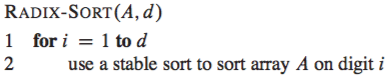
1. (a) Implement a Java method for radix sort algorithm given below. Note that you can start from the least significant digit and proceed from right to left until and including the most significant digit. You can use counting sort as the stable sorting algorithm. You can assume that the inputs are integers and separate the digits of the input numbers in order to process the digits one at a time (from the least to most significant digit). You can use arrays (1D or 2D) to store the digits of the inputs and to perform sorting.



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

[12, 18, 48, 75, 64, 32, 40, 51, 11, 80]

[11, 12, 18, 32, 40, 48, 51, 64, 75, 80]

(c) Choose input sizes in the table below, which are multiples of 10, and initialize the values in your array by random numbers from 100 to 999 (i.e. all numbers have three digits). Compute the running time of radix sort, 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 | Radix sort running time | Counting sort running time | Merge sort running time | Heap sort running time |
| 10 | 86400 | 3700 | 119600 | 353300 |
| 100 | 242200 | 8700 | 772700 | 1240800 |
| 1000 | 1160200 | 83200 | 3513900 | 1092200 |
| 10000 | 6047500 | 6165300 | 7218000 | 4940900 |
| 100000 | 36032700 | 5593500 | 45469000 | 46107500 |
| 1000000 | 71332700 | 58706600 | 222551700 | 341386900 |
| 10000000 | 374521500 | 165395100 | 1750115400 | 2756430700 |

Which algorithm performs best at which input size?

Counting sort at 10 input size.

(d) Set the input size to 100000000 and the input range to 100-999 (i.e all numbers have three digits). Run radix sort, 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 | Radix sort RAM | Counting sort RAM | Merge sort RAM | Heap sort RAM |
| 100000000 | 100-999 | 1 634 265 | 1 553 267 | 1 934 428 | 1 389 216 |