

1.

Create a class named Billing that includes three overloaded computeBill() methods for a photo book store.

- When computeBill() receives a single parameter, it represents the price of one photo book ordered. Add 8% tax, and return the total due.
- When computeBill() receives two parameters, they represent the price of a photo book and the quantity ordered. Multiply the two values, add 8% tax, and return the total due.
- When computeBill() receives three parameters, they represent the price of a photo book, the quantity ordered, and a coupon value. Multiply the quantity and price, reduce the result by the coupon value, and then add 8% tax and return the total due.

Write a main() method that tests all three overloaded methods. Save the application as Billing.java.

```
public class Billing{

    static double computeBill(double a){

        return a*1.08;

    }

    static double computeBill(double a,double b){

        return(a*b)*1.08;

    }

    static double computeBill(double a, double b,double c){

        double n = (a*b)-c;

        return n*1.08;

    }

    public static void main(String[] args) {

        double k = computeBill(20);

        double l = computeBill(40,80);

        double m = computeBill(50,60,100);

        System.out.println("Price of one photo book after adding 8% tax is " + k + "Rs.");

        System.out.println("Price of photo book after computing quality and adding 8% tax is " + l + "Rs.");

    }

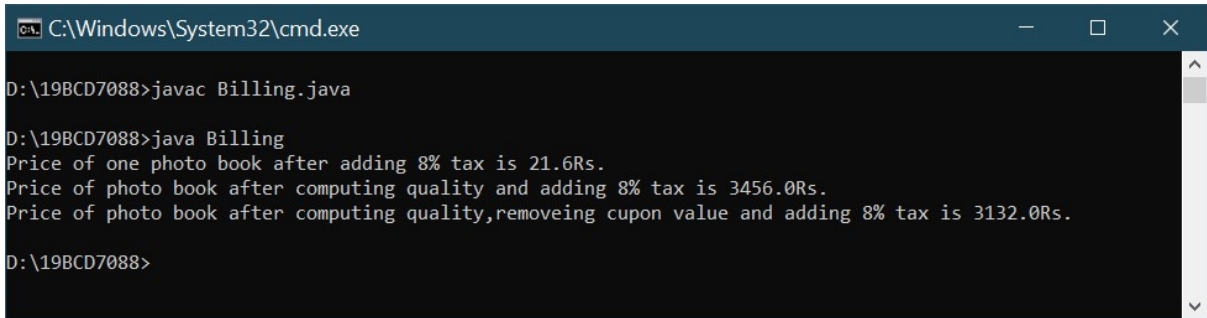
}
```

```

        System.out.println("Price of photo book after computing quality,removeing cupon
value and adding 8% tax is " + m + "Rs.");
    }

}

```



```

C:\Windows\System32\cmd.exe
D:\19BCD7088>javac Billing.java
D:\19BCD7088>java Billing
Price of one photo book after adding 8% tax is 21.6Rs.
Price of photo book after computing quality and adding 8% tax is 3456.0Rs.
Price of photo book after computing quality,removeing cupon value and adding 8% tax is 3132.0Rs.
D:\19BCD7088>

```

2.

a.

Create a class named BloodData that includes fields that hold a blood type (the four blood types are O, A, B, and AB) and an Rh factor (the factors are 1 and –). Create a default constructor that sets the fields to O and 1, and an overloaded constructor that requires values for both fields. Include get and set methods for each field. Save this file as BloodData.java. Create an application named TestBloodData that demonstrates each method works correctly. Save the application as TestBloodData.java.

```

class BloodData{
    String Blood,Rh;
    BloodData(){
        this.Blood = "O";
        this.Rh = "+";
    }
    BloodData(String Blood,String Rh){
        this.Blood=Blood;
        this.Rh=Rh;
    }
    void setBlood(String Blood){
        this.Blood=Blood;
    }
}

