

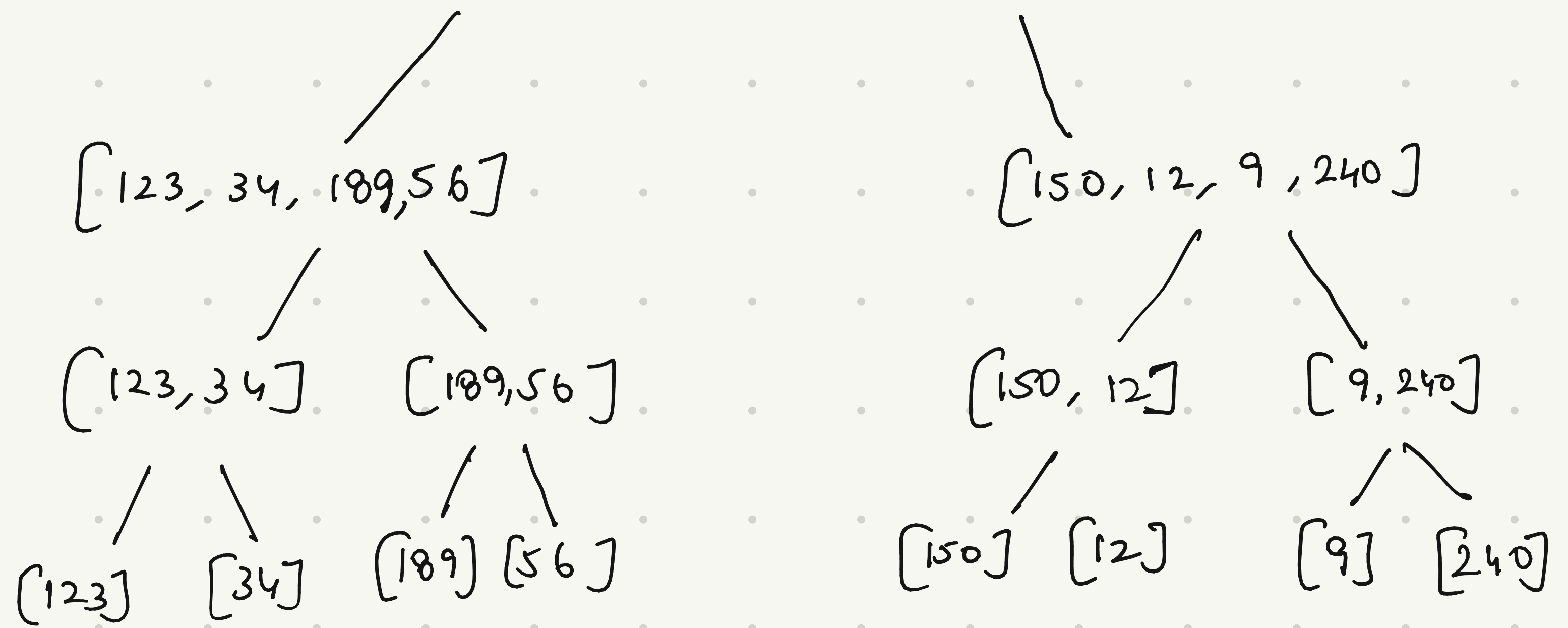
Homework - 2

1) Use the mergesort to sort the following list. Show actions step-by-step.

$[123, 34, 189, 56, 150, 12, 9, 240]$

→ First we split the array until we have only one element in each.

$[123, 34, 189, 56, 150, 12, 9, 240]$



→ After splitting, we merge these subarrays back into sorted subarray.

• Merging subarray in pairs to form subarray of 2 elements.

$[123] [34] \rightarrow [34, 123]$

$[189] [56] \rightarrow [56, 189]$

$[150] [12] \rightarrow [12, 150]$

$[9] [240] \rightarrow [9, 240]$

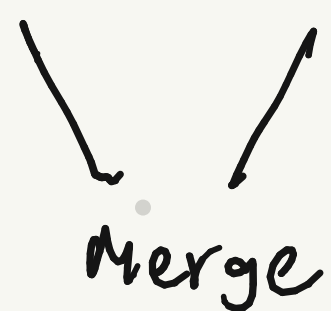
- Merge above sorted subarrays of 2 into ^{sorted} subarrays of 4.

$$[34, 123] \quad [56, 189] \rightarrow [34, 56, 123, 189]$$

$$[12, 150] \quad [9, 240] \rightarrow [9, 12, 150, 240]$$

- Merge above sorted subarrays of 4 into one final sorted array.

$$[34, 56, 123, 189] \quad [9, 12, 150, 240]$$


 Merge

$$[9, 12, 34, 56, 123, 150, 189, 240] \text{ — sorted.}$$

2) Give the tree of recursive calls in above exercise 1.

