One-dimensional Arrays

Mitsu Ogihara

Department of Computer Science
University of Miami

Table of Contents

What is an array?

- An array is a struture to record series of data elements of a kind
- Each has a name and a specific size (i.e., the series length)
- The elements in an array are assigned consecutive indices starting from 0

An array specification

- To declare an array `x` of elements from type <type> we write:

  ```
  <type>[] x;
  ```

  Note that

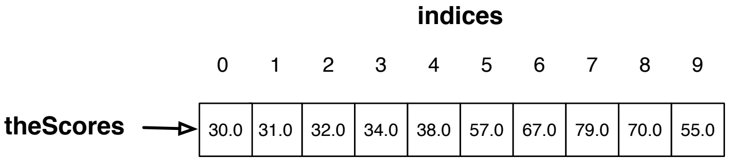
  ```
  <type> x;
  ```

  states that `x` is a variable of type <type>
- To create an array of size `n` elements of type <type> we write:

  ```
  <type>[ n ];
  ```

  The elements of `n` have indices from `0` to `n - 1`
  The element at position `i` is accessed by `x[i]`

View of an Array



Accessing Elements of an Array

As mentioned before:
`ARRAY-NAME[INDEX-VALUE]`
is the way to access an element at `INDEX-VALUE` of the array `ARRAY-NAME`

Accessing Elements of an Array

As mentioned before:
`ARRAY-NAME[INDEX-VALUE]`
is the way to access an element at `INDEX-VALUE` of the array `ARRAY-NAME`

To assign a new value to an array element, use:

```
ARRAY-NAME[INDEX-VALUE] = NEW-VALUE;
```

ArrayIndexOutOfBounds Exception

The declaration `double[10]`

- states that the indices are from 0 to 9
- and so, an attempt to access elements outside the range results in an error

Error Message Example

```
How many test scores? 10
The score no. 1: 30
. . .
The average is 49.30.
Enter an index to examine: -1
Exception in thread "main"
    java.lang.ArrayIndexOutOfBoundsException: -1
. . .
```

Accessing the Number of Elements in an Array

```
ARRAY-NAME.length
```

returns the number of elements in the array `ARRAY-NAME`
This is not a method; rather it is an instance variable of the array
The `length` for the class String is a method, since it demands the pair of parentheses

Illustrating Example

Suppose we want to write a code for receiving some scores and compute the average

- If just for once, the average can be computed using a cumulative algorithm by simply adding up the numbers
- If we want to make changes to the scores and recalculate the average we need to store the scores somewhere; for that, we use an array

The code

```java
import java.util.*;
public class AverageArray {
  public static void main( String[] args ) {
    Scanner console = new Scanner( System.in );
    System.out.print( "Enter # of scores: " );
    int number = console.nextInt();
    if ( number <= 0 ) {
      throw new IllegalArgumentException(
          "The number has to be positive" );
    }

    double[] theScores = new double[ number ];
    double average = 0;
    //-- receive and store data, update the average
    for ( int pos = 0; pos < number; pos ++ ) {
      System.out.printf( "Enter score no.%d: ", pos );
      theScores[ pos ] = console.nextDouble();
      average += theScores[ pos ] / number;
    }
    System.out.printf( "The average = %.2f%n", average );
```

Console

The code

```
import java.util.*;
public class AverageArray {
  public static void main( String[] args ) {
    Scanner console = new Scanner( System.in );
    System.out.print( "Enter # of scores: " );
    int number = console.nextInt();
    if ( number <= 0 ) {
      throw new IllegalArgumentException(
          "The number has to be positive" );
    }

    double[] theScores = new double[ number ];
    double average = 0;
    //-- receive and store data, update the average
    for ( int pos = 0; pos < number; pos ++ ) {
      System.out.printf( "Enter score no.%d: ", pos );
      theScores[ pos ] = console.nextDouble();
      average += theScores[ pos ] / number;
    }
    System.out.printf( "The average = %.2f%n", average );
```

Receive the number of elements from the user
If the number is not postive, terminate

