.0;  
  
 public Cylinder() {  
 super();  
 }  
  
 public Cylinder(double radius) {  
 super(radius);  
 }  
  
 public Cylinder(double radius,double height) {  
 super(radius);  
 this.height = height;  
 }  
  
  
 public Cylinder(double radius, double height, String color) {  
 super(radius, color);  
 this.height = height;  
 }  
  
 public double getHeight() {  
 return height;  
 }  
  
 public void setHeight(double height) {  
 this.height = height;  
 }  
  
 public double getVolume() {  
 return Math.*PI* \* radius \* radius \* height;  
 }  
}

**Circle.java**

public class Circle {  
 double radius = 1.0;  
 String color = "red";  
  
 public Circle(double radius, String color) {  
 this.radius = radius;  
 this.color = color;  
 }  
  
 public Circle() {  
  
 }  
  
 public Circle(double radius) {  
 this.radius = radius;  
 }  
  
 public double getRadius() {  
 return radius;  
 }  
  
 public void setRadius(double radius) {  
 this.radius = radius;  
 }  
  
 public String getColor() {  
 return color;  
 }  
  
 public void setColor(String color) {  
 this.color = color;  
 }  
  
 public double getArea() {  
 return Math.*PI* \* radius \* radius;  
 }  
  
 public String toString() {  
 return "Circle[radius=" + radius + ", color=" + color + "]";  
 }  
}

**testDriver.java**

public class testDriver {  
 public static void main(String[] args) {  
 Circle c1 = new Circle();  
 Circle c2 = new Cylinder();  
 Cylinder c3 = new Cylinder (3.1);  
 Circle c4 = new Circle(2.3);  
 Circle c5 = new Cylinder (2.4, 4.0);  
 Cylinder c6 = new Cylinder (3.1, 5.2, "red");  
 Circle c7 = new Circle(2.1, "yellow");  
  
 System.out.println(c1.getRadius());  
 System.out.println(c1.getArea());  
 System.out.println(c1.getColor());  
 System.out.println("------------");  
  
 System.out.println(c2.getRadius());  
 System.out.println(c2.getArea());  
 System.out.println(c2.getColor());  
 System.out.println("------------");  
  
 System.out.println(c3.getRadius());  
 System.out.println(c3.getArea());  
 System.out.println(c3.getColor());  
 System.out.println(c3.getHeight());  
 System.out.println("------------");  
  
 System.out.println(c4.getRadius());  
 System.out.println(c4.getArea());  
 System.out.println(c4.getColor());  
 System.out.println("------------");  
  
 System.out.println(c5.getRadius());  
 System.out.println("------------");  
  
 System.out.println(c6.getHeight());  
 System.out.println(c6.getRadius());  
 System.out.println(c6.getColor());  
 System.out.println("------------");  
  
 }  
}

**Question 2)**

d) no. because Employee is abstract class. Abstract classes cannot be instantiated directly.

e) name, age and hourRate these are works fine with public ,but the other hand I can’t make them private , after that I can’t access those instances in other classes .

bcz private instances are only allowed in the same class.

**Employee.java**

public abstract class Employee {  
 public String name;  
 public double age;  
 public double hourRate;  
  
 public abstract double salary(double hours); //calculate salary based on hourRate  
 public abstract double totalSalary(double hours,double days);  
  
  
 public String toString() {  
 return "name = " + name + "age = " + age + "hourRate = " + hourRate;  
 }  
  
}

**Clerk.java**

public class Clerk extends Employee {  
  
 Clerk() {  
 }  
  
 Clerk(String name, double age, double hourRate) {  
 this.name = name;  
 this.age = age;  
 this.hourRate = hourRate;  
 }  
  
 public double salary(double hours) {  
 return hourRate \* hours;  
 }  
  
 public double totalSalary(double hours,double days) {  
 return hours \* days \* hourRate;  
 }  
}

