GIFT UNIVERSITY
Realize Your Career Dreams

GIFT School of Engineering and Applied Sciences

Spring 2021

CS-240: Object-oriented Programming

Lab-3 Manual

Methods and Two-Dimensional Arrays

v1.3
3/30/2020

Task #1: Using Arrays and Methods

In this task, you are being asked to write methods that manipulate arrays in Java.

1. Write a method called **`fillArray`** that takes a two-dimensional integer array as argument and fills the array with the user entered numbers.

   You may use the following header for this method:

   ```
   static void fillArray(int[][] array)
   ```

2. Write a method called **`printArray`** that takes a two-dimensional integer array as argument and prints all elements of array. The method should print each row line by line.

   You may use the following header for this method:

   ```
   static void printArray(int[][] array)
   ```

Take two numbers from the user as the size of rows and columns of the array, call the method **`fillArray`**, then call the method **`printArray`**.

The following code can be used for taking input from user in two-dimensional array:

```
Scanner input = new Scanner(System.in);
for (int i = 0; i < array.length; ++i) {
     for (int j = 0; j < array[i].length; ++j){
          System.out.println("Enter the value for index " +
     j + " of row at index " + i + ": ");
          array[i][j] = input.nextInt();
     }//for
}//for
```

NOTE: Perform input validation so that the size of rows and columns must be greater than 0.

1. Create a program called **ArrayPrintLab4.java.**
2. Create appropriate variables and assign values using a Scanner object.
3. Correctly display appropriate messages.

Task #2: Using Arrays and Methods

In this task, you are being asked to write methods that manipulate arrays in Java.

1. Write a method called **`fillArray`** that takes a two-dimensional integer array as argument and fills the array with the user entered numbers.

   You may use the following header for this method:

   ```
   static void fillArray(int[][] array)
   ```

2. Write another method called **`printSumAverage`** which will take a two-dimensional integer array as argument and prints the total sum and average of the array.

   You may use the following header for this method:

   ```
   static void printSumAverage (int[][] array)
   ```

   For example, if we pass the array **`{{4,5,9}, {1,2,3}, {9,10,11}, {19,25,26}}`** to the method, then the following output should be displayed:

   ```
   Total Sum is: 124
   Average is: 10.33
   ```

NOTE: Declare two-dimensional array and take input from the user using ***`fillArray`*** method.

1. Create a program called **ArraySumAverageLab4.java.**
2. Create appropriate variables and assign values using a Scanner object.
3. Correctly display appropriate messages.

Task #3: Using Arrays and Methods

In this task, you are being asked to write methods that manipulate arrays in Java.

1. Write a method called **`fillArray`** that takes a two-dimensional integer array as argument and fills the array with the user entered numbers.

   You may use the following header for this method:

   ```
   static void fillArray(int[][] array)
   ```

2. Write another method called **`printSumAverageByRows`** which will take a two-dimensional integer array as argument and prints the sum and average of each row of the array.

   You may use the following header for this method:

   ```
   static void printSumAverageByRows(int[][] array)
   ```

   For example, if we pass the following two-dimensional array to this method:

   ```
   5 7 9 4 3 1
   2 4 9 7 6 5
   3 6 8 7 9 4
   1 9 7 5 6 3
   ```

   Then the method should print:

   ```
   Row at index 0:
           Sum: 29
           Average: 4.83
   Row at index 1:
           Sum: 33
           Average: 5.50
   Row at index 2:
           Sum: 37
           Average: 6.17
   Row at index 3:
           Sum: 31
           Average: 5.17
   ```

NOTE: Declare two-dimensional array and take input from the user using ***`fillArray`*** method.

1. Create a program called **ArraySumAverageByRowsLab4.java.**
2. Create appropriate variables and assign values using a Scanner object.
3. Correctly display appropriate messages.

Task #4: Using Arrays and Methods

In this task, you are being asked to write methods that manipulate arrays in Java.

1. Write another method called **`replaceNegativeWithZeroes`** which will take a two-dimensional integer array as argument and replace all negative values from the array with zeroes.