The code

```
import java.util.*;
public class AverageArray {
  public static void main( String[] args ) {
    Scanner console = new Scanner( System.in );
    System.out.print( "Enter # of scores: " );
    int number = console.nextInt();
    if ( number <= 0 ) {
      throw new IllegalArgumentException(
          "The number has to be positive" );
    }

    double[] theScores = new double[ number ];
    double average = 0;
    //-- receive and store data, update the average
    for ( int pos = 0; pos < number; pos ++ ) {
      System.out.printf( "Enter score no.%d: ", pos );
      theScores[ pos ] = console.nextDouble();
      average += theScores[ pos ] / number;
    }
    System.out.printf( "The average = %.2f%n", average );
```

Declare and create the array; set the average to 0

The code

```java
import java.util.*;
public class AverageArray {
  public static void main( String[] args ) {
    Scanner console = new Scanner( System.in );
    System.out.print( "Enter # of scores: " );
    int number = console.nextInt();
    if ( number <= 0 ) {
      throw new IllegalArgumentException(
          "The number has to be positive" );
    }

    double[] theScores = new double[ number ];
    double average = 0;
    //-- receive and store data, update the average
    for ( int pos = 0; pos < number; pos ++ ) {
      System.out.printf( "Enter score no.%d: ", pos );
      theScores[ pos ] = console.nextDouble();
      average += theScores[ pos ] / number;
    }
    System.out.printf( "The average = %.2f%n", average );
```

For loop to iterate the indices from 0 to the length − 1

The code

```java
import java.util.*;
public class AverageArray {
  public static void main( String[] args ) {
    Scanner console = new Scanner( System.in );
    System.out.print( "Enter # of scores: " );
    int number = console.nextInt();
    if ( number <= 0 ) {
      throw new IllegalArgumentException(
          "The number has to be positive" );
    }

    double[] theScores = new double[ number ];
    double average = 0;
    //-- receive and store data, update the average
    for ( int pos = 0; pos < number; pos ++ ) {
      System.out.printf( "Enter score no.%d: ", pos );
      theScores[ pos ] = console.nextDouble();
      average += theScores[ pos ] / number;
    }
    System.out.printf( "The average = %.2f%n", average );
```

Prompt the user to enter an element and add it to the specific location

The code

```
import java.util.*;
public class AverageArray {
  public static void main( String[] args ) {
    Scanner console = new Scanner( System.in );
    System.out.print( "Enter # of scores: " );
    int number = console.nextInt();
    if ( number <= 0 ) {
      throw new IllegalArgumentException(
          "The number has to be positive" );
    }

    double[] theScores = new double[ number ];
    double average = 0;
    //-- receive and store data, update the average
    for ( int pos = 0; pos < number; pos ++ ) {
      System.out.printf( "Enter score no.%d: ", pos );
      theScores[ pos ] = console.nextDouble();
      average += theScores[ pos ] / number;
    }
    System.out.printf( "The average = %.2f%n", average );
```

Dynamic calculation of the average: add the entered number divided by the number of elements to the average

The code

```java
import java.util.*;
public class AverageArray {
  public static void main( String[] args ) {
    Scanner console = new Scanner( System.in );
    System.out.print( "Enter # of scores: " );
    int number = console.nextInt();
    if ( number <= 0 ) {
      throw new IllegalArgumentException(
          "The number has to be positive" );
    }

    double[] theScores = new double[ number ];
    double average = 0;
    //-- receive and store data, update the average
    for ( int pos = 0; pos < number; pos ++ ) {
      System.out.printf( "Enter score no.%d: ", pos );
      theScores[ pos ] = console.nextDouble();
      average += theScores[ pos ] / number;
    }
    System.out.printf( "The average = %.2f%n", average );
```

Print the average

The code

```
      int pos;
      do {
        System.out.print( "Which one do you want to see: " );
        pos = console.nextInt();
        if ( pos >= 0 && pos < number ) {
          // show the score
          System.out.printf( "Score=%.2f%n", theScores[ pos ] );
          System.out.print( "Enter a new value: " );
          double newScore = console.nextDouble();
          average = average - theScores[ pos ] / number
              + newScore / number;
          theScores[ pos ] = newScore;
          System.out.printf( "New average = %.2f%n", average );
        }
      } while ( pos >= 0 && pos < number );
    }
  }
```

