7.15 Exercises

The problems in this section cover the basics, including use of keyboard input, looping and branching constructs, simple arrays, and generation of formatted output.

7.1 It is well known that the formula for converting a temperature in Celsius to Fahrenheit is

$$^oF \;=\; \frac{9}{5} \cdot ^o C + 32. \tag{7.10}$$

Write a Java program that will prompt the user for the temperature in degrees Celsius, and compute and print the equivalent temperature in Fahrenheit.

7.2 This problem will give you practice at writing loops, computing arithmetic operations with the modulo operator, and computing a simple summation of values. If we list all the natural numbers below 10 that are multiples of 3 or 5, we get 3, 5, 6 and 9. The sum of these multiples is 23. Write a java program to find and print the sum of all the multiples of 3 or 5 below 1000.

7.3 2520 is the smallest number that can be divided by each of the numbers from 1 to 10 without any remainder. Write a java program to find and print the smallest number that is evenly divisible by all of the numbers from 1 to 20?

Hint. For a given number n, checking that n%2 == 0, n%3 == 0, n%4 == 0, $\cdots$ n%19 == 0, n%20 == 0 is excessive. Recall that all numbers have a unique prime factorization – for example, the 4 can be represented a the product 2 times 2. Similarly, 6 can be represented as the product of 2 and 3, and so forth. Therefore if a number is divisible by 6 then it is automatically divisible by 2 and 3. You should use this fact to minimise the number of modulo evaluations.

7.4 Figure 7.12 shows the profile of a 400 m running track. Suppose that an athelete runs “x” meters from the inner track. Write a Java program that computes the distance around the track for values of “x” ranging from 0 m to 1 m, and prints the results in a tidy table.

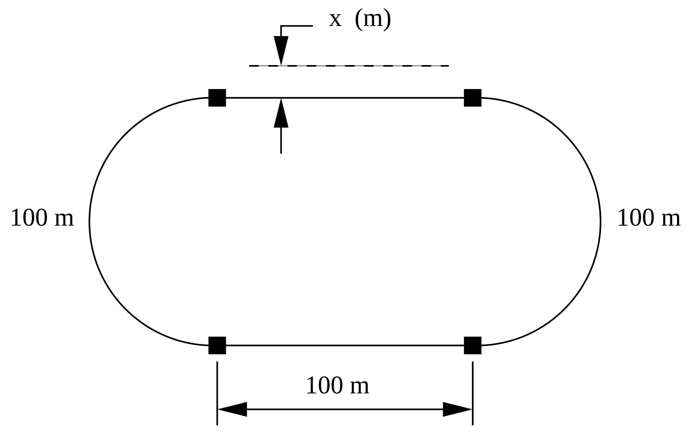

Figure 7.12. Running Track

The output should look something like:

```
 X (m) | Distance (m)
====================
0.0 m        400.0 m
0.2 m        .......
0.4 m        .......
0.6 m        .......
0.8 m        .......
1.0 m        .......
====================
```

Try to format the output so that distances are printed to one decimal place of accuracy and arranged vertically in a nice tidy format.

7.5 Figure 7.13 shows a mass m resting on a frictionless surface. The mass is connected to two walls by springs having stifnesses k_1 and k_2.

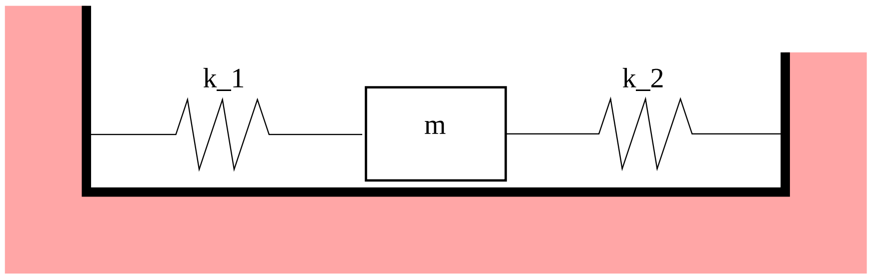

Figure 7.13. Mass-Spring System

The natural period of the mass-spring system is:

$$T = 2\pi\sqrt{\frac{m}{k_1 + k_2}} \tag{7.11}$$

Write a Java program that will prompt a user for m, k_1, and k_2, check that the supplied values are all greater than zero, and then compute and print the natural period of the mass-spring system.

