1.

```
 * Devante Wilson
 * November 2, 2015
 *
 * Class defines static methods
 * to be utilized by other classes
 */

public class Geometry
{
    public static double cubeVolume(double h)
    {
        double cuVolume = Math.pow(h, 3);
        return cuVolume;
    }

    public static double cubeSurface(double h)
    {
        double cuSurface = 6 * Math.pow(h, 2);
        return cuSurface;
    }

    public static double sphereVolume(double r)
    {
        double sVolume = (4/3.0) * Math.PI * Math.pow(r, 3);
        return sVolume;
    }

    public static double sphereSurface(double r)
    {
        double sSurface = 4 * Math.PI * Math.pow(r, 2);
        return sSurface;
    }

    public static double cylinderVolume(double r, double h)
    {
        double cyVolume = Math.PI * Math.pow(r, 2) * h;
        return cyVolume;
    }

    public static double cylinderSurface(double r, double h)
    {
        double cySurface = (2 * Math.PI * r) * (h + r);
        return cySurface;
    }

    public static double coneVolume(double r, double h)
    {
        double coVolume = Math.PI * Math.pow(r, 2) * (h/3.0);
        return coVolume;
    }

    public static double coneSurface(double r, double h)
    {
        double coSurface = (Math.PI * r) * (r + Math.sqrt(Math.pow(h,2) + Math.pow(r, 2)));
        return coSurface;
    }
```

1.1

1.2

1.3

1.4

1.5

1.6

1.7

1.8

```java
// import packages
import java.util.Scanner;

public class GeometryTester
{
    public static void main(String[] args)
    {
        // define variables/objects
        Geometry geo = new Geometry();
        Scanner scan = new Scanner(System.in);
        double r = 0, h = 0;

        // prompt user for values
        System.out.print("Please enter a radius: ");
        r = scan.nextDouble();
        System.out.print("Please enter a height: ");
        h = scan.nextDouble();

        // call methods and print results
        System.out.printf("Cube volume: %.2f" +
                "\nCube surface area: %.2f" +
                "\nSphere volume: %.2f" +
                "\nSphere surface area: %.2f" +
                "\nCylinder volume: %.2f" +
                "\nCylinder surface area: %.2f" +
                "\nCone volume: %.2f" +
                "\nCone surface: %.2f",
                geo.cubeVolume(h),
                geo.cubeSurface(h),
                geo.sphereVolume(r),
                geo.sphereSurface(r),
                geo.cylinderVolume(r,h),
                geo.cylinderSurface(r,h),
                geo.coneVolume(r, h),
                geo.coneSurface(r,h));
    }
}
```

Console

<terminated> GeometryTester [Java Application] C:\Software\IBM Eclipse\ecl

```
Please enter a radius: 4
Please enter a height: 5
Cube volume: 125.00
Cube surface area: 150.00
Sphere volume: 268.08
Sphere surface area: 201.06
Cylinder volume: 251.33
Cylinder surface area: 226.19
Cone volume: 83.78
Cone surface: 130.73
```

2.

```java
/**
 * Devante Wilson
 * November 6, 2015
 *
 * Abstract class to solve question 2.
 */

public abstract class Geo
{
    public abstract double volume(double x);
    public abstract double volume(double r, double h);
    public abstract double surfaceArea(double x);
    public abstract double surfaceArea(double r, double h);
}
```

2.1

```java
/**
 * Devante Wilson
 * November 3, 2015
 *
 * cube subclass - question 2.
 */

public class Cube extends Geo
{
    public double volume(double h)
    {
        double cuVolume = Math.pow(h, 3);
        return cuVolume;
    }

    public double surfaceArea(double h)
    {
        double cuSurface = 6 * Math.pow(h, 2);
        return cuSurface;
    }

    // must implement - but not used in this class
    public double volume(double r, double h)
    {
        return 0;
    }

    // must implement - but not used in this class
    public double surfaceArea(double r, double h)
    {
        return 0;
    }
}
```

