CS303–Spring, 2019 Programming Assignment 2 Out: Oct 27, 2020
Due: Nov. 12, 2020

Homework will be due at the **BEGINNING** *of class on the date due. Your answers should be written legibly if handwritten and your name should be clearly written on your homework assignment. Try to make your answers as clear and concise as possible; style will count in your overall mark. Be sure to read and know the collaboration policy in the course syllabus.*

Assignments are expected to be turned following these requirements:

- ***Submit via blackboard with a zip file containing your code, and a SEPARATE pdf file containing your writeup. Use your name as the zip file name. Try to make sure your code can compile and run, you will lose most of the scores if your code can not run. If there are bugs that you can't fix, try to address them in your write-up. If you prefer to do late submission and take the penalty, please inform TA and professor before the deadline.***
- ***Especially if you write your code in Java or C/C++, make sure your code compiles and include test cases in your code. The output of your code should include the input and the output for those test cases.***
- *Try to answer the question 2 the question 3 with details and show your test cases(output screenshot) for these algorithms in the write-up.*

Problems In this assignment you are asked to implement three different algorithms for median finding, or more generally for selection of the k'th smallest element in an unordered array. You will then be asked to compare these algorithms. The three algorithms have the same input and output:

Input: an unordered array $A[1, ..., n]$ and a number k such that $1 <= k <= n$

Output: the k-th smallest element in A.

For example, the output of $A = [9, 14, 9, 5, 10, 6, 15, 6, 13, 9]$ and $k = 5$ should be 9. Please test your code and cover corner cases.

1. **Algorithm 1: Randomized selection(10pts)**

   This is the algorithm we cover in class. Notes in Chapter 13.5(page 728 -729) of the textbook.

2. **Algorithm 2: Deterministic selection (median of medians, 20pts)**

   Given a Although the randomized selection algorithm runs in time $O(n)$ in expectation, it is actually possible to design a deterministic algorithm for selection which also runs in time $O(n)$. The main difference between the deterministic selection algorithm and the randomized selection algorithm in Section 13.5 of the textbook is how to choose the pivot. In Algorithm 2, divide A into groups of 5 elements (except the last group), then compute the medians $M[1, .., n/5]$ of these groups using sorting (which takes $O(1)$ per group), then use the selection algorithm recursively to compute the median of $M[1, .., n/5]$ and use this value as the pivot to divide A into Sl, Sv, and Sr as before. When the size of input A is not too large (e.g., no more than 10) you are allowed to use any sorting algorithm (quicksort, mergesort, bubblesort, etc.). If you need more details, feel free to look at the following Lecture Notes in `https://www.ics.uci.edu/~eppstein/161/960130.html`.

3. **Algorithm 3: Sorting(20pts)**

   Implement Quicksort (page 731-732 of textbook), then use it to sort A, and return the $k'th$ smallest element. Note that most of the code for Quicksort should be very similar to your Algorithm 1. Quicksort runs in expected $O(nlogn)$ time.

   - Show outputs(your code screenshots) for [7, 2, 4, 6, 9, 11, 2, 6, 10, 6, 15, 6, 14, 2, 7, 5, 13, 9, 12, 15], $k = 10$.
   - Generate a random list of $n = 10^7$ integers, with each random integer being uniformly distributed between 0 and n/100. Use your three algorithms to compute the median of this list (the n/2 smallest element), and time the running time of each algorithm. Depending on your machine and implementation, this may take a few minutes; we suggest starting with smaller n first to make sure everything is working properly.

     Which algorithm performs the best? What if you make the list size n even larger?