Index variable

The code

```
    int pos;
    do {
      System.out.print( "Which one do you want to see: " );
      pos = console.nextInt();
      if ( pos >= 0 && pos < number ) {
        // show the score
        System.out.printf( "Score=%.2f%n", theScores[ pos ] );
        System.out.print( "Enter a new value: " );
        double newScore = console.nextDouble();
        average = average - theScores[ pos ] / number
            + newScore / number;
        theScores[ pos ] = newScore;
        System.out.printf( "New average = %.2f%n", average );
      }
    } while ( pos >= 0 && pos < number );
  }
}
```

Repeat as long as the value of `pos` is between 0 and `number - 1`

The code

```
    int pos;
    do {
      System.out.print( "Which one do you want to see: " );
      pos = console.nextInt();
      if ( pos >= 0 && pos < number ) {
        // show the score
        System.out.printf( "Score=%.2f%n", theScores[ pos ] );
        System.out.print( "Enter a new value: " );
        double newScore = console.nextDouble();
        average = average - theScores[ pos ] / number
            + newScore / number;
        theScores[ pos ] = newScore;
        System.out.printf( "New average = %.2f%n", average );
      }
    } while ( pos >= 0 && pos < number );
  }
}
```

Prompt the user and receive the value for `pos`

The code

```
    int pos;
    do {
      System.out.print( "Which one do you want to see: " );
      pos = console.nextInt();
      if ( pos >= 0 && pos < number ) {
        // show the score
        System.out.printf( "Score=%.2f%n", theScores[ pos ] );
        System.out.print( "Enter a new value: " );
        double newScore = console.nextDouble();
        average = average - theScores[ pos ] / number
            + newScore / number;
        theScores[ pos ] = newScore;
        System.out.printf( "New average = %.2f%n", average );
      }
    } while ( pos >= 0 && pos < number );
  }
}
```

The rest of the loop occurs if the value of `pos` is valid

The code

```java
    int pos;
    do {
      System.out.print( "Which one do you want to see: " );
      pos = console.nextInt();
      if ( pos >= 0 && pos < number ) {
        // show the score
        System.out.printf( "Score=%.2f%n", theScores[ pos ] );
        System.out.print( "Enter a new value: " );
        double newScore = console.nextDouble();
        average = average - theScores[ pos ] / number
            + newScore / number;
        theScores[ pos ] = newScore;
        System.out.printf( "New average = %.2f%n", average );
      }
    } while ( pos >= 0 && pos < number );
  }
}
```

Print the average and receive a new value for the position

The code

```
    int pos;
    do {
      System.out.print( "Which one do you want to see: " );
      pos = console.nextInt();
      if ( pos >= 0 && pos < number ) {
        // show the score
        System.out.printf( "Score=%.2f%n", theScores[ pos ] );
        System.out.print( "Enter a new value: " );
        double newScore = console.nextDouble();
        average = average - theScores[ pos ] / number
            + newScore / number;
        theScores[ pos ] = newScore;
        System.out.printf( "New average = %.2f%n", average );
      }
    } while ( pos >= 0 && pos < number );
  }
}
```

Update the average by subtracting the old score divided by the number and then adding the new score divided by the umber

The code

```
    int pos;
    do {
      System.out.print( "Which one do you want to see: " );
      pos = console.nextInt();
      if ( pos >= 0 && pos < number ) {
        // show the score
        System.out.printf( "Score=%.2f%n", theScores[ pos ] );
        System.out.print( "Enter a new value: " );
        double newScore = console.nextDouble();
        average = average - theScores[ pos ] / number
            + newScore / number;
        theScores[ pos ] = newScore;
        System.out.printf( "New average = %.2f%n", average );
      }
    } while ( pos >= 0 && pos < number );
  }
}
```

Only at this moment you can replace the score

The code

```
    int pos;
    do {
      System.out.print( "Which one do you want to see: " );
      pos = console.nextInt();
      if ( pos >= 0 && pos < number ) {
        // show the score
        System.out.printf( "Score=%.2f%n", theScores[ pos ] );
        System.out.print( "Enter a new value: " );
        double newScore = console.nextDouble();
        average = average - theScores[ pos ] / number
            + newScore / number;
        theScores[ pos ] = newScore;
        System.out.printf( "New average = %.2f%n", average );
      }
    } while ( pos >= 0 && pos < number );
  }
}
```

