COMP 301 Analysis of Algorithms
Instructor: Zafer Aydın
Lab Assignment 10

Introduction

In this lab you will compare the running times and RAM usage of quicksort, heap sort and merge sort algorithms. Submit your answers to the questions below in a text file (e.g. Word document). Name your file in name_surname.docx format. Submit your solution document and Java codes to Canvas.

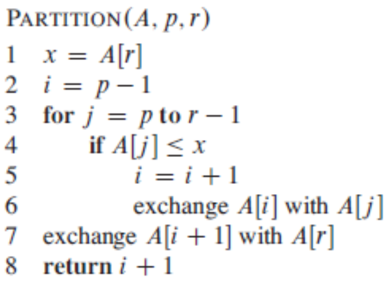
You can use the code templates in `quick.java` in this lab.

Problem Statement

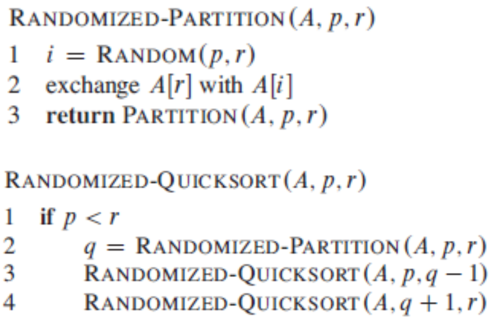
Given an array of integers sort the numbers in this array in ascending order.

Assignment

1. (a) Implement the Java methods for the quick sort algorithm given below.



```
PARTITION(A, p, r)
  $x = A[r]$
  $i = p - 1$
  for $j = p$ to $r - 1$
      if $A[j] \le x$
          $i = i + 1$
          exchange $A[i]$ with $A[j]$
  exchange $A[i + 1]$ with $A[r]$
  return $i + 1$
```



```
RANDOMIZED-PARTITION(A, p, r)
  $i = $ RANDOM$(p, r)$
  exchange $A[r]$ with $A[i]$
  return PARTITION$(A, p, r)$

RANDOMIZED-QUICKSORT(A, p, r)
  if $p < r$
      $q = $ RANDOMIZED-PARTITION$(A, p, r)$
      RANDOMIZED-QUICKSORT$(A, p, q - 1)$
      RANDOMIZED-QUICKSORT$(A, q + 1, r)$
```

(b) Test your algorithm by choosing an array of size 10. Initialize your array by random numbers from 0 to 99. Make sure your program sorts the array correctly. Include the output of your program for this sample input in your report.

(c) Choose input sizes in the table below, which are powers of 4, and initialize the values in your array by random numbers from 0 to 99. Compute the running times of quick sort, heap sort and merge sort in nanoseconds for each of these input sizes and include them to the table below. Write a for loop that performs these operations automatically. Do not run them one at a time.

b-ans)

```
Before Quicksort
[56, 45, 18, 22, 28, 83, 89, 66, 34, 9]
After Quicksort
[9, 18, 22, 28, 34, 45, 56, 66, 83, 89]
```

COMP 301 Analysis of Algorithms
Instructor: Zafer Aydın
Lab Assignment 10

Input size	Quick sort running time	Heap sort running time	Merge sort running time
4	6830	4676	10586
64	27487	19708	20698
256	492854	64722	1889098
1024	2785417	348149	365707
4096	1211552	854769	990221
16384	3997365	2802205	2802402
65536	28629423	9402765	9848537
262144	364650526	29017920	36177320
1048576	5495060291	117932721	130838526
4194304	104450413988	525082053	534034829
16777216	1664137250547	2273295935	2270097766
67108864	Couldnt run	Couldnt run	Couldnt run

(d) Set the input size to 67108864. Run quick sort, heap sort and merge sort for this input size. Open a terminal window and type top. Find the processes for the sorting algorithm you executed and record the RAM usage in MEM column. Include the RAM usage of these algorithms into the table below. Compare and comment on the RAM usage of these sorting algorithms.

Input size	Quick sort RAM	Heap sort RAM	Merge sort RAM
67108864	788	780	2160

d-ans)
As we expected, mergesort used more ram than other two sort because it initializes new arrays in order to sort actual array. quicksort and heapsort have similar ram usages and it is expected because these two method uses memory in that space.