```

```

    }

    void setRh(String Rh){
        this.Rh=Rh;
    }

    void getBlood(){
        System.out.println("Blood type is " + this.Blood);
    }

    void getRh(){
        System.out.println("Rh factor of Blood " + this.Blood + " is " + this.Rh);
    }
}

```

```

public class TestBloodData{
    public static void main(String[] args) {
        BloodData b1 = new BloodData();
        BloodData b2 = new BloodData("AB", "-");
        b1.getBlood();
        b1.getRh();
        b2.getBlood();
        b2.getRh();
        b1.setBlood("A");
        b1.setRh("-");
        System.out.println("After using set methods");
        b1.getBlood();
        b1.getRh();
    }
}

```

```
C:\Windows\System32\cmd.exe
D:\19BCD7088>java TestBloodData
Blood type is O
Rh factor of Blood O is +
Blood type is AB
Rh factor of Blood AB is -
After using set methods
Blood type is A
Rh factor of Blood A is -
D:\19BCD7088>
```

2.

b.

Create a class named Patient that includes an ID number, age, and BloodData. Provide a default constructor that sets the ID number to 0, the age to 0, and the BloodData values to O and 1. Create an overloaded constructor that provides values for each field. Also provide get methods for each field. Save the file as Patient.java. Create an application that demonstrates that each method works correctly, and save it as TestPatient.java.

```
class Patient{
    int Id,age;
    String Blood,Rh;
    Patient(){
        this.Id = 0;
        this.age =0;
        this.Rh="+";
        this.Blood = "O";
    }
    Patient(int Id,int age,String Blood,String Rh){
        this.Id = Id;
        this.age=age;
        this.Rh=Rh;
        this.Blood = Blood;
    }
    void getId(){
        System.out.println("Patient Id number is " + this.Id);
    }
}
```

```

void getage(){
    System.out.println("Age of thr Patient is " + this.age + " years old");
}
void getBlood(){
    System.out.println("Blood type is " + this.Blood);
}
void getRh(){
    System.out.println("Rh factor of Blood is " + this.Rh);
}
}
.....
public class TestPatient{
    public static void main(String[] args) {
        Patient p1 = new Patient();
        Patient p2 = new Patient(152685,48,"B","-");
        System.out.println("Details of Default Patient");
        p1.getId();
        p1.getage();
        p1.getBlood();
        p1.getRh();
        System.out.println("\n Details of Patient");
        p2.getId();
        p2.getage();
        p2.getBlood();
        p2.getRh();
    }
}

```

```
C:\Windows\System32\cmd.exe

D:\19BCD7088>javac TestPatient.java

D:\19BCD7088>java TestPatient
Details of Default Patient
Patient Id number is 0
Age of thr Patient is 0 years old
Blood type is 0
Rh factor of Blood is +

    Details of Patient
Patient Id number is 152685
Age of thr Patient is 48 years old
Blood type is B
Rh factor of Blood is -

D:\19BCD7088>
```

3.

a.

Create a class named Circle with fields named radius, diameter, and area. Include a constructor that sets the radius to 1 and calculates the other two values. Also include methods named setRadius() and getRadius(). The setRadius() method not only sets the radius, but it also calculates the other two values. (The diameter of a circle is twice the radius, and the area of a circle is pi multiplied by the square of the radius. Use the Math class PI constant for this calculation.) Save the class as Circle.java.

```
import java.lang.Math;
```

```
class Circle{

    double r;

    double d;

    double area;

    Circle(){

        this.r = 1;

        this.d = 2*1;

        this.area = Math.PI*1*1;

    }

    void setRadius(double r){

        this.r = r;

        this.d = 2*r;

        this.area = Math.PI*r*r;

    }

}
```

```

    }

    void getRadius(){
        System.out.println(this.r + "radius circle has " + this.d + " diameter and " + this.area
+ " area.");
    }
}

```

3.

b.