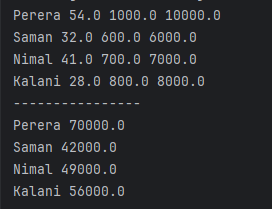
**Manager.java**

public class Manager extends Employee {  
  
 Manager() {  
 }  
  
 Manager(String name, double age, double hourRate) {  
 this.name = name;  
 this.age = age;  
 this.hourRate = hourRate;  
 }  
  
 public double salary(double hours) {  
 return hours \* hourRate;  
 }  
  
 public double totalSalary(double hours,double days) {  
 return hours \* days \* hourRate;  
 }  
}

**HRDivision.java**

public class HRDivision {  
  
 public static void main(String[] args) {  
 Manager manager1 = new Manager("Perera", 54, 1000);  
 Clerk clk1 = new Clerk("Saman", 32, 600);  
 Clerk clk2 = new Clerk("Nimal", 41, 700);  
 Clerk clk3 = new Clerk("Kalani", 28, 800);  
  
 System.*out*.println(manager1.name + " " + manager1.age + " " + manager1.hourRate + " " + manager1.salary(10));  
 System.*out*.println(clk1.name + " " + clk1.age + " " + clk1.hourRate + " " + clk1.salary(10));  
 System.*out*.println(clk2.name + " " + clk2.age + " " + clk2.hourRate + " " + clk2.salary(10));  
 System.*out*.println(clk3.name + " " + clk3.age + " " + clk3.hourRate + " " + clk3.salary(10));  
  
 System.*out*.println("----------------");  
  
 System.*out*.println(manager1.name+ " " + manager1.totalSalary(10, 7));  
 System.*out*.println(clk1.name+ " " + clk1.totalSalary(10, 7));  
 System.*out*.println(clk2.name+ " " + clk2.totalSalary(10, 7));  
 System.*out*.println(clk3.name+ " " + clk3.totalSalary(10, 7));  
  
 }  
}

**Output**

****

**Question 3)**

**a)**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | | Variable Type | | | | |
| Tossable | Ball | Rock | Baseball | Football |
| Object  Type | Tossable | - | - | - | - | - |
| Ball | true | true | x | true | true |
| Rock | true | x | true | x | x |
| Baseball | true | true | x | true | x |
| Football | true | true | x | x | true |

i) Ball ball = new Football("spalding"); **E**

ii) Ball ball = new Football("Spalding"); Baseball baseball (Baseball)ball; **Give Error**

iii) Object obj = new Baseball("spalding"); **E**

iv) Object obj = new Baseball("spalding"); Tossable tossable = obj; **Give Error**

v) Tossable tossable = new Baseball("spalding"); Object objtossable; **C**

vi) Rock rock new Baseball("spalding"); **Give Error**

vii) Tossable tossable = new Rock(); **E**

**Ball.java**

public class Ball implements Tossable {  
 public String brandName;  
  
 Ball(String brand) {  
 this.brandName = brand;  
 }  
  
 public void toss() {  
  
 }  
  
 public void bounce() {  
 System.*out*.println("Ball Will Bounce");  
 }  
  
 public String getBrandName() {  
 return brandName;  
 }  
}

**Rock.java**

public class Rock implements Tossable {  
 public void toss() {  
 System.*out*.println("Rock Will Toss");  
 }  
}

**Tossable.java**

interface Tossable {  
 public void toss();  
}

**Baseball.java**

public class Baseball extends Ball {  
  
 Baseball(String brand) {  
 super(brand);  
 }  
  
 @Override  
 public void toss() {  
 super.toss();  
 }  
  
 @Override  
 public void bounce() {  
 System.*out*.println("Baseball Will Bounce");  
 }  
  
  
}

**Football.java**

public class Football extends Ball {  
  
 Football(String brand) {  
 super(brand);  
 }  
  
 @Override  
 public void toss() {  
 super.toss();  
 }  
  
 @Override  
 public void bounce() {  
 System.*out*.println("Football Will Bounce");  
 }  
}

**testBallToss.java**

public class testBallToss {  
 public static void main(String[] args) {  
  
 Ball ball = new Football("spalding");  
 Object obj = new Baseball("spalding");  
 Tossable tossable = new Rock();  
  
 }  
}

Question 4)