   You may use the following header for this method:

   ```
   static void replaceNegativeWithZeroes(int[][] array)
   ```

2. Write another method called **`printArray`** which will take a two-dimensional integer array as argument and prints all elements of array. The method should print each row line by line.

   You may use the following header for this method:

   ```
   static void printArray (int[][] array)
   ```

Take two numbers from the user as the size of rows and columns of the array and call the methods in the following order:

- **`printArray()`**
- **`replaceNegativeWithZeroes()`**
- **`printArray()`**

NOTE: Declare and initialize the array without taking input from user.

You can initialize a two-dimensional array as follows:

```
int[][] array = new int[][] {{4,5,9}, {1,2,3}, {9,10,11},
                                                 {19,25,26}};
```

1. Create a program called **ArrayReplaceNegativesLab4.java.**
2. Correctly display appropriate messages.

Task #5: Using Arrays and Methods

In this task, you are being asked to write methods that manipulate arrays in Java.

1. Write a method called **`swapFirstAndLastRow`** which will take a two-dimensional integer array as argument and swap all the values of first row and last row with each other.

   You may use the following header for this method:

   ```
   static void swapFirstAndLastRow(int[][] array)
   ```

   For example, if we pass the following two-dimensional array to this method:

   ```
   5 7 9 4 3 1
   2 4 9 7 6 5
   3 6 8 7 9 4
   1 9 7 5 6 3
   ```

   Then the array will become:

   ```
   1 9 7 5 6 3
   2 4 9 7 6 5
   3 6 8 7 9 4
   5 7 9 4 3 1
   ```

2. Write a third method called **`printArray`** which will take a two-dimensional integer array as argument and prints all elements of array. The method should print each row line by line.

   You may use the following header for this method:

   ```
   static void printArray (int[][] array)
   ```

Take two numbers from the user as the size of rows and columns of the array and call the methods in the following order:

- **`printArray()`**
- **`swapFirstAndLastRow()`**
- **`printArray()`**

NOTE: Perform input validation so that the size of rows and columns must be greater than 0.

1. Create a program called **ArraySwapRowsLab4.java.**
2. Correctly display appropriate messages.

Task #6: Using Arrays and Methods

1. Write a method called **`getSumOfLeftDiagonal`** which will take a two-dimensional integer array as argument. If the rows and columns are equal in size, then the method should return the sum of the left diagonal of the array, or **-1** otherwise.

   You may use the following header for this method:

   ```
   static int getSumOfLeftDiagonal(int[][] array)
   ```

   For example, if we pass the following two-dimensional array to this method:

   ```
   5 7 9 4 3
   2 4 9 7 6
   3 6 8 7 9
   1 9 7 5 6
   3 8 7 9 2
   ```

   Then the method should return **`24`** as the sum of the diagonal.

   For the following two-dimensional array, the method should return **`-1`** because the rows and columns are not equal.

   ```
   5 7 9 4 3
   2 4 9 7 6
   3 6 8 7 9
   1 9 7 5 6
   ```

NOTE: Declare and initialize the array without taking input from user.

1. Create a program called **ArraySumOfDiagonalLab4.java.**
2. Correctly display appropriate messages.

Task #7: Using Arrays and Methods

1. Write a method called **`getSumOfMiddleColumns`** which will take a two-dimensional integer array as argument. The method should return the sum of middle column if the number of columns are odd, or return the average of two middle columns if the number of columns are even.

   You may use the following header for this method:

   ```
   static double getSumOfMiddleColumns(int[][] array)
   ```

   For example, if we pass the following two-dimensional array to this method:

   ```
   5 7 9 4 3
   2 4 9 7 6
   3 6 8 7 9
   1 9 7 9 6
   ```

   Then the method should return **`33.0`**.

   If we pass the following two-dimensional array to this method:

   ```
   5 7 9 4 3 1
   2 4 9 7 6 3
   3 6 8 7 9 8
   1 9 7 8 6 5
   ```

   Then the method should return **`29.5`** (the average of the middle two columns).