Create a class named TestCircle whose main() method declares several Circle objects. Using the setRadius() method, assign one Circle a small radius value, and assign another a larger radius value. Do not assign a value to the radius of the third circle; instead, retain the value assigned at construction. Display all the values for all the Circle objects. Save the application as TestCircle.java.

```

public class TestCircle{

    public static void main(String[] args) {

        Circle c1 = new Circle();

        Circle c2 = new Circle();

        Circle c3 = new Circle();

        c1.setRadius(10);

        c2.setRadius(50);

        System.out.println("Values of small radius circle.");

        c1.getRadius();

        System.out.println("\nValues of larger radius circle.");

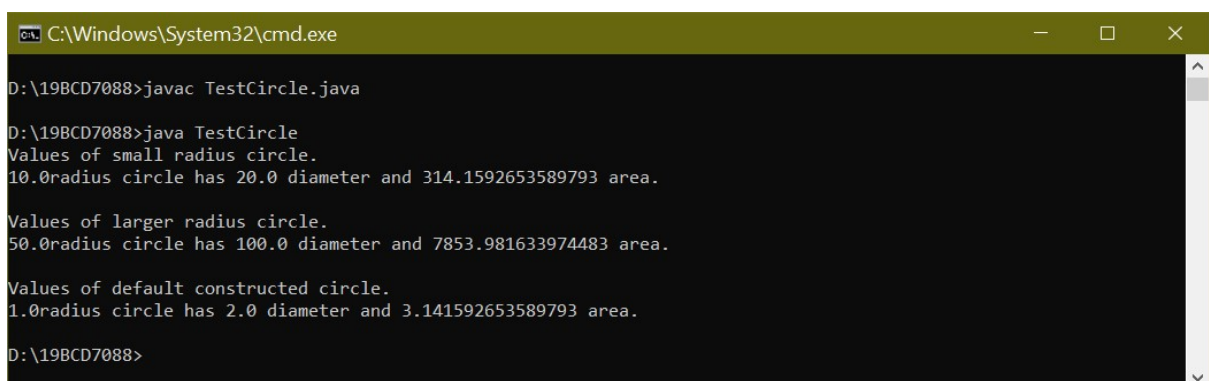
        c2.getRadius();

        System.out.println("\nValues of default constructed circle.");

        c3.getRadius();

    }
}

```



```

C:\Windows\System32\cmd.exe
D:\19BCD7088>javac TestCircle.java

D:\19BCD7088>java TestCircle
Values of small radius circle.
10.0radius circle has 20.0 diameter and 314.1592653589793 area.

Values of larger radius circle.
50.0radius circle has 100.0 diameter and 7853.981633974483 area.

Values of default constructed circle.
1.0radius circle has 2.0 diameter and 3.141592653589793 area.

D:\19BCD7088>

```

4.

Write a Java application that uses the Math class to determine the answers for each of the following:(Use java.lang.Math class)

- a. The square root of 37
- b. The sine and cosine of 300
- c. The value of the floor, ceiling, and round of 22.8
- d. The larger and the smaller of the character 'D' and the integer 71
- e. A random number between 0 and 20 (Hint: The random() method returns a value between 0 and 1; you want a number that is 20 times larger.)

