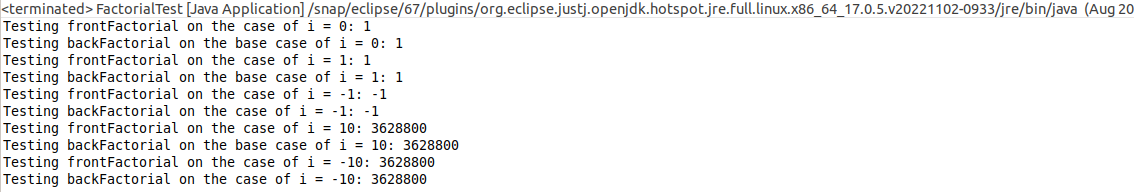
Exercise 4 screenshots:



Exercise 4 eval:

As this touches every number between 0 – n ( or 0 - -n), the evaluation goes as follows:

t(0) = 1, both methods will run exactly 1 time if 0 is passed into either one.

As these methods can process either positive or negative numbers and get the factorial,

t(n) = 1 + t(n-1) for any value greater than 0

t(n) = 1 + t(n+1) for any value less than 0

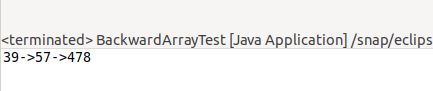
Further evaluation takes us to:

t(n) = 1 + (1 + t(n-2)) for positive or

t(n) = 1 + (1 + t(n+2)) for negative

Both of these will evaluate to t(n) = n+1 for any absolute value of n greater than 0, which will make it O(n) complexity. It can be simplified to remove the +1 by making another base case each for positive and negative 1, but that would result in slightly less clear code.

Exercise 5 screenshot:



Exercise 5 eval:

As this method will touch every element between the start of the subset of the array and the end of the subset of the array, the evaluation is as follows

where d = end – start

t(d) is evaluated as:

t(0) = 1, for where the two indexes are the same

t(d) = 1 + t(n-1), where the end is greater than the start by any amount

t(d) would evaluate down to t(d) = d as each and every element in the subset would be touched. This is equivalent to t(n) = n, which is big O O(n)