COMP 301 Analysis of Algorithms
Instructor: Zafer Aydın
Lab Assignment 11

Introduction

In this lab you will compute the running times and RAM usage of the linear sorting algorithm counting sort and compare it to comparison sort algorithms such as merge sort and heap sort. 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 `linear.java` in this lab.

Problem Statement

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

Assignment

1. (a) Implement a Java method for the counting sort algorithm given below. Be careful with the indices such that indices of array C start from 0 while those of arrays A and B start from 1. You may need to do the necessary adjustments in the indexing of these arrays.

```
COUNTING-SORT(A, B, k)
 1  let C[0..k] be a new array
 2  for i = 0 to k
 3      C[i] = 0
 4  for j = 1 to A.length
 5      C[A[j]] = C[A[j]] + 1
 6  // C[i] now contains the number of elements equal to i.
 7  for i = 1 to k
 8      C[i] = C[i] + C[i − 1]
 9  // C[i] now contains the number of elements less than or equal to i.
10  for j = A.length downto 1
11      B[C[A[j]]] = A[j]
12      C[A[j]] = C[A[j]] − 1
```

(b) Test your algorithm by choosing an array of size 10. Initialize your array by random numbers from 0 to 9. 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 multiples of 10, and initialize the

values in your array by random numbers from 0 to array_size-1. Compute the running time of counting sort, merge sort and heap sort in nanoseconds for each of these input sizes and include them to the table below. The codes for merge sort and heap sort are available in the code template. Write a for loop that performs these operations automatically. Do not run them one at a time.

Input size	Input range	Counting sort running time	Merge sort running time	Heap sort running time
10	0-9			
100	0-99			
1000	0-999			
10000	0-9999			
100000	0-99999			
1000000	0-999999			
10000000	0-9999999			
100000000	0-99999999			

Which algorithm performs best at which input size?

(d) Now set the input range to 0-9999999 and fill the table below by repeating the analysis you made in part (c).

Input size	Input range	Counting sort running time	Merge sort running time	Heap sort running time
10	0-9999999			
100	0-9999999			
1000	0-9999999			
10000	0-9999999			

100000	0-9999999			

Which algorithm performs best at which input size?

(e) Set the input size to 100000000 and the input range to 0-9999999. Run count sort, merge sort and heap sort one at a time 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	Input range	Counting sort RAM	Merge sort RAM	Heap sort RAM
100000000	0-9999999			