Save the application as MathTest.java.

```
import java.lang.Math;

public class MathTest{

    public static void main(String[] args) {

        double a = 37;

        System.out.println("Square root of " + a + " is " + Math.sqrt(a));

        double b = 300;

        double n=Math.toRadians(b);

        System.out.println("Sine of " + b + " is " + Math.sin(n));

        System.out.println("Cosine of " + b + " is " + Math.cos(n));

        double c = 22.8;

        System.out.println("Floor value of " + c + " is " + Math.floor(c));

        System.out.println("ceiling value of " + c + " is " + Math.ceil(c));

        System.out.println("Round value of " + c + " is " + Math.round(c));

        if ((char)(Math.max('D',71)) == 'D'){

            System.out.println("Largest of Character 'D' and integer 71 is 'D'");

        }

        else {

            System.out.println("Largest of Character 'D' and integer 71 is 71.");

        }

    }

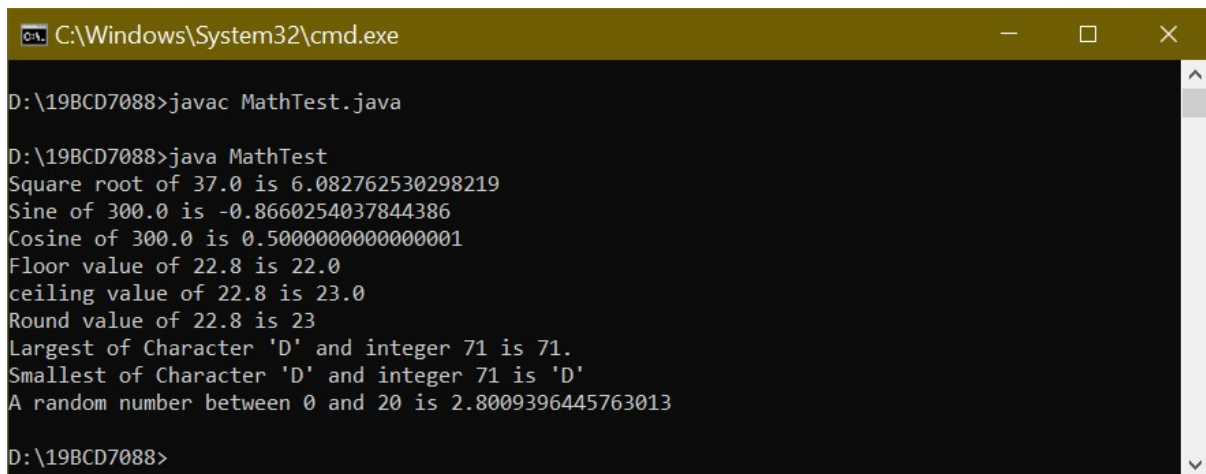
}
```



```

        if ((char)(Math.min('D',71)) == 'D'){
            System.out.println("Smallest of Character 'D' and integer 71 is 'D'");
        }
        else {
            System.out.println("Smallest of Character 'D' and integer 71 is 71.");
        }
        System.out.println("A random number between 0 and 20 is " + 20*Math.random());
    }
}

```



The screenshot shows a Windows command prompt window titled "C:\Windows\System32\cmd.exe". The user is in the directory "D:\19BCD7088". They run the command "javac MathTest.java" to compile the file. Then they run "java MathTest" to execute it. The output shows various mathematical calculations and comparisons, including the square root of 37.0, sine and cosine of 300.0, floor and ceiling values of 22.8, round value of 22.8, and the largest/smallest of character 'D' and integer 71. Finally, it displays a random number between 0 and 20.

```

D:\19BCD7088>javac MathTest.java

D:\19BCD7088>java MathTest
Square root of 37.0 is 6.082762530298219
Sine of 300.0 is -0.8660254037844386
Cosine of 300.0 is 0.5000000000000001
Floor value of 22.8 is 22.0
ceiling value of 22.8 is 23.0
Round value of 22.8 is 23
Largest of Character 'D' and integer 71 is 71.
Smallest of Character 'D' and integer 71 is 'D'
A random number between 0 and 20 is 2.8009396445763013

D:\19BCD7088>

```

5.

a.