7.6 The area of a triangle having side lengths a, b, and c is

$$Area = \sqrt{s(s-a)(s-b)(s-c)} \tag{7.12}$$

where s = (a + b + c)/2. Write a Java program that will prompt the user for the three side lengths of the triangle, and then compute and print the area of the triangle via equation 7.12.

Note. To ensure that the triangle is well defined, your program should check that all side lengths are greater than zero, and that the sum of the shorter two side lengths exceeds the length of the longest side.

7.7 Figure 7.14 shows a triangle defined by the vertex coordinates (x_1, y_1), (x_2, y_2) and (x_3, y_3). Write a Java program that will:

1. Interactively prompt a user for the x and y coordinates at each of the triangle vertices.
2. Print the triangle vertices in a tidy table.
3. Compute and print the area of the triangle (make sure that it is positive) and its perimeter.

7.8 Figure 7.15 shows the circular cross section of a torus ring having radius `R` and cross section diameter `D`. The volume and surface area of the torus ring is given by:

$$\text{Volume} = 2\pi^2 R \left[\frac{D^2}{4} \right] \tag{7.13}$$

and

$$\text{Surface Area} = 4\pi^2 R \left[\frac{D}{2} \right] \tag{7.14}$$

respectively. Write a Java program that will:

1. Interactively prompt a user for the torus radius `R` and the cross section diameter `D`.
2. Compute and print the volume and surface area of the torus.

Note. To ensure that the torus is well defined, your program should check that `R` and `D` are positive, and that `D/2` is smaller than `R`.

7.9 The fragment of code:

```
import java.lang.Math;

public class EulerMath {
   public static boolean isPrime( long num ) {

      if (num < 2 || (num % 2 == 0 && num != 2))
          return false;

      for (int i = 3; i <= Math.sqrt(num); i += 2)
          if (num % i == 0)
              return false;

      return true;
   }
}
```

defines a class called EulerMath. The method `isPrime()` that determines whether or not an integer (actually a long integer) integer `num` is prime. The method will return `true` if `num` is prime; otherwise it will return `false`.

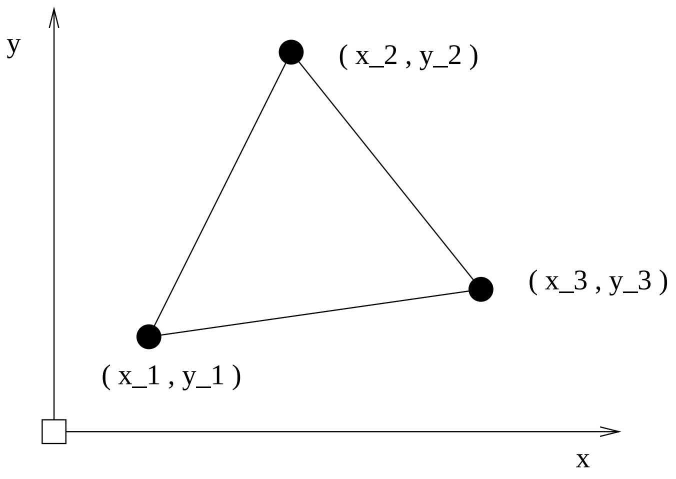

Figure 7.14. Vertex coordinates for triangle.

