Assignment 4

Due Nov 14 at 11:59 pm

1. (5 points) In Clojure, implement Quicksort. Do so in a function called quicksort with one argument, a vector of numbers. The function returns a sequence of sorted numbers. For example (quicksort [2 6 1 9 1]) returns (1 1 2 6 9).

2. (5 points) In Clojure, implement Mergesort. Do so in a function called mergesort with one argument, a vector of numbers. The function returns a sequence of sorted numbers. For example (mergesort [2 6 1 9 1]) returns (1 1 2 6 9).

3. (5 points) Test the two functions to ensure they sort correctly.

4. (5 points) Implement Quicksort and Mergesort in any other language and make sure they sort correctly.

5. (5 points) In both languages, time your quicksort and merge sort functions on a sorted list of 1,000,000 numbers. Time them on a reverse sorted list of 1,000,000 numbers. If 1,000,000 numbers are not enough to see a difference between the two algorithms, increase the input size.

6. (15 points) Evaluate your results. Which program was faster on the sorted input? Was it also faster on the input that was in reverse sorted order? Is the result the same in both languages? Explain your results. Were the results as expected? Can you say anything about the languages used for sorting in terms of performance and ease of implementation?

7. (Extra Credit 5 points) Compare the performance of your implementations in both languages with the performance of the built-in sort function in each language. Explain the result.

Late Policy

An assignment turned in 1-7 days late will lose 5% of the total value of the assignment per day late. The eight-day late penalty will be 40% of the assignment, the ninth-day late penalty will be 60%, and after the ninth day late, the penalty will be 90%. Once a solution to an assignment has been posted or discussed in class, the assignment will no longer be accepted. Late penalties are always rounded up to the next integer value.