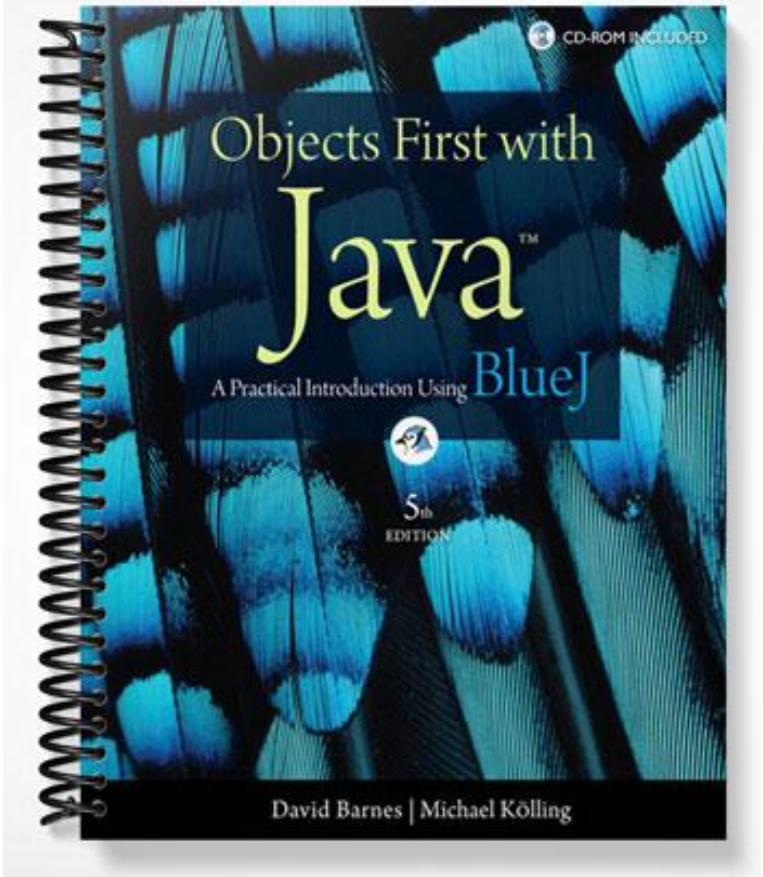
SOLUTIONS MANUAL



Exercise 1.3

Use a negative parameter value to move left, e.g. -70

Exercise 1.9

The House Picture

The main building:

- Create a new Square object
- Invoke its method makeVisible()
- Make the square bigger by invoking the method changeSize(newSize) (100 is a good size)
- Move the square down by invoking the method moveVertical(distance) (again 80 is a good value)

The window:

- Create a new Square object.
- Invoke its method makeVisible()
- Change its color by invoking changeColor()
- write "black" in the popupwindow
- Move the square down by invoking the method moveVertical(distance) (100 is a good value)
- Move it to the right by invoking moveRight()

The roof:

- Create a new triangle object.
- Invoke its method makeVisible()
- Change its size with changeSize(newHeight, newWidth) (50,140)
- moveVertical(70)
- moveHoizontal(60)

The Sun:

- Create new Circle object.
- Invoke its method makeVisible()
- Change its color by invoking changeColor() (write "yellow" in the popup window) Optionally change its size with changeSize(60)
- Move it to the right by invoking moveHorizontal(180)

The Hilltop Picture

The hill:

- Create a new Circle object.
- Invoke its makeVisible() method.
- Change its color by invoking changeColor (write “green” in the popup window) Change its size with changeSize to something like 1000.
- Move it left with moveHorizontal(-500).
- Move it down with moveVertical(125).

The sun:

- Create a Circle.
- Make it visible.
- Set its size to 30.
- Move it right by 150 pixels.
- Move it down by 50 pixels.

The larger figure:

- Create a new Person object.
- Invoke its makeVisible method.
- Change its size to something like 50 high and 25 wide.
- Move it left by 30 pixels and up by around 8.

The smaller figure:

- The size should be around 40 by 20.
- Move it left by 3 pixels and down by 2.

Exercise 1.14

- It uses the objects of the classes Circle, Square and Triangle.
- It then moves these objects to the right places and changes the sizes and colors of the obejcts. Essentially calling the same methods as used in exercise 1.9

Exercise 1.16

Change:

```
sun.changeColor("yellow");
```

to be:

```
sun.changeColor("blue");
```

Note that this change should be made in both the draw() and setColor() methods. Students often forget to do it in the latter. Illustrate the problem by drawing the picture, then calling setBlackAndWhite() then setColor() – the sun will have been changed to yellow, rather than blue, if the second change is not made.

Exercise 1.17

The second sun will need to be positioned somewhere different from the first sun to be visible. As with the previous exercise, it is common for students to miss the need to change the setColor and setBlackAndWhite methods for compatibility with the addition. This is an earlier introduction to the need for *regression testing*!

Exercise 1.18

After the line sun.makeVisible() insert the following:

- sun.slowMoveVertical(250);
- Compile the Picture class (Press compile in the editor window)
- Create instance of class Picture and invoke its draw() method.

Exercise 1.19

Remove the line (if added in the previous exercise): slowMoveVertical(250); Right below the last } after the draw() method, add the sunset() method :

```
/**
 * Animates the sunset.
 */
public void sunset()
{
    sun.slowMoveVertical(250);
}
```

Compile! And run it.

Exercise 1.20
Define a new field:

```
    private Person person;
```

Initialize and position them in the draw():

```
    person = new Person();
    person.changeSize(80, 40);
    // Place them at ground level.
    person.moveVertical(15);
    // Make sure they are to the right of the house to start.
    person.moveHorizontal(200);
```

Make them visible and move up to the house in sunset():

```
    person.makeVisible();
    // Walk up to the house.
    person.slowMoveHorizontal(-150);
```

Exercise 1.22

When calling the method getName(), the name of the student is displayed in a popup window. The name displayed is the one typed in when the object was created.

Exercise 1.24

It shows the number of students in the LabClass which is zero.

Exercise 1.31

Students looking these values up in the tutorial might not be able to readily identify that the integer types used here are int, as opposed to byte, short or long. Similarly, they may well suggest float rather than double for the final one. Be sure to point out that String always has an initial upper-case letter, because these subtleties are often missed.

0	int
"hello"	String
101	int
-1	int
true	boolean
"33"	String
3.1415	double

Exercise 1.32

First you would have to decide which type the field should have. String would be a good type to hold a name, so we add the following line to the source file of Circle:

```
    private String name;
```

The above line could be placed after this line in the source code of the Circle class:

```
    private boolean isVisible;
```

Exercise 1.33

```
public void send(String msg)
```

The name msg is arbitrarily chosen.

Exercise 1.34

```
public int average(int firstNumber, int secondNumber)
```

Exercise 1.35

The book is an object because it is *a specific instance* of the Book class.

Exercise 1.36

Yes, an object can belong to several classes. One of the more famous examples are

the platypus, which is both a mammal and egg-laying.