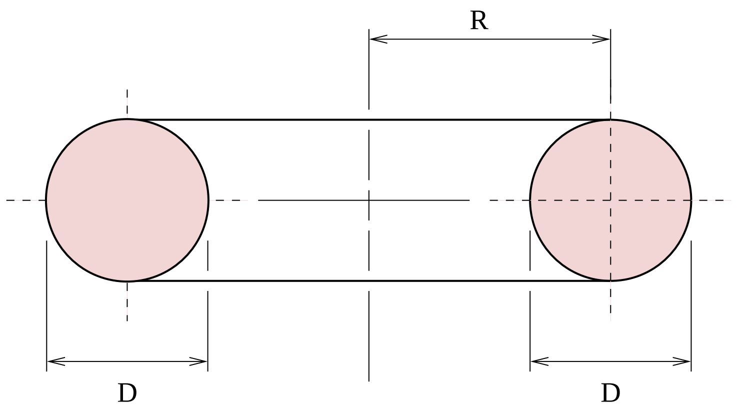

Figure 7.15. Cross-section and dimensions of torus.

Notice that the method declaration includes the keyword static. This makes `isPrime()` a class method, meaning that it can be called without first having to create an object. To call the method we simply write:

```
       EulerMath.isPrime ( ... );
```

and the result with either be true or false. (e.g., EulerMath.isPrime(4) evaluates to false).

Write a java program to find and print all of the prime numbers less than 1000 in a tidy table.

7.10 The prime 41, can be written as the sum of six consecutive primes:

$$41 = 2 + 3 + 5 + 7 + 11 + 13 \tag{7.15}$$

This is the longest sum of consecutive primes that adds to a prime below one-hundred.

The longest sum of consecutive primes below one-thousand that adds to a prime, contains 21 terms, and is equal to 953.

Which prime, below one-million, can be written as the sum of the most consecutive primes?

7.11 The velocity of water in a rectangular open channel is described by Manning's equation, namely:

$$U = \frac{\sqrt{S}}{n} \left[\frac{BH}{B + 2H} \right]^{2/3} \tag{7.16}$$

where B and H are the width and height of the channel (m), S is the channel slope, and n is the roughness coefficient.

Suppose that the properties of three open channels are stored in a rectangular array:

n	S	B	H
0.022	0.0003	15	2.5
0.020	0.0002	8	1.0
0.025	0.0001	10	2.0

Use a two-dimensional array to define and store the array problem parameters, and then compute the water speed for each of the three channel configurations.

Hint. A suitable program structure and array declaration is:

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

      // Define two-dimensional array of channel properties....

      double channel[][] = { { ... fill in details here ... },
                             { ... fill in details here ... },
                             { ... fill in details here ... } };

      // Print channel properties ....

      ... fill in details here ...

      // Compute and print water speeds ....

