Programming 1

Tutorial 8

Activity 1

Given the following 2D array:

```
int[][] table = {
        {8, 1, 6},
        {3, 5, 7},
        {4, 9, 2}
};
```

Print the following table on the console:

```
+-----------+
| 8 | 1 | 6 |
+-----------+
| 3 | 5 | 7 |
+-----------+
| 4 | 9 | 2 |
+-----------+
```

Hint

Apply what you learned about using nested `for` loops for multi-dimentional array in the lecture. It is also possible to use the enhanced for loop (for-each loop) in this exercise.

Deliverable

PrintTable.java

Activity 2

Write a method `public static String cleanText(String content)` which converts the content to lowercase and remove all the special characters from the extracted content. Essentially, only `a-z`, `0-9` and white spaces are allowed to exist in the content. Write a `main` method to test the `cleanText` method.

Sample output

```
Original: Roses are red. Violets are blue, Sugar is sweet... And so are you.
Cleaned: roses are red violets are blue sugar is sweet and so are you
```

Hint

This method should make use of `StringBuilder` to achieve better performance than using the regular `String`. You need to return a `String` as output of this method. Use `toString()` method to get a `String` out of a `StringBuilder` object.

Deliverable

TextCleaning.java

Activity 3

Write a method `removeEvenLength` that takes an `ArrayList` of strings as a parameter, then removes all of the strings of even length from the list. The method header should be:

```
public static void removeEvenLength(ArrayList<String> arrList)
```

Supply a main method to test this `removeEvenLength` method.

Hint

In this case, the method receives a reference to an `ArrayList` object and modifies its content. So the purpose is to modify the existing `ArrayList` object, not creating and returning a new one. This situation is called "Call by Reference" and it applies to all types of objects including arrays. Write a main method to test this method.

- Use `ArrayList.remove(int index)` to remove an element at a certain index.
- Use `String.length() % 2 == 0` to check if a `String` has even length.
- Because `ArrayList` size decreases when you remove an element, avoid looping in an increasing manner (i.e. from `0` to `size() - 1`). Try to loop decreasingly from `size() - 1` to `0` instead.

Sample output

```
Original list: [odd, even, abc, abcd, abcde, abcdef]
Output list: [odd, abc, abcde]
```

Deliverable

RemoveEvenLength.java

Activity 4

(optional)

Write a program to generate 100 random integers between 2 and 1000 and print out the probability (in percentage) that a number is a prime number.

Sample output *(may vary between multiple runs because of randomness)*

```
Among 100 random integers, 15 are prime numbers
Probability is 15%
```

Hint

Use `Math.random()` to generate a double value x where: $0.0 \leq x < 1.0$. Then scale x in some way so as to obtain the desired random integer.

Or use `nextInt(int bound)` method from `java.lang.Random` to generate an integer value x where: $0 \leq x < \text{bound}$. Then further modify x to obtain the desired random integer.

Deliverable

PrimeProbability.java

Activity 5

(optional)

Write a method `sumWithoutSmallest` that computes the sum of an array of values, except for the smallest one, using only a single loop. Write a main method to test this method.

Some test cases:

```
{5,4,1,3,2} -> 14
{3,2,2,1,2,3,4} -> 16
{3,2,1,2,1,2,3,4} -> 17
```

(In the last case where there are two instances of the smallest value, only one instance is excluded from the sum)

Hint

Calculate the sum and find the smallest value in a single loop. Then subtract the smallest number from the sum amount.

Deliverable

SumWithoutSmallest.java

Activity 6 (*)

(optional)

Write a static method named `countLastDigits` that accepts an array of integers as a parameter and examines its elements to determine how many end in `0`, how many end in `1`, how many end in `2`, etc. Your method will return an array of counts. The count of how many elements end in `0` should be stored in its element at index `0`, how many of the values end in `1` should be stored in its element at index `1`, and so on.

For example, if an array named `list` contains the values:

```
{9, 29, 44, 103, 2, 52, 12, 12, 76, 35, 20}
```

…the call of `countLastDigits(list)` should return the array:

```
{1, 0, 4, 1, 1, 1, 1, 0, 0, 2}
```

Deliverable

CountLastDigits.java

Activity 7 (*)

Write a method named `shiftRight` that accepts an array of integers as a parameter and shifts the values in the array to the right (forward) by one position. Each element moves right by one, except the last element, which moves to the front. For example, if a variable named `list` refers to an array containing the values `{3, 8, 19, 7}`, the call of `shiftRight(list)` should modify it to contain `{7, 3, 8, 19}`. A subsequent call of `shiftRight(list)` would turn the array into the following: `{19, 7, 3, 8}`.

Hint

Because the input array is passed into the method by reference, there are 2 different ways to do this exercise:

1. Create a new array, copy elements from input array over, return the new array.

   ```
   public static int[] shiftRight(int[] a)
   ```

2. Write the method as `void` and manipulate the input array. In this way, the input array is modified.

   ```
   public static void shiftRight(int[] a)
   ```

Deliverable

ShiftRight.java