Print the average

Throwing Dice Made Easy

We can solve the problem of estimating the probability that a fair dice shows each face using an array

Instead of distinctly named six counts, we will use an array of size 6

ThrowDiceAgain.java

```java
import java.util.*;
public class ThrowDiceAgain {
  public static void main( String[] args ) {
    Scanner console = new Scanner( System.in );
    Random rand = new Random();
    int numberOfRounds, die;
    double[] counts = new double[ 6 ];
    System.out.print( "How many throws ? " );
    numberOfRounds = console.nextInt();
    for ( int round = 1; round <= numberOfRounds; round ++ ) {
      die = rand.nextInt( 6 );
      counts[ die ] += 1.0 / numberOfRounds;
    }
    for ( int index = 0; index < 6; index ++ ) {
      System.out.printf( "%d:%.6f%n",
          ( index + 1 ), counts[ index ] );
    }
  }
}
```

Array declaration

ThrowDiceAgain.java

```java
import java.util.*;
public class ThrowDiceAgain {
  public static void main( String[] args ) {
    Scanner console = new Scanner( System.in );
    Random rand = new Random();
    int numberOfRounds, die;
    double[] counts = new double[ 6 ];
    System.out.print( "How many throws ? " );
    numberOfRounds = console.nextInt();
    for ( int round = 1; round <= numberOfRounds; round ++ ) {
      die = rand.nextInt( 6 );
      counts[ die ] += 1.0 / numberOfRounds;
    }
    for ( int index = 0; index < 6; index ++ ) {
      System.out.printf( "%d:%.6f%n",
          ( index + 1 ), counts[ index ] );
    }
  }
}
```

Throw dice

Number of Days in a Month

An application for receive a month (1 .. 12) from the user and return the number of days in that month

Instead of 12, we will use 13 elements in an array

An array literal can be defined by:

```
<type>[] <name> = new <type>[] <element1>, ...
                  <elementk>;
```

We use:

```
int[] nDays = new int[] 0, 31, 28, ..., 31 ;
```

Number of Days in a Month

```java
import java.util.*;
public class NumberOfDaysInAMonth {
  public static void main( String[] args ) {
    Scanner console = new Scanner( System.in );
    int[] nDays = new int[]{ 0, 31, 28, 31, 30, 31, 30,
                                 31, 31, 30, 30, 30, 31 };
    int month;
    do {
      System.out.print( "Enter a month (0 to quit): " );
      month = console.nextInt();
      if ( month >= 1 && month <= 12 ) {
        System.out.printf( "The # of days in month %d is %d%n",
            month, nDays[ month ] );
      }
    } while ( month >= 1 && month <= 12 );
  }
}
```

Array declaration

Number of Days in a Month

```java
import java.util.*;
public class NumberOfDaysInAMonth {
  public static void main( String[] args ) {
    Scanner console = new Scanner( System.in );
    int[] nDays = new int[]{ 0, 31, 28, 31, 30, 31, 30,
                                31, 31, 30, 30, 30, 31 };
    int month;
    do {
      System.out.print( "Enter a month (0 to quit): " );
      month = console.nextInt();
      if ( month >= 1 && month <= 12 ) {
        System.out.printf( "The # of days in month %d is %d%n",
            month, nDays[ month ] );
      }
    } while ( month >= 1 && month <= 12 );
  }
}
```

Variable for the month

Number of Days in a Month

```java
import java.util.*;
public class NumberOfDaysInAMonth {
  public static void main( String[] args ) {
    Scanner console = new Scanner( System.in );
    int[] nDays = new int[]{ 0, 31, 28, 31, 30, 31, 30,
                                31, 31, 30, 30, 30, 31 };
    int month;
    do {
      System.out.print( "Enter a month (0 to quit): " );
      month = console.nextInt();
      if ( month >= 1 && month <= 12 ) {
        System.out.printf( "The # of days in month %d is %d%n",
            month, nDays[ month ] );
      }
    } while ( month >= 1 && month <= 12 );
  }
}
```

