NUS National University of Singapore | Computing

NUS WebMail IVLE LIBRARY MAPS

Search search for... in NUS Websites GO

CodeCrunch

Home | My Courses | Browse Tutorials | Browse Tasks | Search | My Submissions | Logout | Logged in as: **e0817840**

CS2030 Lab #7: Java Streams

Tags & Categories

Tags:

Categories:

Related Tutorials

Task Content

Java Streams

Topic Coverage

- Application of Java Streams

Requirements

- Java Streams are to be used for the method implementations as specified in this assignment

The Tasks

There are several tasks in this assignment.

For each task, you are to define the appropriate method(s) within the `Main` class in `Main.java`. You may also submit other utility classes if necessary.

Task 1 — Count Twin Primes

A prime number is a natural number greater than `1` that is only divisible by `1` and itself. A twin prime is one of a pair of prime numbers with a difference of `2`. For example, `41` and `43` are twin primes.

Define the method `countTwinPrimes` in class `Main` which takes in an integer n and counts the number of distinct twin primes from 0 until n inclusive.

```
static long countTwinPrimes(int n)
```

Save your `Main` class in the file `Main.java`.

```
jshell> Main.countTwinPrimes(100)
$.. ==> 15

jshell> Main.countTwinPrimes(2)
$.. ==> 0

jshell> Main.countTwinPrimes(3)
$.. ==> 1
```

Task 2 — Reverse String

Define the method `reverse` in class `Main` that takes in a `String str` and returns the reverse of `str`.

```
static String reverse(String str)
```

Save your `Main` class in the file `Main.java`.

```
jshell> Main.reverse("orange")
$.. ==> "egnaro"

jshell> Main.reverse("one two three")
$.. ==> "eerht owt eno"

jshell> Main.reverse("")
$.. ==> ""

jshell> Main.reverse("the quick brown fox jumps over the lazy dog.")
$.. ==> ".god yzal eht revo spmuj xof nworb kciuq eht"
```

Task 3 — Counting Repeats

Define the method `countRepeats` in class `Main` that takes in an integer array of digits `0` to `9` and returns the number of occurrences of adjacent repeated digits. You may assume that there are at least three elements in the array.

```
static long countRepeats(int... array)
```

For example,

- the array `{0, 1, 2, 2, 1, 2, 2, 1, 3, 3, 1}` has three occurrences of repeated digits
- the array `{0, 1, 1, 1, 1, 2}` has one occurrence

Save your `Main` class in the file `Main.java`.

```
jshell> Main.countRepeats(0,1,2,2,1,2,2,1,3,3,1)
$.. ==> 3
```

Task 4 — Normalized Mean

Given a list `T` of `n` integers t_i, the normalized value of each t_i is defined as

$$\bar{t_i} = \frac{t_i - \min_{\mathbf{T}}}{\max_{\mathbf{T}} - \min_{\mathbf{T}}}$$

where $\min_T$ and $\max_T$ represent the minimum and maximum values among all `n` values in `T`.

For example, the list of values `{1,2,3,4,5}` upon normalizing will become `{0,0.25,0.5,0.75,1}` since $\min_T = 1$ and $\max_T=5$.With the set of normalized values generated, the normalized mean can be easily computed to be `0.5`.

Notice from the above that finding the normalized mean requires values in the list to be accessed twice: once for finding the maximum/minimum, and a second time to compute each normalized value and finding the mean.

Alternatively, we can re-expressed the normalized mean as

$$\bar{t}_{mean} = \frac{\sum_i \bar{t_i}}{n} = \frac{\sum_i \frac{t_i - \min_{\mathbf{T}}}{\max_{\mathbf{T}} - \min_{\mathbf{T}}}}{n} = \frac{\frac{\sum_i t_i - n\min_{\mathbf{T}}}{\max_{\mathbf{T}} - \min_{\mathbf{T}}}}{n} = \frac{\frac{\sum_i t_i}{n} - \min_{\mathbf{T}}}{\max_{\mathbf{T}} - \min_{\mathbf{T}}}$$

This way we need to only access each element in the list exactly once.

Define the method `normalizedMean` in class `Main` that takes in a Stream of Integer elements and returns the normalized mean

```
static double normalizedMean(Stream<Integer> stream)
```

Save your `Main` class in the file `Main.java`.

```
jshell> Main.normalizedMean(Stream.<Integer>of(1, 2, 3, 4, 5))
$.. ==> 0.5

jshell> Main.normalizedMean(Stream.<Integer>of(1, 1))
$.. ==> 0.0

jshell> Main.normalizedMean(Stream.<Integer>of(1))
$.. ==> 0.0
```

```
jshell> Main.normalizedMean(Stream.<Integer>of())
$.. ==> 0.0
```

Terms of Use | Privacy | Non-discrimination

MySoC | Computing Facilities | Search | Campus Map

School of Computing, National University of Singapore