Spring 2021

CS-240: Object-oriented Programming - Lab

Lab-2 Manual

Methods and Arrays – Review

GIFT School of Engineering and Applied Sciences

v1.3
3/30/2020

Task #1: Arrays and Methods

In this task, you are being asked to write a method that manipulates an array in Java.

Write a method called **`sumEvenOdd`** that accepts one integer array as argument and prints the sum of all even and odd numbers in separate lines.

You may use the following header for this method:

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

For example, if we pass **`{3, 2, 12, 15, 17, 22, 25, 26, 28}`** to this method, then the method should print:

```
Sum of all even numbers: 90
Sum of all odd numbers: 60
```

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

1. Create a program called **ArraySumEvenOddLab2.java**
2. Correctly display appropriate messages.

Task #2: Arrays and Methods

In this task, you are being asked to write a method that manipulates an array in Java.

Write a method called **`printPrimes`** that accepts one integer array as argument and print all the prime numbers form the array.

You may use the following header for this method:

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

For example, if you pass **`{2, 4, 31, 12, 7, 47, 63, 41, 67, 55}`** to this method, then the method should print **`2, 31, 7, 47, 41, 67`**, as prime numbers.

To help with finding a prime number, the **`printPrimes()`** method would call another method:

```
static boolean isPrime(int number)
```

that takes an integer as an argument, and returns **true** if that number is a prime, otherwise, returns **false.**

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

1. Create a program called **ArrayPrintPrimesLab2.java**
2. Correctly display appropriate messages.

Task #3: Arrays and Methods

In this task, you are being asked to write a method that manipulates arrays in Java.

Write a method called **`printCommon`** that accepts two integer arrays as arguments and print all the common elements from the two arrays.

You may use the following header for this method:

```
static void printCommon(int[] firstArray, int[] secondArray)
```

For example, if the elements of two arrays are **`{4, 2, 3, 17, 19, 12, 16, 7, 100}`** and **`{3, 9, 5, 12, 6, 17, 8, 7}`**, then the method should print **`{3, 17, 12, 7}`**.

NOTE:

- Declare and initialize both arrays without taking input from user.
- Both arrays do not need to be of the same size.

1. Create a program called **ArrayPrintCommonLab2.java**
2. Correctly display appropriate messages.

Task #4: Arrays and Methods

In this task, you are being asked to write a method that manipulates arrays in Java.

Write a method called **`printUnique`** that accepts one integer array as argument, and print all unique elements form the array.

You may use the following header for this method:

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

For example, if you pass **`{2, 4, 2, 15, 4, 5, 6, 9, 12, 2}`** to this method, then the method should print **`15, 5, 6, 9, 12`** as unique elements.

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

1. Create a program called **ArrayPrintUniqueLab2.java.**
2. Correctly display appropriate messages.

Task #5: Arrays and Methods

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

1. Write a method called **`replaceWithNextMultiple`** that accepts two arguments. The first argument is an integer array, and second argument is an integer **number**. The method should replace all the elements with the *next multiple* of the **number** argument**.**

   You may use the following header for this method:

   ```
   static void replaceWithNextMultiple(int[] array, int number)
   ```

   For example, if we pass **`{3, 8, 13, 18, 21, 24, 29, 36, 37, 41}`** as first argument and **`4`** as second argument, then the method should change the array to **`{4, 8, 16, 20, 24, 24, 32, 36, 40, 44}`**. Note that, if an element is already a multiple of the number argument then the method should not change that element.

2. Write another method called **`printArray`** that accepts one integer array as argument and prints all the elements of the array.

   You may use the following header for this method:

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

NOTE:

- Declare and initialize the array without taking input from user.
- Print the array before and after calling the **`replaceWithNextMultiple`** method.

1. Create a program called **ArrayReplaceWithNextMultipleLab2.java.**
2. Correctly display appropriate messages.

Task #6: Arrays and Methods

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

1. Write a method called **`sortArrayZeroOneTwo`** that accepts one integer array of **0's**, **1's** and **2's** as argument, and sort the array in such a way that all **0's** comes first, all **1's** comes in middle and all **2's** comes at the end of the array.

   The method should only sort the array if all the elements are either **0, 1 or 2**, otherwise it should return the original array as it is.