Prompt the user to receive the month

Number of Days in a Month

```java
import java.util.*;
public class NumberOfDaysInAMonth {
  public static void main( String[] args ) {
    Scanner console = new Scanner( System.in );
    int[] nDays = new int[]{ 0, 31, 28, 31, 30, 31, 30,
                            31, 31, 30, 30, 30, 31 };
    int month;
    do {
      System.out.print( "Enter a month (0 to quit): " );
      month = console.nextInt();
      if ( month >= 1 && month <= 12 ) {
        System.out.printf( "The # of days in month %d is %d%n",
            month, nDays[ month ] );
      }
    } while ( month >= 1 && month <= 12 );
  }
}
```

Print the number of the month if the onth is valid

Array Initialization

After creation (via `new TYPE[DIMENSION];`) the elements of the array are initialized by the TYPE's default value

- The number type and char: value 0
- boolean: false
- String: null (meaning “undefined”)
- Any object type: null

Examples

```java
public class InitialValue {
  // the constant for the SIZE of arrays
  public static int SIZE = 3;
  public static void main( String[] args ) {
    int[] integers = new int[ SIZE ];
    double[] doubles = new double[ SIZE ];
    boolean[] booleans = new boolean[ SIZE ];
    char[] chars = new char[ SIZE ];
    String[] strings = new String[ SIZE ];
    for ( int i = 0; i < SIZE; i ++ ) {
      System.out.printf( "i=%d: int:%d,double:%f,boolean:%b,",
        i, integers[ i ], doubles[ i ], booleans[ i ] );
      System.out.printf( "char:%c,string:%s%n",
        chars[ i ], strings[ i ] );
    }
  }
}
```

Creating the arrays

Examples

```
public class InitialValue {
  // the constant for the SIZE of arrays
  public static int SIZE = 3;
  public static void main( String[] args ) {
    int[] integers = new int[ SIZE ];
    double[] doubles = new double[ SIZE ];
    boolean[] booleans = new boolean[ SIZE ];
    char[] chars = new char[ SIZE ];
    String[] strings = new String[ SIZE ];
    for ( int i = 0; i < SIZE; i ++ ) {
      System.out.printf( "i=%d: int:%d,double:%f,boolean:%b,",
        i, integers[ i ], doubles[ i ], booleans[ i ] );
      System.out.printf( "char:%c,string:%s%n",
        chars[ i ], strings[ i ] );
    }
  }
}
```

Printing the entries
`%c` and `%b` are respectively for `char` and `boolean` in `printf` and `String.format`

Table of Contents

What Is Class Arrays

Class Arrays is a class that provides a variety of methods for manipulating arrays
To use Arrays, you need one of the following:

- import java.util.Arrays;
- import java.util.*;

Arrays Methods

Below T is any type

- **boolean Arrays.equals(T[] x, T[] y)**: returns whether the two arrays x and y are of the same length and element-wise equal to each other as a primitive data type
- **void Arrays.fill(T[] x, T v)**: fills the array T with the value v
- **T[] Arrays.copyOf(T[] x, int copyLength)**: creates and returns a copy of x of length copyLength; if the copyLength is greater than the length of x, the method fills the remaining part of the array with the default value of class T
- **void Arrays.sort(T[] x)**: reorders the elements of x in the increasing order

ArraysExample.java

```java
import java.util. * ;
// examples of using Arrays methods
public class ArraysExample {
  //--- print the elements of array in one line
  public static void print( double[] data ) {
    for ( int i = 0; i < data.length; i ++ ) {
      System.out.printf( "%.4f", data[ i ] );
      if ( i <= data.length - 2 ) {
        System.out.print( " " );
      }
    }
    System.out.println();
  }
```

Method for printing the elements on an array in a single line

ArraysExample.java

```java
import java.util. * ;
// examples of using Arrays methods
public class ArraysExample {
  //--- print the elements of array in one line
  public static void print( double[] data ) {
    for ( int i = 0; i < data.length; i ++ ) {
      System.out.printf( "%.4f", data[ i ] );
      if ( i <= data.length - 2 ) {
        System.out.print( " " );
      }
    }
    System.out.println();
  }
```