      ... fill in details here ...
   }
}
```

7.12 Use a simple looping construct to demonstrate that the series summation:

$$S = \frac{1}{1 \cdot 2} + \frac{1}{2 \cdot 3} + \frac{1}{3 \cdot 4} + \frac{1}{4 \cdot 5} + \cdots \tag{7.17}$$

approaches 1.

7.13 Leibnez's series is given by:

$$\frac{\pi}{4} = 1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \frac{1}{9} + \cdots \tag{7.18}$$

Write a Java program to compute Leibniz's series summation for 1000 terms. First, use a for-loop construct to compute the series summation directly. Then, repeat the experiment using an array for the series coefficients, and a for-loop construct to compute the sum of matrix element terms.

7.14 Write a Java program that uses a looping construct to find the least value of "n" so that

$$S = 1 + 2 + 3 \cdots + n > 10,000 \tag{7.19}$$

is true. For each iteration of the loop, print the sum S and whether or not the inequality is true or false. Check your result via the summation formula

$$S = 1 + 2 + 3 \cdots + n = \frac{n(n+1)}{2}. \tag{7.20}$$

7.15 The sum of the squares of the first ten natural numbers is,

$$1^2 + 2^2 + 3^2 + \cdots + 10^2 = 385. \tag{7.21}$$

The square of the sum of the first ten natural numbers is,

$$[1 + 2 + ... + 10]^2 = 55^2 = 3025. \tag{7.22}$$

Hence the difference between the sum of the squares of the first ten natural numbers and the square of the sum is 3025 - 385 = 2640.

Find the difference between the sum of the squares of the first one hundred natural numbers and the square of the sum.

7.16 Write a computer program that will find and print all non-negative integer values of x, y and z such that:

$$x^2 + y^2 + z^2 = 377. \tag{7.23}$$

Compute and print the maximum and minimum values of

$$x + y + z \tag{7.24}$$

that also satisfy equation 7.23.

7.17 Write a Java program that will print a list of points (x, y) on the graph of the equation

$$y(x) = \left[\frac{x^4 + \left[\frac{x}{sin(x)}\right]}{x - 2} \right] \tag{7.25}$$

for the range $-4 \le x \le 10$ in intervals of 0.25.

Note: You can approach this problem in one of two ways: (1) detect a numerical problem before it occurs and print out an appropriate message, or (2) evaluate $y(x)$ for all values of x and then test for various types of error. You should find that $y(0)$ and $y(2)$ evaluate to not-a-number (NaN) and positive infinity, respectively. These quantities can be tested for via the error condition constants `Double.POSITIVE_INFINITY` and `Double.NaN`.

7.18 A square of sheet metal having side length 2L cm has four pieces cut out symmetrically from the corners as shown in Figure 7.16. Assuming that L is a constant and L $>$ 2x, then the remaining metal can be folded into a pyramid.

Things to do:

1. Show that the pyramid volume is given by:

$$\text{Volume}(x) = \frac{4x^2}{3}\sqrt{(}L^2 - 2Lx)\text{cm}^3 \tag{7.26}$$

and that the maximum volume occurs when x = 2L/5 cm.

2. Write a Java program that will prompt a user for the length L, and then compute and print volumes for appropriate values of x. Organize your output into a tidy table, e.g., something like:

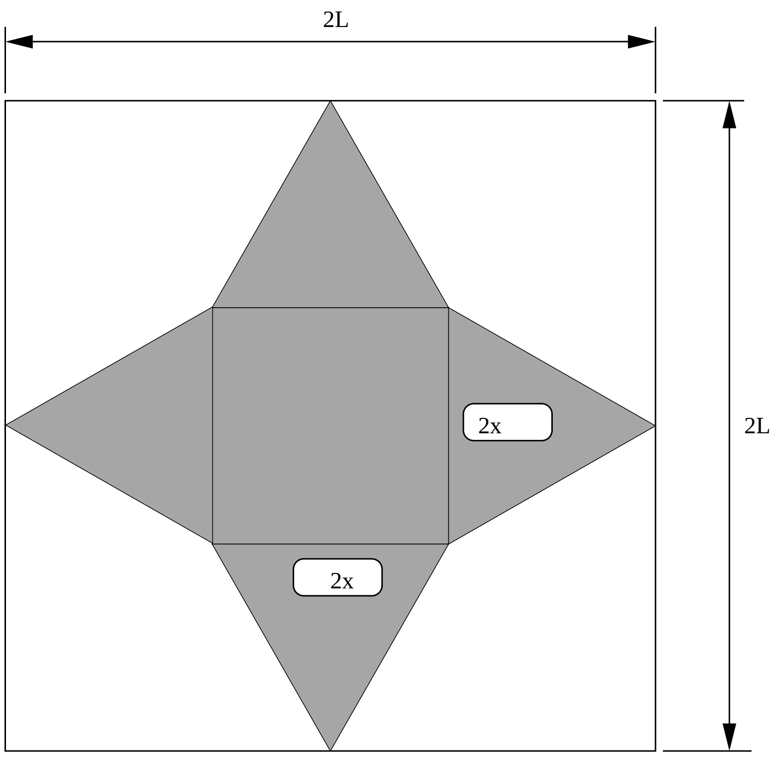

Figure 7.16. Sheetmetal schematic for a folded pyramid.