Write a program that declares two LocalDate objects and assign values that represent January 31 and December 31 in the current year. Display output that demonstrates the dates displayed when one, two, and three months are added to each of the objects. Save the application as TestMonthHandling.java.

```
import java.time.*;
```

```

public class TestMonthHandling{
    public static void main(String[] args) {
        LocalDate mon1;
        LocalDate mon2;
        LocalDate temp1;
    }
}

```

```

        LocalDate temp2;

        int mo1 = 1;

        int day1 = 31;

        int year = 2020;

        mon1 = LocalDate.of(year, mo1, day1);

        int mo2 = 12;

        int day2 = 31;

        mon2 = LocalDate.of(year, mo2, day2);

        System.out.println("Present dates are " + mon1.getYear() + " " + mon1.getMonth()+
        " " + mon1.getDayOfMonth() + " and " + mon2.getYear() + " " + mon2.getMonth()+" " +
        mon2.getDayOfMonth());

        temp1 = mon1.plusMonths(1);

        temp2 = mon2.plusMonths(1);

        System.out.println("Dates after adding one month to each " + temp1.getYear() + " " +
        temp1.getMonth() + " " + temp1.getDayOfMonth() + " and " + temp2.getYear() + " " +
        temp2.getMonth()+" " + temp2.getDayOfMonth() );

        temp1 = mon1.plusMonths(2);

        temp2 = mon2.plusMonths(2);

        System.out.println("Dates after adding two month to each " + temp1.getYear() + " "+
        temp1.getMonth() + " " + temp1.getDayOfMonth() + " and " + temp2.getYear() + " " +
        temp2.getMonth()+" " + temp2.getDayOfMonth() );

        temp1 = mon1.plusMonths(3);

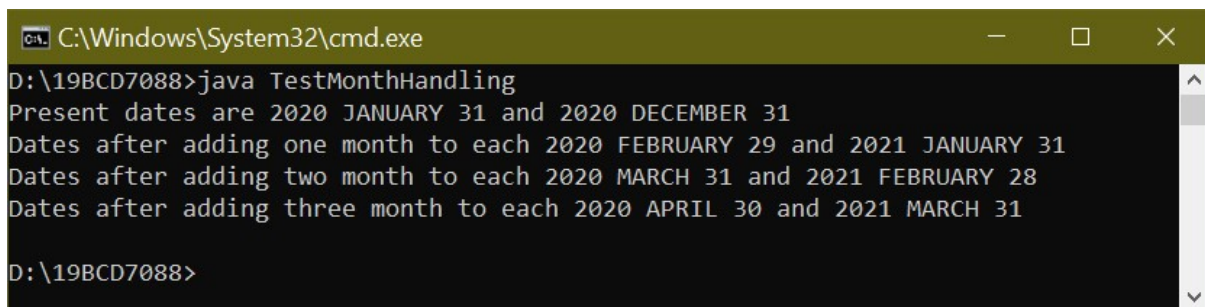
        temp2 = mon2.plusMonths(3);

        System.out.println("Dates after adding three month to each " + temp1.getYear() + "
        " + temp1.getMonth() + " " + temp1.getDayOfMonth() + " and " + temp2.getYear() + " " +
        temp2.getMonth()+" " + temp2.getDayOfMonth() );

    }

}

```



The screenshot shows a Windows command prompt window with the title bar 'C:\Windows\System32\cmd.exe'. The command prompt is open at the directory 'D:\19BCD7088'. The user has entered the command 'java TestMonthHandling'. The output of the program is displayed as follows:

```

D:\19BCD7088>java TestMonthHandling
Present dates are 2020 JANUARY 31 and 2020 DECEMBER 31
Dates after adding one month to each 2020 FEBRUARY 29 and 2021 JANUARY 31
Dates after adding two month to each 2020 MARCH 31 and 2021 FEBRUARY 28
Dates after adding three month to each 2020 APRIL 30 and 2021 MARCH 31

D:\19BCD7088>

```

5.

b.