Use a for-loop to go through the indices
Use .length for the number of elements

ArraysExample.java

```java
import java.util. * ;
// examples of using Arrays methods
public class ArraysExample {
  //--- print the elements of array in one line
  public static void print( double[] data ) {
    for ( int i = 0; i < data.length; i ++ ) {
      System.out.printf( "%.4f", data[ i ] );
      if ( i <= data.length - 2 ) {
        System.out.print( " " );
      }
    }
    System.out.println();
  }
```

Print the data using printf with format `%.4f"`

ArraysExample.java

```java
import java.util. * ;
// examples of using Arrays methods
public class ArraysExample {
  //--- print the elements of array in one line
  public static void print( double[] data ) {
    for ( int i = 0; i < data.length; i ++ ) {
      System.out.printf( "%.4f", data[ i ] );
      if ( i <= data.length - 2 ) {
        System.out.print( " " );
      }
    }
    System.out.println();
  }
```

If not the last element print one white space

ArraysExample.java

```
import java.util. * ;
// examples of using Arrays methods
public class ArraysExample {
  //--- print the elements of array in one line
  public static void print( double[] data ) {
    for ( int i = 0; i < data.length; i ++ ) {
      System.out.printf( "%.4f", data[ i ] );
      if ( i <= data.length - 2 ) {
        System.out.print( " " );
      }
    }
    System.out.println();
  }
```

At the very end go to the next line

ArraysExample.java

```
  //--- compare two arrays for equality
  public static void compare( double[] array1, double[] array2 ) {
    System.out.print( "Array No.1: " );
    print( array1 );
    System.out.print( "Array No.2: " );
    print( array2 );
    if ( Arrays.equals( array1, array2 ) ) {
      System.out.println( "They are equal to each other." );
    }
    else {
      System.out.println( "They are not equal to each other." );
    }
  }
```

Method for comparing two arrays for equality
Note the parameter declaration

ArraysExample.java

```java
  //--- compare two arrays for equality
  public static void compare( double[] array1, double[] array2 ) {
    System.out.print( "Array No.1: " );
    print( array1 );
    System.out.print( "Array No.2: " );
    print( array2 );
    if ( Arrays.equals( array1, array2 ) ) {
      System.out.println( "They are equal to each other." );
    }
    else {
      System.out.println( "They are not equal to each other." );
    }
  }
```

Print the array number 1

ArraysExample.java

```
  //--- compare two arrays for equality
  public static void compare( double[] array1, double[] array2 ) {
    System.out.print( "Array No.1: " );
    print( array1 );
    System.out.print( "Array No.2: " );
    print( array2 );
    if ( Arrays.equals( array1, array2 ) ) {
      System.out.println( "They are equal to each other." );
    }
    else {
      System.out.println( "They are not equal to each other." );
    }
  }
```

Print the array number 1

ArraysExample.java

```
//--- compare two arrays for equality
public static void compare( double[] array1, double[] array2 ) {
  System.out.print( "Array No.1: " );
  print( array1 );
  System.out.print( "Array No.2: " );
  print( array2 );
  if ( Arrays.equals( array1, array2 ) ) {
    System.out.println( "They are equal to each other." );
  }
  else {
    System.out.println( "They are not equal to each other." );
  }
}
```

Compare the two arrays and report the outcome

ArraysExample.java

```java
  //--- main method
  public static void main( String[] args ) {
    double[] reals1 = new double[ 5 ], reals2 = new double[ 5 ];
    System.out.println( "----Filled the array with 0.5" );
    Arrays.fill( reals1, 0.5 );
    print( reals1 );
    System.out.println( "----Random elements and then copied" );
    for ( int index = 0; index < 5; index ++ ) {
      reals1[ index ] = Math.random();
      reals2[ index ] = reals1[ index ];
    }
    compare( reals1, reals2 );
    System.out.println( "----One array sorted" );
    Arrays.sort( reals1 );
    compare( reals1, reals2 );
  }
}
```