2.2

```java
/**
 * Devante Wilson
 * November 3, 2015
 *
 * sphere subclass - question 2.
 */

public class Sphere extends Geo
{
    public double volume(double r)
    {
        double sVolume = (4/3.0) * Math.PI * Math.pow(r, 3);
        return sVolume;
    }

    public double surfaceArea(double r)
    {
        double sSurface = 4 * Math.PI * Math.pow(r, 2);
        return sSurface;
    }

    // must implement - but not used in this class
    public double volume(double r, double h)
    {
        return 0;
    }

    // must implement - but not used in this class
    public double surfaceArea(double r, double h)
    {
        return 0;
    }
}
```

2.3

```java
/**
 * Devante Wilson
 * November 3, 2015
 *
 * cylinder subclass - question 2.
 */

public class Cylinder extends Geo
{
    public double volume(double r, double h)
    {
        double cyVolume = Math.PI * Math.pow(r, 2) * h;
        return cyVolume;
    }

    public double surfaceArea(double r, double h)
    {
        double cySurface = (2 * Math.PI * r) * (h + r);
        return cySurface;
    }

    // must implement - but not used in this class
    public double volume(double x)
    {
        return 0;
    }

    // must implement - but not used in this class
    public double surfaceArea(double x)
    {
        return 0;
    }
}
```

2.4

```java
/**
 * Devante Wilson
 * November 3, 2015
 *
 * cone subclass - question 2.
 */

public class Cone extends Geo
{
    public double volume(double r, double h)
    {
        double coVolume = Math.PI * Math.pow(r, 2) * (h/3.0);
        return coVolume;
    }

    public double surfaceArea(double r, double h)
    {
        double coSurface = (Math.PI * r) * (r + Math.sqrt(Math.pow(h,2) + Math.pow(r, 2)));
        return coSurface;
    }

    // must implement - but not used in this class
    public double volume(double x)
    {
        return 0;
    }

    // must implement - but not used in this class
    public double surfaceArea(double x)
    {
        return 0;
    }
}
```

```
13      public static void main(String[] args)
14      {
15          // define variables/objects
16          Geometry geo = new Geometry();
17          Cube cu = new Cube();
18          Sphere s = new Sphere();
19          Cylinder cy = new Cylinder();
20          Cone co = new Cone();
21          Scanner scan = new Scanner(System.in);
22          double r = 0, h = 0;
23
24          // prompt user for values
25          System.out.print("Please enter a radius: ");
26          r = scan.nextDouble();
27          System.out.print("Please enter a height: ");
28          h = scan.nextDouble();
29
30          // close Scanner object
31          scan.close();
32
33          // call methods and print results - from Geometry class
34          System.out.printf("Cube volume: %.2f" +
35                  "\nCube surface area: %.2f" +
36                  "\nSphere volume: %.2f" +
37                  "\nSphere surface area: %.2f" +
38                  "\nCylinder volume: %.2f" +
39                  "\nCylinder surface area: %.2f" +
40                  "\nCone volume: %.2f" +
41                  "\nCone surface: %.2f",
42                  geo.cubeVolume(h),
43                  geo.cubeSurface(h),
44                  geo.sphereVolume(r),
45                  geo.sphereSurface(r),
46                  geo.cylinderVolume(r,h),
47                  geo.cylinderSurface(r,h),
48                  geo.coneVolume(r, h),
49                  geo.coneSurface(r,h));
50
51          // call methods and print results - from Geo class
52          System.out.printf("\n\nCube volume: %.2f" +
53                  "\nCube surface area: %.2f" +
54                  "\nSphere volume: %.2f" +
55                  "\nSphere surface area: %.2f" +
56                  "\nCylinder volume: %.2f" +
57                  "\nCylinder surface area: %.2f" +
58                  "\nCone volume: %.2f" +
59                  "\nCone surface: %.2f",
60                  cu.volume(h),
61                  cu.surfaceArea(h),
62                  s.volume(r),
63                  s.surfaceArea(r),
64                  cy.volume(r,h),
65                  cy.surfaceArea(r,h),
66                  co.volume(r, h),
67                  co.surfaceArea(r,h));
```