NOTE: Declare and initialize the array without taking input from user.

1. Create a program called **ArraySumOfMiddleColumnsLab4.java.**
2. Correctly display appropriate messages.

Task #8: Using Arrays and Methods

1. Write a method called `getSumByRows` which will take a two-dimensional integer array as argument, and returns the sum of each row as a one-dimensional integer array.

   You may use the following header for this method:

   ```
       static int[] getSumByRows(int[][] array)
   ```

   If the numbers of rows are **5** of a two-dimensional array, then the method should return a one-dimensional array of size **5** having sum of each row of the one-dimensional array.

   For example. if you pass the following two-dimensional array to the method:

   ```
   {{4, 5, 9}, {1,2,3}, {9, 10, 11}, {19, 25, 26}, {10,11, 12}};
   ```

   Then the method should return the following one-dimensional array

   ```
   {18, 6, 30, 70, 33}
   ```

NOTE: Declare and initialize the array without taking input from user.

1. Create a program called **ArraySumByRowsLab4.java.**
2. Correctly display appropriate messages.

Task #9: Using Arrays and Methods

1. Write a method called **`addMatrices`** which will take two matrices (two-dimensional integer arrays) as argument, and returns the addition of these two matrices as a two-dimensional integer array.

   You may use the following header for this method:

   ```
   static int[][] addMatrices(int[][] firstMatrix, int[][]
   secondMatrix)
   ```

   For example, if we pass the following two two-dimensional arrays to the method:

   ```
   5 7 9 4 3      7 3 2 1 5
   2 3 9 7 6      3 9 4 0 3
   3 6 8 7 9      6 2 8 9 4
   1 9 7 5 6      5 0 3 8 7
   3 8 7 9 2      1 6 8 4 3
   ```

   Then the method should return the following two-dimensional array:

   ```
   12 10 11  5 8
   5  12 13  7 9
   9  8  16 16 13
   6  9  10 13 13
   4  14 15 13 5
   ```

2. Write a method called **`printArray`** that takes a two-dimensional integer array as argument and prints all elements of array. The method should print each row line by line.

   You may use the following header for this method:

   ```
   static void printArray(int[][] array)
   ```

   In the main method:

   - Declare and initialize two two-dimensional arrays.
   - Pass these two-dimensional arrays to **`addMatrices()`** method.
   - Print the resulting matrix by calling **`printArray()`** method.

NOTE: Declare and initialize the arrays without taking input from user.

1. Create a program called **ArrayMatricesAdditionLab4.java.**
2. Correctly display appropriate messages.

Task #10: Using Arrays and Methods

1. Write a method called **`isIdentityMatrix`** which will take a two-dimensional integer array as argument, and returns true if the array is an identity matrix, or false otherwise.

   You may use the following header for this method:

   ```
   static boolean isIdentityMatrix(int[][] array)
   ```

   A matrix is an identity matrix if it is a square matrix, and all its elements at the left diagonal are ones, and all other elements are zeroes.

   For example, the following matrix is an identity matrix and the method should return *true* if we pass it to the **`isIdentityMatrix`** matrix as two-dimensional:

   ```
   1 0 0 0 0
   0 1 0 0 0
   0 0 1 0 0
   0 0 0 1 0
   0 0 0 0 1
   ```

   For the following two-dimensional array, the method should return *false* because there are some *non-zero elements* in the matrix.

   ```
   1 0 0 5 0
   0 1 0 2 0
   0 3 1 0 0
   0 0 0 1 5
   0 5 0 0 1
   ```

   Similarly, for the following two-dimensional array, the method should return *false* because some elements at left diagonal are *not ones*.

   ```
   1 0 0 0 0
   0 0 0 0 0
   0 0 0 0 0
   0 0 0 1 0
   0 0 0 0 1
   ```

NOTE: Declare and initialize the array without taking input from user.

1. Create a program called **ArrayIdentityMatrixLab4.java.**
2. Correctly display appropriate messages.