Main method

ArraysExample.java

```java
  //--- main method
  public static void main( String[] args ) {
    double[] reals1 = new double[ 5 ], reals2 = new double[ 5 ];
    System.out.println( "----Filled the array with 0.5" );
    Arrays.fill( reals1, 0.5 );
    print( reals1 );
    System.out.println( "----Random elements and then copied" );
    for ( int index = 0; index < 5; index ++ ) {
      reals1[ index ] = Math.random();
      reals2[ index ] = reals1[ index ];
    }
    compare( reals1, reals2 );
    System.out.println( "----One array sorted" );
    Arrays.sort( reals1 );
    compare( reals1, reals2 );
  }
}
```

Declare and create two double arrays of five elements each

ArraysExample.java

```java
  //--- main method
  public static void main( String[] args ) {
    double[] reals1 = new double[ 5 ], reals2 = new double[ 5 ];
    System.out.println( "----Filled the array with 0.5" );
    Arrays.fill( reals1, 0.5 );
    print( reals1 );
    System.out.println( "----Random elements and then copied" );
    for ( int index = 0; index < 5; index ++ ) {
      reals1[ index ] = Math.random();
      reals2[ index ] = reals1[ index ];
    }
    compare( reals1, reals2 );
    System.out.println( "----One array sorted" );
    Arrays.sort( reals1 );
    compare( reals1, reals2 );
  }
}
```

Fill the first one with 0.5 and print the contents

ArraysExample.java

```
  //--- main method
  public static void main( String[] args ) {
    double[] reals1 = new double[ 5 ], reals2 = new double[ 5 ];
    System.out.println( "----Filled the array with 0.5" );
    Arrays.fill( reals1, 0.5 );
    print( reals1 );
    System.out.println( "----Random elements and then copied" );
    for ( int index = 0; index < 5; index ++ ) {
      reals1[ index ] = Math.random();
      reals2[ index ] = reals1[ index ];
    }
    compare( reals1, reals2 );
    System.out.println( "----One array sorted" );
    Arrays.sort( reals1 );
    compare( reals1, reals2 );
  }
}
```

Generate random elements five times and store them in both arrays

ArraysExample.java

```java
  //--- main method
  public static void main( String[] args ) {
    double[] reals1 = new double[ 5 ], reals2 = new double[ 5 ];
    System.out.println( "----Filled the array with 0.5" );
    Arrays.fill( reals1, 0.5 );
    print( reals1 );
    System.out.println( "----Random elements and then copied" );
    for ( int index = 0; index < 5; index ++ ) {
      reals1[ index ] = Math.random();
      reals2[ index ] = reals1[ index ];
    }
    compare( reals1, reals2 );
    System.out.println( "----One array sorted" );
    Arrays.sort( reals1 );
    compare( reals1, reals2 );
  }
}
```

Compare the two arrays

ArraysExample.java

```java
  //--- main method
  public static void main( String[] args ) {
    double[] reals1 = new double[ 5 ], reals2 = new double[ 5 ];
    System.out.println( "----Filled the array with 0.5" );
    Arrays.fill( reals1, 0.5 );
    print( reals1 );
    System.out.println( "----Random elements and then copied" );
    for ( int index = 0; index < 5; index ++ ) {
      reals1[ index ] = Math.random();
      reals2[ index ] = reals1[ index ];
    }
    compare( reals1, reals2 );
    System.out.println( "----One array sorted" );
    Arrays.sort( reals1 );
    compare( reals1, reals2 );
  }
}
```

Sort the first one

ArraysExample.java

```
  //--- main method
  public static void main( String[] args ) {
    double[] reals1 = new double[ 5 ], reals2 = new double[ 5 ];
    System.out.println( "----Filled the array with 0.5" );
    Arrays.fill( reals1, 0.5 );
    print( reals1 );
    System.out.println( "----Random elements and then copied" );
    for ( int index = 0; index < 5; index ++ ) {
      reals1[ index ] = Math.random();
      reals2[ index ] = reals1[ index ];
    }
    compare( reals1, reals2 );
    System.out.println( "----One array sorted" );
    Arrays.sort( reals1 );
    compare( reals1, reals2 );
  }
}
```

Compare the two arrays