OUTPUT

```
Please enter a radius: 4
Please enter a height: 5
Cube volume: 125.00
Cube surface area: 150.00
Sphere volume: 268.08
Sphere surface area: 201.06
Cylinder volume: 251.33
Cylinder surface area: 226.19
Cone volume: 83.78
Cone surface: 130.73

Cube volume: 125.00
Cube surface area: 150.00
Sphere volume: 268.08
Sphere surface area: 201.06
Cylinder volume: 251.33
Cylinder surface area: 226.19
Cone volume: 83.78
Cone surface: 130.73
```

3. & 4.

```java
/**
 * Devante Wilson
 * November 3, 2015
 *
 * Class implements the java.awt.Rectangle class
 * and supplies methods to compute
 * the area and perimeter of a rectangle
 */

// import rectangle class
import java.awt.Rectangle;

public class BetterRectangle extends Rectangle
{
    // compute rectangle perimeter
    public double getPerimeter()
    {
        double perimeter = 2 * (width + height);
        return perimeter;
    }

    // compute rectangle area
    public double getArea()
    {
        double area = width * height;
        return area;
    }

    // constructor
    public BetterRectangle(int x, int y, int width, int height)
    {
        // create Rectangle object (invoke constructor of superclass)
        Rectangle rect = new Rectangle();

        // set the location of the rectangle
        setLocation(x, y);
        // set the size of the rectangle
        setSize(width, height);
    }

    // default no parameter constructor
    public BetterRectangle()
    {

    }
}
```

```java
 * Devante Wilson
 * November 3, 2015
 *
 * Class tests the BetterRectangle class
 */

// import package(s)
import java.util.Scanner;

public class BetterRectangleTester
{
    public static void main(String[] args)
    {
        // define variables/objects
        Scanner scan = new Scanner(System.in);
        int width = 0, height = 0, x = 0, y = 0;
        BetterRectangle rect;

        // prompt user for rectangle information
        System.out.print("Enter rectangle width: ");
        width = scan.nextInt();
        System.out.print("Enter rectangle height: ");
        height = scan.nextInt();
        System.out.print("Enter rectangle position x: ");
        x = scan.nextInt();
        System.out.print("Enter rectangle position y ");
        y = scan.nextInt();
        rect = new BetterRectangle(x,y,width,height);

        // call methods and print results
        System.out.printf("Rectangle perimeter: %.2f", rect.getPerimeter());
        System.out.printf("\nRectangle area: %.2f", rect.getArea());
        rect.setSize(width, height);
        rect.setLocation(x,y);

        System.out.println("\n" + rect.getSize());
        System.out.println(rect.getLocation());
    }
}
```

Console

```
<terminated> BetterRectangleTester [Java Application] C:\Software\IBM Eclipse\eclipseDevelopmentPackage\
Enter rectangle width: 2
Enter rectangle height: 4
Enter rectangle position x: 1
Enter rectangle position y 2
Rectangle perimeter: 12.00
Rectangle area: 8.00
java.awt.Dimension[width=2,height=4]
java.awt.Point[x=1,y=2]
```

