Chapter 7 Exercise Solutions

EX 7.1. Which of the following are valid declarations? Which instantiate an array object? Explain your answers.

```
int primes = {2, 3, 4, 5, 7, 11};
```

Invalid; an int cannot be declared and initialized using an intializer list. The brackets are missing.

```
float elapsedTimes[] = {11.47, 12.04, 11.72, 13.88};
```

Valid; the brackets can be placed either after the element type or after the reference variable. However, this is not the preferred technique. This declaration creates an array object.

```
int[] scores = int[30];
```

Invalid; the right hand side of the assignment operator must contain either an initializer list or a `new` *operation.*

```
int[] primes = new {2,3,5,7,11};
```

Invalid; "new" on the right hand side of the assignment operator is neither necessary nor acceptable.

```
int[] scores = new int[30];
```

Valid; the assignment is correct Java syntax. This declaration creates an array object.

```
char grades[] = {'a', 'b', 'c', 'd', 'f'};
```

Valid; the brackets can be placed either after the element type or after the reference variable. However, this is not the preferred technique. This declaration creates an array object.

```
char[] grades = new char[];
```

Invalid; the size of the array must be indicated when the array is instantiated.

EX 7.2. Describe five programs that would be difficult to implement without using arrays.

A program to find the average midterm score of 600 students enrolled in an introductory computer science course.

A program to record and compute the sum of the snowfalls, recorded on a daily basis for the 40 days preceding the Winter Olympics.

A program to determine the relative frequency of each character in the Cyrillic alphabet in the original version of The Brothers Karamasov.

A program to compute the mean and standard deviation of the Dow Jones Industrial Average closings since September 11, 2001.

A program to store the coordinates of the vertices of polygons approximating the surface of a beating heart.

EX 7.3. Describe what problem occurs in the following code. What modifications should be made to it to eliminate the problem?

```
int[] numbers = {3, 2, 3, 6, 9, 10, 12, 32, 3, 12, 6};
for (int count = 1; count <= numbers.length; count++)
   System.out.println (numbers[count]);
```

The for loop fails to print the 0th element of the array, and attempts to print the nonexistent 11th element of the array. As a consequence, an `ArrayIndexOutOfBoundsException` *is thrown. The problem can be eliminated by providing a for loop which initializes count to 0 (rather than 1) and tests if count is less than (rather than less than or equal to) numbers.length.*

EX 7.4. Write an array declaration and any necessary supporting classes to represent the following statements:

a. students' names for a class of 25 students

```
String[] students = new String[25];
```

b. students' test grades for a class of 40 students

```
int[] grades = new int[40];
```

or, for simple letter grades:

```
char[] grades = new char[40];
```

or, for letter grades that include pluses and minuses

```
String[] grades = new String[40];
```

c. credit-card transactions that contain a transaction number, a merchant name, and a charge

```
Transactions[] charges = new Transactions[MAX];

public class Transactions
{
   private int transactionNumber;
   private String merchantName;
   private double charge;
   // etc.
}
```

d. students' names for a class and homework grades for each student

```
Student[] myClass = new Student[MAX];

public class Student
{
   private String name;
   private int[] grades;
   // etc.
}
```

e. for each employee of the L&L International Corporation: the employee number, hire date, and the amount of the last five raises

```
Employee[]employees = new Employee[MAX];

public class Employee
{
   private int employeeNumber;
   private String hireDate;
```

```
      private double raise[] = new double[5];
      // etc.
   }
```

EX 7.5. Write code that sets each element of an array called nums to the value of the contstant `INITIAL`.

```
   for (int index = 0; index < nums.length; index++)
      nums[index] = INITIAL;
```

EX 7.6. Write code that prints the values stored in an array called names backwards.

```
   for (int index = names.length-1; index >= 0; index--)
      System.out.println(names[index]);
```

EX 7.7. Write code that sets each element of a `boolean` array called `flags` to alternating values (`true` at index 0, `false` at index 1, etc.).

```
   for (int index = 0; index < flags.length; index++)
      flags[index] = (index%2 == 0);
```

EX 7.8. Write a method called `sumArray` that accepts an array of floating point values and returns the sum of the values stored in the array.

```
   public float sumArray (float[] values)
   {
      float sum = 0;
      for (int index = 0; index < values.length; index++)
         sum += values[index];
      return sum;
   }
```

EX 7.9. Write a method called `switchThem` that accepts two integer arrays as parameters and switches the contents of the arrays. Take into account that the arrays may be of different sizes.

```
   public void switchThem (int[] first, int[] second)
   {
      if (first.length == second.length)
      {
         // copy contents of first into temp
         int [] temp = new int[first.length];
         for (int i=0; I < first.length; i++)
            temp[i] = first[i];

         //copy contents of second into first
         for (int i=0; I < first.length; i++)
            first[i] = second[i];

         //copy contents of temp into second
         for (int i=0; I < first.length; i++)
            second[i] = temp[i];
      }
      else
         System.out.println("Arrays are of different sizes");
   }
```

EX 7.10. Describe a program for which you would use the `ArrayList` class instead of arrays to implement choices. Describe a program for which you would use arrays instead of the `ArrayList` class. Explain your choices.