```
L = 20 cm
==========================================
             x (cm)         volume (cm^3)
==========================================
             ......              ........
             ......              ........
             ......              ........
             ......              ........
==========================================
```

3. Use your Java program to show that when L = 10 cm, the maximum pyramid volume occurs when x = 4cm.

7.19 Empirical studies indicate that the water-flow rate, Q, needed for business district firefighting is:

$$Q(p) = 3.86\sqrt{p}(1 - 0.01\sqrt{p}) \;\; m^3/min, \tag{7.27}$$

where p is the business district population in 1000s. Write a Java program to compute and print a table of water-flow rates for populations varying from 1,000 through 1,000,000, in increments of:

$$\text{Population Increment(p)} = \begin{cases} 1,000 & 1 < p < 10 \\ 10,000 & 10 < p \leq 100 \\ 100,000 & 100 < p \leq 1,000 \end{cases} \tag{7.28}$$

7.20 Suppose that during squally conditions, regular one second wind gusts produce a forward thrust on a yacht sail corresponding to

$$F(t) = \begin{cases} 4 + 15 \cdot t - 135 \cdot t^3 & 0.0 \leq t \leq 0.3, \\ (731 - 171t)/140 & 0.3 < t \leq 1.0 \end{cases} \tag{7.29}$$

F(t) has units kN. Write a Java program that computes and prints $F(t)$ for $0 \leq t \leq 3$ seconds, and conforms to the following specifications:

1. The wind force should be computed in a method WindForce() having the declaration:

```
public double WindForce ( double fTime );
```

2. The main() method should use a while or for looping construct to cycle over the time interval 0 through 3 seconds in increments of 0.25 seconds.

3. The program output should look something like the following:

```
     Time          Thrust
(seconds)            (kN)
=========================
     0.00            4.00
     0.25            5.64
     0.50            4.61
     0.75            4.31
     1.00            4.00
     1.25            5.64
     1.50            4.61
     1.75            4.31
     2.00            4.00
     2.25            5.64
     2.50            4.61
     2.75            4.31
     3.00            4.00
```

Hint. Two methods from the Math class library that you may find useful are `Math.pow()` and `Math.floor()`.

7.21 The array of floating point numbers

```
float fRainFall [] = { 1.1F, 0.8F, 1.1F, 1.2F, 0.5F, 0.2F, 0.05F, 0.0F,
                       0.0F, 0.0F, 0.4F, 0.5F, 0.6F, 0.8F,  1.0F, 1.2F,
                       3.0F, 2.4F, 1.5F, 1.0F, 1.0F, 0.0F,  0.0F, 0.0F,
                       0.0F, 0.0F, 0.0F, 2.0F, 2.0F, 2.4F };
```

stores the daily rainfall in a rain forest recorded over a one month period. Now suppose that we are intersted in computing the range and statistics of rainfall. The mean value of the data points is given by

$$\mu_x = \frac{1}{N} \left[\sum_{i=1}^{N} x_i \right] \tag{7.30}$$

and the standard deviation by

$$\sigma_x = \left[\sum_{i=1}^{N} \frac{[x_i - \mu_x]^2}{N} \right]^{1/2} = \left[\frac{1}{N} \sum_{i=1}^{N} [x_i^2] - \mu_x{}^2 \right]^{1/2}. \tag{7.31}$$

where N is the total number of data points.

Write a Java program to compute and print the range, mean, and standard deviation of daily rainfall for the one month period. Since at this point we are not creating objects (...these details will be covered next), method declarations need to explicitly say “this method can be called even if an instance of the class does not exist,” i.e.,