5.

```java
/**
 * Devante Wilson
 * November 3, 2015
 *
 * Program models a circuit.
 * This class is a superclass.
 */

public class Circuit
{
    public double getResistance()
    {
        return 0;
    }
}
```

```java
/**
 * Devante Wilson
 * November 3, 2015
 *
 * Program models a circuit.
 * This subclass represents a single resistor.
 */

public class Resistor extends Circuit
{
    // class variable
    private double resistance;

    // constructor/mutator method
    public Resistor(double r)
    {
        resistance = r;
    }

    // accessor method
    public double getResistance()
    {
        return resistance;
    }
}
```

```java
 * Devante Wilson
 * November 3, 2015
 *
 * Program models a circuit.
 * This subclass models a set of circuits in parallel.
 */

// import package(s)
import java.util.ArrayList;

public class Parallel extends Circuit
{
    // declare new ArrayList
    private ArrayList<Circuit> parallel;

    // constructor
    public Parallel()
    {
        parallel = new ArrayList<Circuit>();
    }

    // method to add resistors
    public void add(Circuit c)
    {
        parallel.add(c);
    }

    // compute combined resistance
    public double getResistance()
    {
        // define variables/objects
        double rTotal = 0, rVal;
        Circuit resist;

        for (int a = 0; a < parallel.size(); a++)
        {
            resist = parallel.get(a);   // retrieve arrayList object
            rVal = resist.getResistance();  // call mutator method from Resistor class
            rTotal += 1.0/rVal;
        }

        if (rTotal != 0) // prevent divide by 0
            rTotal = 1.0/rTotal;

        return rTotal;
    }
}
```

```java
/**
 * Devante Wilson
 * November 3, 2015
 *
 * Program models a circuit.
 * This subclass models a series of circuits.
 */

// import package(s)
import java.util.ArrayList;

public class Serial extends Circuit
{
    // define new ArrayList
    ArrayList<Circuit> serial;

    // constructor
    public Serial()
    {
        serial = new ArrayList<Circuit>();
    }

    // method to add resistors
    public void add(Circuit c)
    {
        serial.add(c);
    }

    // compute combined resistance
    public double getResistance()
    {
        double rTotal = 0, rVal;
        Circuit resist;

        for (int b = 0; b < serial.size(); b++)
        {
            resist = serial.get(b); // retrieve arrayList object
            rVal = resist.getResistance();  // call mutator method from Resistor class
            rTotal += rVal;
        }

        return rTotal;
    }
}
```

```java
/**
 * Devante Wilson
 * November 6, 2015
 *
 * Class is a tester for Circuit program
 */

public class CircuitTester
{
    public static void main(String[] args)
    {
        // create objects
        Parallel p = new Parallel();
        Serial s = new Serial();

        p.add(new Resistor(100));
        s.add(new Resistor(100));
        s.add(new Resistor(200));
        p.add(s);

        // print result
        System.out.println("Combined resistance: " + p.getResistance());
        System.out.println("Expected: 75.0");
    }
}
```

Console

<terminated> CircuitTester [Java Application] C:\Software\IBM Eclipse\eclipseDevelopmentPackage\ibm_sdk80\bin\javaw.exe (Nov 14, 2015, 7:08

```
Combined resistance: 75.0
Expected: 75.0
```

6.

```java
/**
 * Devante Wilson
 * November 5, 2015
 *
 * Program computes the alternating sum of an array's elements.
 * (Operator switches from - to + between each element)
 */

// import package
import java.util.Arrays;

public class Q6
{
    static int alternatingSum(int[] arr)
    {
        int sum = 0;

        // calculute alternating sum
        for (int a = 0; a < arr.length; a+=2)
            sum += arr[a];

        for (int b = 1; b < arr.length; b+=2)
            sum -= arr[b];

        return sum;
    }

    public static void main(String[] args)
    {
        // define array
        int[] sumArr = {1,4,9,16,9,7,4,9,11};

        // call method and print result
        System.out.print("Array: " + Arrays.toString(sumArr) +
                "\nAlternating sum: " + alternatingSum(sumArr) +
                "\nExpected: -2");
    }
}
```

Console

<terminated> Q6 [Java Application] C:\Software\IBM Eclipse\eclipseDevelopmentPackage\ibm_sdk80\bin\

```
Array: [1, 4, 9, 16, 9, 7, 4, 9, 11]
Alternating sum: -2
Expected: -2
```