A program associated with a mail order Website for backpacks would use an object of the ArrayList class to implement the choices of colors of the backpacks because the colors and number of colors change with the seasons and as colors gain and lose popularity. An object of the `ArrayList` class can grow and shrink dynamically to accommodate these changes.

A program associated with a personal day planner, with entries possible for each hour of the day, would use an array object to implement the choices for each hour of the day because the number of hours in a day, and hence the number of hours for which choices can be made for any given day, never changes. There is no need for the array object to grow or shrink to accommodate a larger or smaller number of hours.

EX 7.11. What would happen if, in the `Dots` program, we did not provide empty definitions for one or more of the unused mouse events?

The compiler would complain because class `DotsListener`, which implements `MouseListener`, must provide definitions for all methods specified in the interface. If it doesn't, the class must be declared as `abstract`.

EX 7.12. The `Dots` program listens for a mouse pressed event to draw a dot. How would the program behave differently if it listened for a mouse released event instead? A mouse clicked event?

If the program listened instead for a mouse released event, the position and appearance of the dot would be determined at the time of the release, and wouldn't appear until the mouse button was released. Similarly, if the program listened for a mouse clicked event, the position and appearance of the dot would be determined at the time of a click (which requires a press followed by a release).

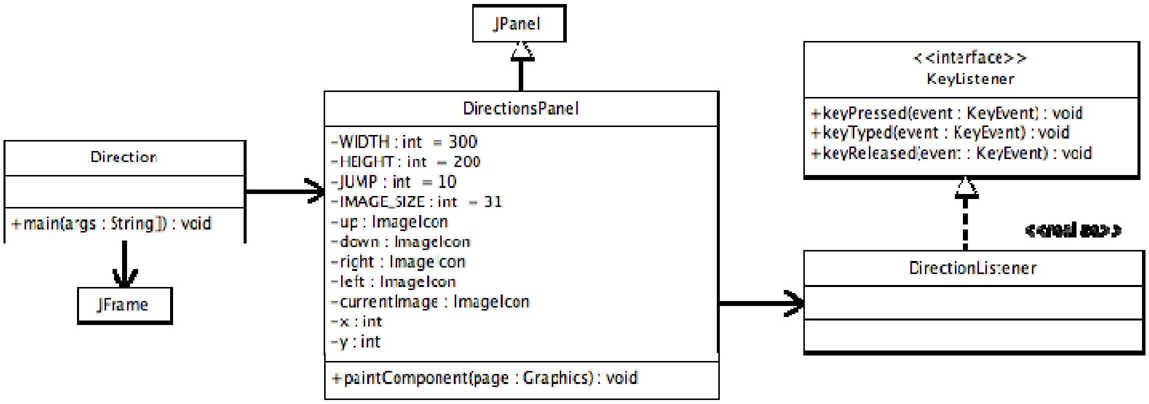
EX 7.13. What would happen if the call to `super.paintComponent` were removed from the `paintComponent` method of the `DotsPanel` class? Remove it and run the program to test your answer.

The `paintComponent` method is a member of class `DotsPanel` which extends `JPanel` which means that the call to `super.paintComponent` is a call to the `paintComponent` method of the `JPanel` class. Without this call, the `paintComponent` method of the `JPanel` class is not called. As a consequence, the background is not cleared each time. This means that there is no need to store and redraw all of the dots each time, but it causes other problems, such as the background color not being drawn and the count value overwriting itself so that it becomes unreadable.

EX 7.14. What would happen if the call to `super.paintComponent` were removed from the `paintComponent` method of the `RubberLinesPanel` class? Remove it and run the program to test your answer. In what ways is the answer different from the answer to Exercise 7.13?

Without this call, the `paintComponent` *method of the* `JPanel` *class is not called, and the background is not cleared and redrawn each time. As a consequence, previously drawn lines remain on the screen. Furthermore, since the lines are drawn over and over again as the mouse is dragged, moving the mouse in a curve causes several different lines to be drawn during one drag.*

EX 7.15. Create a UML class diagram for the `Direction` program.


JPanel
Direction
+main(args : String[]) : void
JFrame
DirectionsPanel
-WIDTH : int = 300
-HEIGHT : int = 200
-JUMP : int = 10
-IMAGE_SIZE : int = 31
-up : ImageIcon
-down : ImageIcon
-right : ImageIcon
-left : ImageIcon
-currentImage : ImageIcon
-x : int
-y : int
+paintComponent(page : Graphics) : void
<<interface>>
KeyListener
+keyPressed(event : KeyEvent) : void
+keyTyped(event : KeyEvent) : void
+keyReleased(event : KeyEvent) : void
DirectionListener