```
public static float minValue { float rainfall [] ) { .... }
public static float maxValue { float rainfall [] ) { .... }
public static float meanValue { float rainfall [] ) { .... }
public static float stdDeviation { float rainfall [] ) { .... }
```

Each method should return a basic data type (i.e., the numerical result) of type float.

7.22 Figure 7.17 shows a two-dimensional grid of masses.

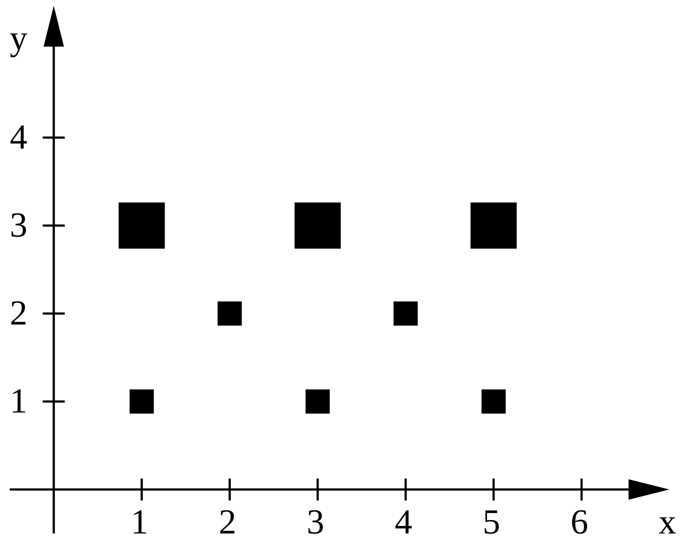

Figure 7.17. Two-dimensional grid of masses

If the total number of point masses is denoted by N, then the mass moments of inertia about the x- and y-axes are given by:

$$I_{xx} = \sum_{i=1}^{N} y_i^2 \cdot m_i \quad \text{and} \quad I_{yy} = \sum_{i=1}^{N} x_i^2 \cdot m_i \tag{7.32}$$

respectively. The polar moment of inertia is given by

$$I_{rr} = I_{xx} + I_{yy} = \sum_{i=1}^{N} \left[x_i^2 + y_i^2 \right] \cdot m_i \tag{7.33}$$

Now suppose that the (x,y) coordinates and masses are stored in two arrays;

```
double daMass[] = { 1.0, 1.0, 1.0, 1.0, 1.0, 2.0, 2.0, 2.0 };

double daaCoord[][] = { { 1.0,1.0 },
                        { 2.0,2.0 },
                        { 3.0,1.0 },
                        { 4.0,2.0 },
                        { 5.0,1.0 },
                        { 5.0,3.0 },
                        { 3.0,3.0 },
                        { 1.0,3.0 } };
```

Write a Java program to evaluate equations 7.32 and 11.4. Your program should: (1) use simple looping construct to systematically walk along the array elements, and (2) output the results of each evaluation.

7.23 **(Brain Teaser):** The modulo operator, `%`, computes the remainder that occurs after an integer `m` has been divided by a second integer `n`. For example,

```
m = 5, n = 3, 5 = 1*3 + 2  --> 5%3 evaluates to 2
m = 6, n = 3, 6 = 2*3 + 0  --> 6%3 evaluates to 0
m = 7, n = 3, 7 = 2*3 + 1  --> 7%3 evaluates to 1
m = 8, n = 3, 8 = 2*3 + 3  --> 8%3 evaluates to 2
```

and so forth. It is important to notice that `m%n` will always return an integer between 0 and (n-1). Now let A be a (7×7) matrix whose elements are given by

$$\text{A}(i,j) = \left[(i-1)^2 + (j-1)^2) \right] \%7. \tag{7.34}$$

Things to do:

1. Write a short Java program to evaluate and print equation 7.34.
2. Now let `p` and `q` be integers that cover the interval 0 through 100. Extend your Java program to find combinations of `p` and `q` where $p^2 + q^2$ will be divisible by 7.
3. Prove that if $p^2 + q^2$ is divisible by 7, then it will also be divisible by 49.

Hint. The last part of this problem is not as difficult as it looks. Write `p` as $7 * p_1 + r_1$ and `q` as $7 * q_1 + r_2$ and then an expression for $p^2 + q^2$. The result follows directly from the expression and the matrix element values in equation 7.34.

7.24 Figure 7.18 is a schematic of an irregular polygon having seven sides.

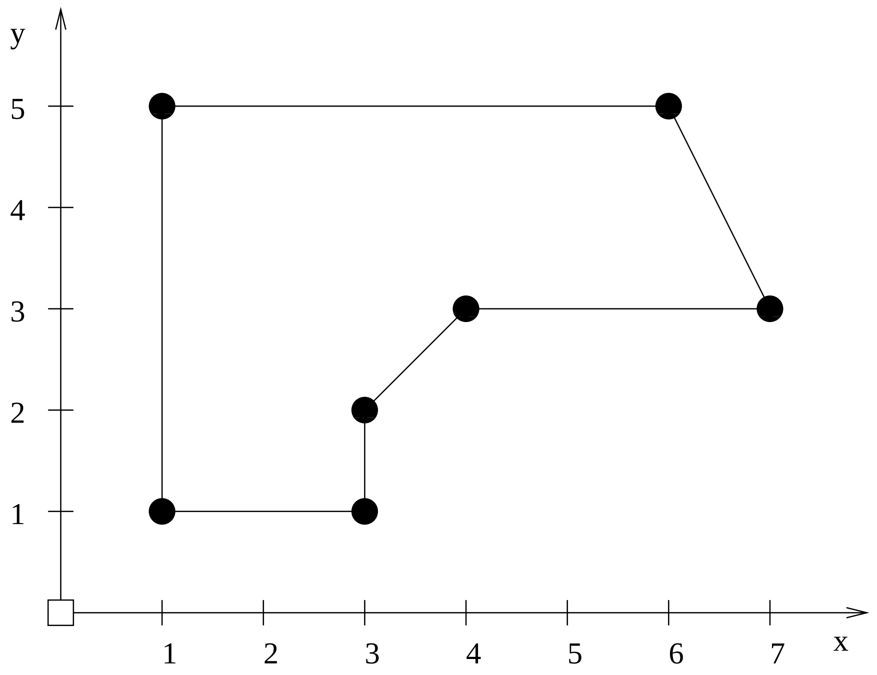

Figure 7.18. Seven-sided irregular polygon

Suppose that the x and y vertex coordinates are stored as two columns of information in the array