   You may use the following header for this method:

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

   For example, if we pass **`{1, 2, 1, 0, 0, 2, 1, 0, 1, 2, 2, 1, 1, 1, 0, 2, 1}`** then the method should sort it as: **`{0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2}`**.

2. Write another method called **`isValidArray`** that accepts one integer array as argument, and return **true** if the all elements of the array are either **0 , 1 or 2**, or **false** otherwise.

   You may use the following header for this method:

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

3. Write a third method called **`printArray`** that accepts one integer array as argument and prints all elements of the array.

You may use the following header for this method:

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

NOTE:

- Declare and initialize the array without taking input from user.
- Declare and initialize an integer array, call the method **`sortArrayZeroOneTwo`**. This method will then call the **`isValidArray`** method and will only sort the array if it gets **true** as the result.
- Print the array before and after calling the **`sortArrayZeroOneTwo`** method.

1. Create a program called **ArraySortLab2.java.**
2. Correctly display appropriate messages.

Task #7: Arrays and Methods

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

Write a method called **`sumOfArrays`** which will take two integer arrays as argument and return an integer **array** having the sum of these two arrays.

You may use the following header for this method:

```
public static int[] sumOfArrays(int[] arrayA, int[] arrayB)
```

For example, if we pass **`{2, 4, 9, 12, 15, 5}`** and **`{7, 4, 3, 6, 2, 9}`** to this method, then the method should return **`{9, 8, 12, 18, 17, 14}`** as an integer array.

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

1. Create a program called **SumArraysLab2.java.**
2. Correctly display appropriate messages**.**

Task #8: Arrays and Methods

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

Write a method called **`fillArrays`** which will take one integer array, one String array and one integer number (length of id) as argument.

You may use the following header for this method:

```
public static String fillArrays(String[] nameArray, int[] idArray, int idLength)
```

The method ask the user to enter the id and name of the student one by one (using a loop of your choice). Please note that the length of the id of student should be equal to the ***`idLength`*** parameter. To check the validity of length, you may use the following method inside this method:

```
public static boolean isIdValid(int id, int idLength){
  int minRange = 1;
  for (int i = 0; i < idLength - 1; ++i){
    minRange *= 10;
  }//for
  int maxRange = (minRange * 10) - 1;
  if (id >= minRange && id <= maxRange){
    return true;
  }
  else{
    return false;
  }//if
}//isIdValid
```

The method ***`isIdValid`*** will return true if id is valid, or false otherwise.

If the length of id of student is not valid, then the program must ask repeatedly for the valid id and this process will continue until the user enters valid id. This whole process will run ***`n times`*** where n is the length of ***nameArray*** or ***idArray***. (The length of both arrays will be same.)

Sample run of the method: (Assume that the length of nameArray and idArray is 3.)

```
Enter the length of the id: 7
```

Enter the id of the student at index [0]: 1724052

Enter the name of the student at index [0] : Ali

Enter the id of the student at index [1]: 18240525

Invalid id entered.

Enter the id of the student at index [1]: 1824025

Enter the name of the student at index [1] : Abdullah

Enter the id of the student at index [2]: 1624026

Enter the name of the student at index [2] : Abdulrehman

Write another method called **`getNameOfStudent`** which will take one integer array, one String array and one integer number (id of the student) as argument and return name of student as a String from String array whose id is equal to the id (parameter).

You may use the following header for this method:

```
public static String getNameOfStudent(String[] nameArray,
int[] idArray, int id)
```

The program will return "***`Not found`***" as String if the ***`id`*** does not exist in ***`idArray`***.

In the main method:

- Ask the user for the size of the array.
- Declare the ***`nameArray`*** and ***`idArray`*** of size entered by user.
- Then ask for the length of the id (Make sure that the id length should be greater than or equal to 5 and less than or equal to 9.)
- Call ***`fillArrays`*** method by passing appropriate arguments.
- Then Call ***`getNameOfStudent`*** by passing appropriate arguments.
- Display the *name of student* or "***`Not found`***" message according to the output of ***`getNameOfStudent`*** method.

1. Create a program called **StudentLab2.java.**
2. Correctly display appropriate messages**.**