Methods

Exercises and solutions

1. What is a method in Java?

**Answer:**

A method is a named block of code. A method in a class defines a behavior of the objects of that class or the behavior of the class itself.

1. Describe the difference between a static method and a non-static method of a class.

**Answer:**

A static method represents behavior of the class itself whereas a non-static method represents behavior of the instances of the class. A static method can refer to only static variables of the class. A non-static method can refer to class variables as well as instance variables of the class.

1. Can a static method access instance variables of a class? If your answer is no, explain the reason.

**Answer:**

No. The class definition, (including class variables) is always loaded into memory before the first instance of a class is created. It is not necessary to create an instance of a class to load its definition into memory. When a static method of a class is called, an instance of that class may not exist. Therefore, it is not allowed to refer to instance variables from inside a static method.

1. What is the meaning of void when it is used as the return type of a method?

**Answer:**

Return type void indicates that method does not return any value to the caller.

1. Create a class named Point2D with two int instance variables named x and y. Both instance variables should be declared private. Do not initialize the two instance variables. Add setters and getters for the two instance variables that will allow the users of the Point2D class to change and access their values. Declare the setters as setX(int x), setY(int y) and getters as getX() and getY().

**Solution:**

public class Point2D {  
 private int x;

private int y;

public int getX() {

return x;

}

public void setX(int x) {

this.x = x;

}

public int getY() {

return y;

}

public void setY(int y) {

this.y = y;

}  
}

1. Implement a method named distance in the Point2D class that you created in the previous exercise. The method accepts an instance of the Point2D class and returns the distance between the current point and the point represented by the parameter. The method should be declared as follows:  
     
   public class Point2D {  
    /\* Code from the previous exercise goes here. \*/  
     
    public double distance(Point2D p) {  
    /\* Your code for this exercise goes here. \*/  
    }  
   }  
     
   Hint: The distance between two points (x1, y1) and (x2, y2) is computed as . You can use Math.sqrt(n) method to compute the square root of a number n.

**Solution:**

public class Point2D {  
 private int x;

private int y;

public int getX() {

return x;

}

public void setX(int x) {

this.x = x;

}

public int getY() {

return y;

}

public void setY(int y) {

this.y = y;

}  
  
 public double distance(Point2D p) {  
 int xSqr = (this.x – p.x) \* (this.x – p.x);

int ySqr = (this.y – p.y) \* (this.y – p.y);

return Math.sqrt(xSqr + ySqr);  
 }  
}

1. Enhance the Point2D method by adding a static factory method named create(). A factory method in a class is used to create objects of the class. The create() method should be declared as follows:  
     
   public class Point2D {  
    /\* Code from the previous exercise goes here. \*/  
     
    public Point2D create(int x, int y) {  
    /\* Your code for this exercise goes here. \*/  
    }  
   }  
     
   The x and y instance variables of the returned Point2D object from the create() method should be initialized to the x and y parameters of this method, respectively.

**Solution:**

public class Point2D {  
 private int x;

private int y;

public int getX() {

return x;

}

public void setX(int x) {

this.x = x;

}

public int getY() {

return y;

}

public void setY(int y) {

this.y = y;

}  
  
 public double distance(Point2D p) {  
 int xSqr = (this.x – p.x) \* (this.x – p.x);

int ySqr = (this.y – p.y) \* (this.y – p.y);

return Math.sqrt(xSqr + ySqr);  
 }

public static Point2D create(int x, int y) {

Point2D p = new Point2D();

p.setX(x);

p.setY(y);  
 return p;

}  
}

1. Create a class named MathUtil with a method name avg(). It computes and returns average of a list of numbers. The method must accept variable-arguments of double types with a minimum of two double values. Run the MathUtil class and verify that the output prints the correct results.

// MathUtil.java  
package com.jdojo.cls.excercise;  
  
public class MathUtil {  
 public static void main(String[] args) {  
 System.out.println("avg(10, 15) = " + avg(10, 15));  
 System.out.println("avg(2, 3, 4) = " + avg(2, 3, 4));  
 System.out.println("avg(20.5, 30.5, 40.5) = "   
 + avg(20.5, 30.5, 40.5));  
 System.out.println("avg(-2.0, 0.0, 2.0) = "   
 + avg(-2.0, 0.0, 2.0));  
 }  
  
 public static double avg(/\* Your parameters goes here. \*/) {  
 /\* Your code goes here. \*/  
 }  
}

**Solution:**

public class MathUtil {

public static void main(String[] args) {

System.out.println("avg(10, 15) = " + avg(10, 15));

System.out.println("avg(2, 3, 4) = " + avg(2, 3, 4));

System.out.println("avg(20.5, 30.5, 40.5) = " + avg(20.5, 30.5, 40.5));

System.out.println("avg(-2.0, 0.0, 2.0) = " + avg(-2.0, 0.0, 2.0));

}

/\*\*

\* Declaring the first two double parameters will force the caller to

\* pass at least two arguments to this method.

\*/

public static double avg(**double n1, double n2**, double... others) {

double sum = n1 + n2;

for(double num : others) {

sum += num;

}

int count = 2 + others.length;

double average = sum/count;

return average;

}

}

1. The main() method of a class serves as an entry point of a Java application. It is declared as follows:  
     
   public static void main(String[] args) {  
    // Your code goes here  
   }  
     
   Change this declaration of the main() method using a varargs.

**Solution:**

public static void main(String… args) {  
 // Your code goes here  
}

1. What will be the output when the following PassByValueTest class is run?  
     
   // PassByValueTest.java  
   package com.jdojo.cls.excercise;  
     
   public class PassByValueTest {  
    public static void main(String[] args) {  
    int x = 100;  
    System.out.println("x = " + x);  
    change(x);  
    System.out.println("x = " + x);  
      
    Point2D p = new Point2D();  
    p.setX(40);  
    p.setY(60);  
    System.out.println("p.x = " + p.getX()   
    + ", p.y = " + p.getY());  
      
    changePointReference(p);  
    System.out.println("p.x = " + p.getX()   
    + ", p.y = " + p.getY());  
      
    changePoint(p);  
    System.out.println("p.x = " + p.getX()   
    + ", p.y = " + p.getY());  
    }  
     
    public static void change(int x) {  
    x = 200;  
    }  
     
    public static void changePointReference(Point2D p) {  
    p = new Point2D();  
    }  
      
    public static void changePoint(Point2D p) {  
    int newX = p.getX() / 2;  
    int newY = p.getY() / 2;  
    p.setX(newX);  
    p.setY(newY);  
    }  
   }

Answer:

x = 100

x = 100

p.x = 40, p.y = 60

p.x = 40, p.y = 60

p.x = 20, p.y = 30