```java
public class ArrayOps {
   public static void main(String[] args) {

   }

   // Input: array with integers.
   // Output: smallest missing integer from array.
   public static int findMissingInt (int [] array) {
      for (int i = 0; i < array.length; i++) {
         if (!isInArray(i, array)) {
            return i;
         }
      }
      return array.length; //if array has all previous integers, it doesn't include the next integer.
   }

   // Input: array
   // Output: second largest value in the array.
   // Assumption: the array has at least 2 values
   public static int secondMaxValue(int [] array) {
      int largest = Math.max(array[0], array[1]);
      int secondLargest = Math.min(array[0], array[1]);
      for (int i = 2; i < array.length; i++) {
         if (array[i] > largest) {
            // found new largest: change both values accordingly
            secondLargest = largest;
            largest = array[i];
         } else if(array[i] > secondLargest) {
            // found new secondLargest: change value accordingly
            secondLargest = array[i];
         }
      }
      return secondLargest;
   }

   // Input: 2 arrays;
   // Output: boolean whether or not the arrays contain all of eachothers numbers
```

```java
public static boolean containsTheSameElements(int [] array1,int [] array2) {
    for (int i = 0; i < array1.length; i++) {
        if (!isInArray(array1[i], array2)) {
            // checks if all array1 values are in array2
            return false;
        }
    }
    for (int i = 0; i < array2.length; i++) {
        if (!isInArray(array2[i], array1)) {
            // checks if all array2 values are in array1
            return false;
        }
    }
    return true;
}

// Input: array
// Output: whether or not the array is Sorted = Ascending/Descending
// Equal values count as both possibly ascending or descending
public static boolean isSorted(int[] array) {
    if (array.length < 2){
        // if has 0 or 1 value, it is Sorted.
        return true;
    }
    int i = 1;
    while (i < array.length && array[i] == array[i-1]) {
        // checks if the array starts off ascending
        i++;
    }
    if (i == array.length) {
        // all values in array are equal = isSorted
        return true;
    }
    boolean startsAscending = (array[i] > array[i-1]);
    if (startsAscending) {
        // checks if the continuation of the array is also ascending
        for (int j = 1; j < array.length; j++) {
```

```
            if (array[j] < array[j-1]) {
                return false;
            }
        }
    } else {
        // checks if the continuation of the array is also descending
        for (int j = 1; j < array.length; j++) {
            if (array[j] > array[j-1]) {
                return false;
            }
        }
    }
    return true;
}

// returns true if num is in an array
// if num isn't in array - returns false
public static boolean isInArray(int num, int[] arr) {
    for (int i = 0; i < arr.length; i++) {
        if (arr[i] == num) {
            return true;
        }
    }
    return false;
}
}
```

```java
public class StringOps {

    public static void main(String[] args) {

    }

    // Input: String
    // Output: same string but capitalizes all vowels and makes all consonants lowercase.
    public static String capVowelsLowRest (String string) {
        String result = "";
        String vowels = "aeiouAEIOU";
        for (int i = 0; i < string.length(); i++) {
            char currentLetter = string.charAt(i);
            if(vowels.indexOf(currentLetter) == -1) {   // currentLetter isn't a vowel
                result += toLowercase(currentLetter);
            } else {                                    // currentLetter is a vowel
                result += capitalize(currentLetter);
            }
        }
        return result;
    }

    // Input: string
    // Output: 1) lowercase the first letter of first word
    //         2) first letters of all other words will be capitalized
    //         3) all other letters will be lowercase
    //         4) all spaces are removed
    public static String camelCase (String string) {
        if (string.length() == 0) {
            return ("");
        }
        int startIndex = 0;
        while (startIndex < string.length() && string.charAt(startIndex) == ' ') { // to ignore spaces before the first word
            startIndex++;
        }
        String result = "";
        boolean capitalizeNext = false;     // so that first letter will be lowercase
```

```
    for (int i = startIndex; i < string.length(); i++) {
        char currentLetter = string.charAt(i);
        if (currentLetter == ' ') {     // capitalize after spaces and don't add to result
            capitalizeNext = true;
        } else {                        // else add to result and next letter should be lowercase
            if(capitalizeNext) {
                result += capitalize(currentLetter);
            } else {
                result += toLowercase(currentLetter);
            }
            capitalizeNext = false;
        }
    }
    return result;
}

public static int[] allIndexOf (String string, char chr) {
    int counter = 0;     //counts needed length for future array (result)
    for (int i = 0; i < string.length(); i++) {
        if (string.charAt(i) == chr) {
            counter ++;
        }
    }
    int[] result = new int[counter];
    int placement = 0;
    for (int i = 0; i < string.length(); i++) {
        if (string.charAt(i) == chr) {
            result[placement++] = i;
        }
    }
    return result;
}

// Input: char
// Output: capital form of the letter
//         does not change other chars
public static char capitalize(char c) {
```

```
    if ('a' <= c && c <= 'z') {
        // turn lowercase letters to capital
        int letterIndex = (c - 'a');
        return ((char) ('A' + letterIndex));
    }
    return c; // needed for compilation
  }

  // Input: char
  // Output: lower case form of the letter
  //         does not change other chars
   public static char toLowercase(char c) {
    if ('A' <= c && c <= 'Z') {
        // turn capital letters to lowercase
        int letterIndex = (c - 'A');
        return ((char) ('a' + letterIndex));
    }
    return c; // needed for compilation
  }

  public static void printArray (int[] array) {
    System.out.print("{");
    if (array.length > 0) {
        System.out.print(array[0]);
        for (int i = 1; i < array.length; i++) {
            System.out.print(", " + array[i]);
        }
    }
    System.out.println("}");
  }

}
```