7.

```java
/**
 * Devante Wilosn
 * November 5, 2015
 *
 * Program generates a sequence of 20 random values from 0-99
 * prints the sequence, sorts it, and prints the sorted sequence.
 */

// import package(s)
import java.util.Random;
import java.util.Arrays;

public class Q7
{
    public static void main(String[] args)
    {
        // create variables/objects
        Random randomInt = new Random();
        int[] randomArr = new int[20];

        System.out.print("Unsorted: ");

        // generate sequence of random integers
        for (int i = 0; i < randomArr.length; i++)
        {
            // store values in array
            randomArr[i] = randomInt.nextInt(100);

            // print random sequence
            System.out.print(randomArr[i] + " ");
        }

        // sort array
        Arrays.sort(randomArr);

        System.out.print("\nSorted: ");

        for (int j = 0; j < randomArr.length; j++)
        {
            // print sorted array
            System.out.print(randomArr[j] + " ");
        }
    }
}
```

Console

<terminated> Q7 [Java Application] C:\Software\IBM Eclipse\eclipseDevelopmentPackage\ibm_sdk80\bin\ja

```
Unsorted: 45 85 2 45 19 4 29 30 61 67 67 22 33 79 51 90 52 0 26 60
Sorted: 0 2 4 19 22 26 29 30 33 45 45 51 52 60 61 67 67 79 85 90
```

8.

```java
/**
 * Devante Wilson
 * November 6, 2015
 *
 * Class provides the nominal and actual values
 * for the voltage divider's gain.
 */

public class VoltageDivider
{
    // instance variables
    private Resistor2 r1, r2;

    /* constructor:
     * accepts two Resistor objects,
     * nominal resistance and resistor tolerance.
     */
    public VoltageDivider(Resistor2 r1, Resistor2 r2)
    {
        this.r1 = r1;
        this.r2 = r2;
    }

    // retrieve nominal value of voltage divider's gain
    public double getNominalGain()
    {
        return r1.getNominal()/(r1.getNominal() + r2.getNominal());
    }

    // retrieve actual value of voltage divider's gain
    public double getActualGain()
    {
        return r1.getActual()/(r1.getActual() + r2.getActual());
    }

    // default no parameter constructor
    public VoltageDivider()
    {

    }
}
```

```java
// import package(s)
import java.util.concurrent.ThreadLocalRandom;

public class Resistor2
{
    // define instance variables
    private double nominalResistance, tolerance, actualResistance;

    /* constructor:
     * determines actual resistance value randomly
     * given nominal resistance and tolerance (+/- 5%)
     */
    public Resistor2(double nominal, double tolerance)
    {
        // determine actual value randomly
        actualResistance = ThreadLocalRandom.current().nextDouble(
                nominal - (nominal * (tolerance/100.0)),
                nominal * ((tolerance/100.0) + 1));

        nominalResistance = nominal;
        this.tolerance = tolerance;
    }

    // Default no parameter constructor
    public Resistor2()
    {

    }

    // get nominal resistance
    public double getNominal()
    {
        return nominalResistance;
    }

    // get resistance tolerance
    public double getTolerance()
    {
        return tolerance;
    }

    // get actual resistance
    public double getActual()
    {
        return actualResistance;
    }
}
```

```java
/**
 * Devante Wilson
 * November 7, 2015
 *
 * Class tests the VoltageDivider and Resistor classes.
 * Question 8
 */

public class GainTester
{
    public static void main(String[] args)
    {
        final int circuitAmount = 10; // amount of voltage dividers

        // output actual and nominal gains
        for (int i = 1; i <= circuitAmount; i++)
        {
            Resistor2 r1 = new Resistor2(250, 5);
            Resistor2 r2 = new Resistor2(750, 5);
            VoltageDivider divider = new VoltageDivider(r1, r2);

            if (i == 1)
            System.out.print("Nominal Gain (for all Resistors): " +
                    divider.getNominalGain() + " Ohms\n\nActual Gains (Ohms):\n\n");

            System.out.printf(i + ": %.4f", divider.getActualGain());
            System.out.print("\n");
        }
    }
}
```

```
Nominal Gain (for all Resistors): 0.25 Ohms

Actual Gains (Ohms):

1: 0.2585
2: 0.2449
3: 0.2492
4: 0.2546
5: 0.2562
6: 0.2425
7: 0.2408
8: 0.2434
9: 0.2526
10: 0.2577
```