```
float faaPolygon [ 7 ][ 2 ] = { { 1.0, 1.0 },
                                { 1.0, 5.0 },
                                { 6.0, 5.0 },
                                { 7.0, 3.0 },
                                { 4.0, 3.0 },
                                { 3.0, 2.0 },
                                { 3.0, 1.0 } };
```

Write a Java program that will compute and print

1. The minimum and maximum polygon coordinates in both the x and y directions.
2. The minimum and maximum distance of the polygon vertices from the coordinate system origin.
3. The perimeter and area of the polygon.

Note. For Parts 1 and 2, use the `Math.max()` and `Math.min()` methods in java.lang.Math. In Part 3, use the fact that the vertices have been specified in a clockwise manner.

7.25 Figure 7.19 shows a sphere of radius r and density ρ floating in water.

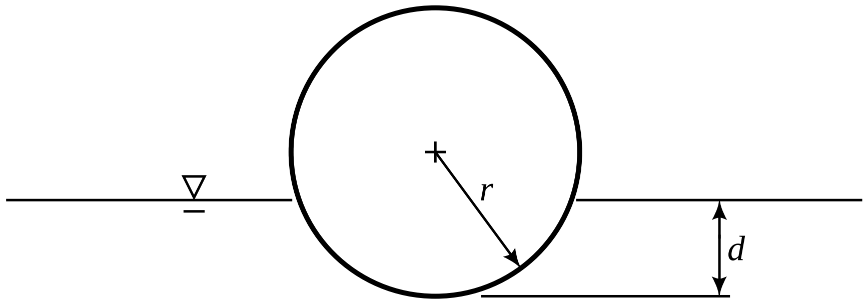

Figure 7.19. Sphere floating in water.

The weight of the sphere will be $4/3\rho\pi r^3$. The volume of the spherical segment displacing water is $1/3\pi(3rd^2 - d^3)$.

1. Show that the depth of the sphere floating in water is given by solutions to

$$f(x, \rho) = x^3 - 3.x^2 + 4.\rho = 0 \tag{7.35}$$

   where x = d/r is a dimensionless quantity.

2. Write a Java program that will compute the depth to which a sphere will sink as a function of its radius for $\rho = 0$ to $\rho = 1$ in increments of 0.1.

Hint. Perhaps the most straightforward way of solving this problem is to write a numerical procedure that computes the root of the cubic equation. This is not as hard as it might seem since $f(0, \rho)$ is always greater than zero and $f(2, \rho)$ is always less than zero. Only one solution to Equation 7.35 lies within the interval $f([0, 2], \rho)$ and so standard root finding techniques such as bisection and Newton Raphson will work.