In each iteration, we split the array into subarray and again call the same MergeSort function on it. If we assume the MergeSort function as $f(n, arr)$ where 'n' is size of the array 'arr'.

$f(n, arr) \{$

$$h = n//2, \quad m = n - h$$

$$subarr1 = u[1 \dots h], \quad subarr2 = v[1 \dots m]$$

$$\text{copy } arr[1 \dots h] \Rightarrow u[1 \dots h]$$

$$arr[h+1 \dots n] \Rightarrow v[1 \dots m]$$

$$f(h, u)$$

$$f(m, v)$$

$$\text{merge}(h, u, m, v, s)$$

$\}$

arr = [123, 34, 189, 56, 150, 12, 9, 240]

n = 8

f(8, [123, 34, 189, 56, 150, 12, 9, 240])

f(4, [123, 34, 189, 56])

f(4, [150, 12, 9, 240])

f(2, [123, 34])

f(2, [189, 56])

f(2, [150, 12])

f(2, [9, 240])

f(1, [123])

f(1, [34])

f(1, [189])

f(1, [56])

f(1, [150])

f(1, [12])

f(1, [9])

f(1, [240])

3) $n! = 1 \cdot 2 \cdot 3 \dots n$

dca-fact(p, q) {

if $p > q$:

return 1

if $p == q$:

return p

m = (p + q) // 2

return dca-fact(p, m) * dca-fact(m + 1, q)

}

→ Constant time C

dca-fact(p = 1, q = n) → computes $n!$

Input size :

$n = q - p + 1$

Worst case scenario Time Complexity:

$$p=1, q=n$$

$$m = \frac{2+p}{2}$$

$$W(n) = W(2-p+1)$$

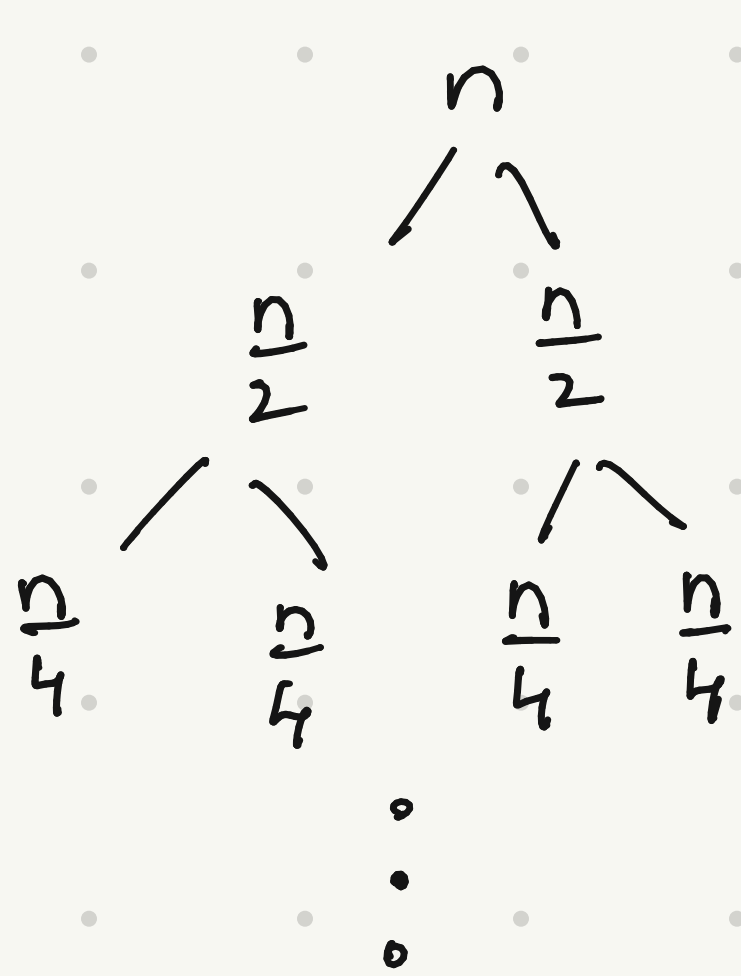
$$\text{dca-fact}(p, m) \rightarrow W\left(\frac{2-p}{2} + 1\right) \approx W(n/2)$$

$$\text{dca-fact}(m+1, q) \rightarrow W\left(\frac{2-p}{2}\right) \approx W\left(\frac{n}{2}\right)$$

$$W(n) = W(n/2) + W(n/2) + \underline{C}$$

↳ constant for comparisons

$$W(n) = 2W\left(\frac{n}{2}\right) + C$$



log n

$$1 + 2 + 2^2 + \dots + 2^{\log_2 n}$$

$$= 2^{(\log_2 n + 1)} - 1$$

$$= 2n - 1$$

$$W(n) = 2n - 1 = O(n) \text{ linear time}$$

∴ Time complexity is linear but not exponential.
 $O(n)$