Write an application that computes and displays the day on which you become (or became) 10,000 days old. Save the application as `TenThousandDaysOld.java`.

```
import java.time.*;
```

```
public class TenThousandDaysOld{
```

```
    public static void main(String[] args) {
```

```
        LocalDate dob;
```

```
        int mo = 7;
```

```
        int day = 22;
```

```
        int year = 2001;
```

```
        dob = LocalDate.of(year, mo, day);
```

```
        System.out.println("Born on " + dob.getYear() + " " + dob.getMonth() + " " +  
dob.getDayOfMonth());
```

```
        dob=dob.plusDays(10000);
```

```
        System.out.println("Date on which I become 10,000 days old is " + dob.getYear() + "  
" + dob.getMonth()+ " " + dob.getDayOfMonth());
```

```
    }
```

```
}
```



```
C:\Windows\System32\cmd.exe

D:\19BCD7088>javac TenThousandDaysOld.java

D:\19BCD7088>java TenThousandDaysOld
Born on 2001 JULY 22
Date on which I become 10,000 days old is 2028 DECEMBER 7

D:\19BCD7088>
```

5.

c.

The `LocalDate` class includes an instance method named `lengthOfMonth()` that returns the number of days in the month. Write an application that uses methods in the `LocalDate` class to calculate how

many days are left until the first day of next month. Display the result, including the name of the next month. Save the file as DaysTilNextMonth.java.

```
import java.time.*;

public class DaysTilNextMonth{

    public static void main(String[] args) {

        LocalDate mon;

        LocalDate temp;

        int mo = 6;

        int day = 12;

        int year = 2010;

        mon = LocalDate.of(year, mo, day);

        int remain = mon.lengthOfMonth() - day;

        System.out.println("Todays date " + mon.getYear() + " " + mon.getMonth()+ " " +
mon.getDayOfMonth());

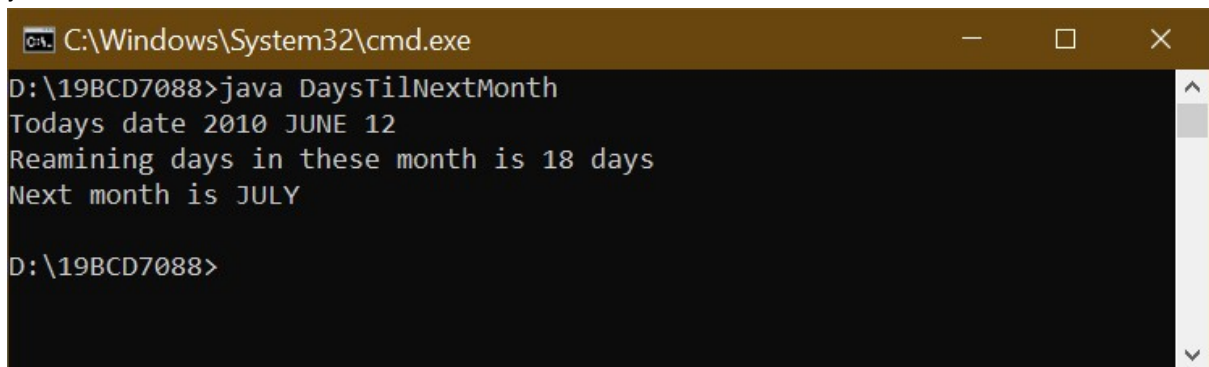
        System.out.println("Reamining days in these month is " + remain + " days");

        mon = mon.plusMonths(1);

        System.out.println("Next month is " + mon.getMonth());

    }

}
```

A screenshot of a Windows command prompt window. The title bar shows the path 'C:\Windows\System32\cmd.exe'. The command prompt shows the following text: 'D:\19BCD7088>java DaysTilNextMonth', followed by the output: 'Todays date 2010 JUNE 12', 'Reamining days in these month is 18 days', and 'Next month is JULY'. The prompt then shows 'D:\19BCD7088>' again. There is a scrollbar on the right side of the window.

```
C:\Windows\System32\cmd.exe
D:\19BCD7088>java DaysTilNextMonth
Todays date 2010 JUNE 12
Reamining days in these month is 18 days
Next month is JULY
D:\19BCD7088>
```