16:332:573

Homework #2 (100pt), Due. 03-07-2017

Q1. [20pt] **Recursion:** We discussed how recursions are used for handling computer programming languages. Recall findTheEndOfPrefix() function which takes a string as input and returns the end of the legal prefix expressions. Our grammar looks as follows.

< prefix > = < identifier > | < operator > < prefix > < prefix >

< operator > = + | - | * | /

Implement a findTheEndOfPrefix() function. Provide test inputs and print the result to stdout.

Q2. [20pt] **Backtracking**: Recall the airline route search problem discussed in the class. Implement two version of algorithms – one based on recursion and the other based on stacks. Create an input text file based on the lecture slide (p111-112 of sp17-2-Basic*.pdf). Compare these two implementations. To get a full credit, your implementation needs to generate traces, i.e., enumerate the search process by showing a sequence of visited nodes.

Q3. [20pt] **Mergesort**: Implement the two versions of Mergesort that we discussed in the class. Create a plot for the total number of comparisons to sort the data using attached data for both cases. Explain the results.

Q4. [20pt] **Mergesort Best Case**: In the class, we saw in the table that the best case complexity for mergesort (in terms of # of comparisons) is $\frac{1}{2}NlogN$. Find out and describe when the mergesort will show the best case performance and prove the bound (either by picture or induction as in the class).

Q5. [20pt] **Quicksort:** We discussed quicksort algorithm in class which was a recursive algorithm. Additionally implement an iterative version of quicksort algorithm and compare the performance with the recursive implementation.

Note 1) You need to provide big enough input to your program (over million is recommended).

Note 2) You need to implement both algorithms. For recursive version, it is ok to be based on the code in the lecture slide.