CSc 130 Assignment 4

Performance Evaluation of Heapsort, Mergesort, and Quicksort

Dr. Jinsong Ouyang

1 Requirements

1. Create a Java class with heapSort, mergeSort, and qucikSort methods.
2. Design and implement a driver (the main method) that does the following:
   (a) Create an array contains a list of 500000 integers from 0 to 499999 sequentially.
   (b) Randomly shuffle the array so that the values in the array are not in order.
   (c) Copy the array to 3 other arrays used as part of input parameters for heapsort, mergesort, and quicksort respectively.
   (d) Enter a forever while loop to do the following:
      i. Collect the first timestamp, call method heapSort to sort its input array, then collect the 2nd timestamp, compute the Δ of the two timestamps, and display ”It took Δ msecs to heapsort the array”.
      ii. Collect the first timestamp, call method mergeSort to sort its input array, then collect the 2nd timestamp, compute the Δ of the two timestamps, and display ”It took Δ msecs to mergeSort the array”.
      iii. Collect the first timestamp, call method quickSort to sort its input array, then collect the 2nd timestamp, compute the Δ of the two timestamps, and display ”It took Δ msecs to quickSort the array”.
      iv. Prompt user to enter “Press any key to start next round”.
      v. Copy the original shuffled array to the 3 input arrays for heapsort, mergesort, and quicksort.

Table 1: Performance Measurement

	Round 1	Round 2	Round 3	Round 4	Round 5	Average
Heapsort						
Mergesort						
Quicksort						

2 Deliverables

1. Source code
2. Performance evaluation. Run your program a number of times and record the running times for each round. Represent in a table the performance results by the 3 sorting algorithms, then compare the differences and explain possible causes to the difference in running times.