COSC 601 – Lab 1 – Arrays

Your first two labs in COSC 601 will involve creating a utility class of simple but useful array methods. Create a single class called ArrayUtils without a package. All of the methods you create will go into this class.

1. Take the code we wrote in class for finding max and min values in an array and create two methods:

    max()
    min()

Your methods will both take an array of ints as parameters and return a single int value which will be either the max value from the array or min value from the array.

2. Using the code we wrote in class, create a method called total() which will take an array of ints as a parameter and return the sum of all items in the array.

3. Create a countOdd method

- create a method that takes an array of ints as a parameter and returns an int value
- name the method countOdd
- set up a counter variable called counterOdd;
- set up a for-each loop to go through the array with a variable called value
- for each value in the array, check if its odd. if it is, increment the counter variable
- return counterOdd

4. Create a toString method that outputs the contents of an array in a nice format.

- create a method that takes an array of ints as a parameter and returns a string
- name the method toString
- set up two variables: a string called result initialized as "[" and an int called counter
- set up a for-each loop to go through the array with variable called strVal
- for each strVal,
    - add the value to the result string and then add a ", "
    - increment the counter
    - use the counter to add a line break \n to result every 10 characters by checking if its divisible by 10
- after the loop is done, finish the result string by first getting rid of the last two characters since there will be a ", " leftover.
- add a "]" to the result string
- return result

5. Create a reverseToString method that outputs the contents of an array in reverse order in a nice format.

This method will work exactly the same as the previous. instead of counting from 0 to the end, you'll count from the end to 0
Your output should look the same as the previous but in reverse order.

6. create a timesOccur method to count the num of occurrences of a value

- create a method that takes an array of ints and an int value, key, as parameters and returns an int value
- name the method timesOccur
- set up a counter variable called countTimes;
- set up a for-each loop to go through the array with a variable called value
- for each value in the array, check if its the same as key. If it is, increment the counter variable
- return countTimes

7. create a reverse method to reverse the values in the array

- create a method that takes an int array as a parameter and returns an int array
- call the method reverse
- start by declaring a new array to return results in. Call it returnArr set its size to be the same as your input parameter's size
- declare a counter variable and initialize it to 0
- set up a for loop with your loop variable counting from the length of the input array-1 to 0
- for each iteration of your loop, set your returnArr using the current value of the counter as index and assign the value from the input array using the loop var as index
- once the loop is completely done, return your returnArr

8. Create the following helpers:

    - isOdd
    - isEven

- Create two simple helper methods, each of which takes an int as a parameter and returns a boolean. The methods will return true or false depending on whether the int parameter is odd or even.
For example, an even int passed to isOdd will return false. An even int passed to isEven will return true.

9. Create a method that returns only the odd values from an inputted array.

- create a method that takes an int array as a parameter and returns an int array
- call the method oddValues

- start by declaring a new array to return results in. Call it returnArr set its size using your countOdd method you created previously.
- declare a counter variable and initialize it to 0
- set up a for loop with your loop variable counting from 0 through the length of the input array
- for each iteration of your loop, check the current value of the input array to see if its odd. if it is, write it to the return array using your counter above as the index and then increment the counter
- once the loop is completely done, return your returnArr

10. Create an array copy method that does a complete copy of one array to another

- create a method that takes an int array as a parameter and returns an int array
- call the method copyArr
- start by declaring a new array to return results in. Call it returnArr set its size to be the same as your input parameter's size
- declare a counter variable and initialize it to 0
- set up a for-each loop to go through the input array using a variable called arrVal
- for each item of your array, assign the value to the return array using the counter as index and then increment the counter
- once the loop is completely done, return your returnArr

11. Create a copy method that works like ArrayCopy

- create a boolean method that takes two int arrays and one int value as parameters
- name your parameters arrSrc, arrDest for the arrays and sizeCpy for the int
- call the method copy
- start by doing some checking
    - first find the smallest array size by setting an int var minArraySize to the length of either arrSrc or arrDest, whichever is smaller
    - next, set another int var, copySize to the smaller of minArraySize and sizeCpy
- set up a for loop initialized to 0 and using copySize as the stopping condition.
- at each iteration assign the value from arrSrc to arrDest using the current index value

13. Create a reduce method which reduces an array to a given size.

- Create a method that takes an int array and int value as parameters and returns an int array
- call the method reduce
- first set a new int variable to either the size of the inputted array or the int parameters, whichever is smaller
- next use this int variable to set the size of new array called returnArr that will be the return array
- call your copy method with the inputted array, your return array and your variable from the last step. This will set returnArr to the result we want. No need for any loop here.
- return your return array

14. Create a grow method which increases an array by a specified amount

- Create a method that takes an int array and int value as parameters and returns an int array call the method grow
- create a new int var called newSize by adding the size of the inputted array and the int parameter
- if newSize is less than or equal to zero, return null
- create a return array using newSize as its size
- again, call your copy method with the inputted array, the return array and newSize
- return your return array

Testing

To test your methods, I will be making available a JUnit test class.