CS2040C Semester 1 2020/2021

Data Structures and Algorithms

Tutorial 02 - Sorting, ADT, List

For Week 04

Document is last modified on: September 2, 2020

1 Introduction and Objective

In this tutorial, we will discuss various Sorting algorithms and the concept of Abstract Data Type (ADT) that will appear several times throughout this course via discussion of List ADT.

You are encouraged to review e-Lecture of `https://visualgo.net/en/sorting?slide=1` (to the last slide 18-5) and `https://visualgo.net/en/list?slide=2` (to the last slide 9-5) for this tutorial.

2 Tutorial 02 Questions

Sorting

Q1). At `https://visualgo.net/en/sorting?slide=1-2`, Steven mentions a few array applications that become easier/more efficient if the underlying array is sorted. In this tutorial, we will quickly discuss application *a few* of these application 1-6 in algorithmic level only as many of these are asked in PS1 A (/basicprogramming2).

Sorting, Mini Experiment

Q2). Please use the 'Exploration Mode' of `https://visualgo.net/en/sorting` to complete the following table. You can use 'Create' menu to create input array of various types. You will need to fully understand the individual strengths and weaknesses of each sorting algorithms discussed in class in order to be able to complete this mini experiment properly.

For example, on random input, Optimized (Opt) Bubble Sort that stops as soon as there is no more swap inside the inner loop runs in $O(N^2)$ but if we are given an ascending numbers as input, that Optimized Bubble Sort can terminate very fast, i.e. in $O(N)$.

Note that the (Rand) Quick Sort in VisuAlgo as implemented currently (Aug 2019) has issue with input where all input numbers are equal.

Input type →			Sorted		Nearly Sorted	
↓ Algorithm	Random	Equal	Ascending	Descending	Ascending	Descending
(Opt) Bubble Sort	$O(N^2)$		$O(N)$			
(Min) Selection Sort						$O(N^2)$
Insertion Sort			$O(N)$			
Merge Sort					$O(N \log N)$	
Quick Sort		$O(N^2)$				
(Rand) Quick Sort	$O(N \log N)$	$O(N^2)$				
Counting Sort						$O(N)$
Radix Sort			$O(N)$			

Finding k-th Smallest Element in an Unsorted Array (Selection Algorithm)

Q3). We will revisit the concept of QuickSort's partitioning algorithm (please review `https://visualgo.net/en/sorting?slide=11-2` to slide 11-7) and combine it with a binary search-like algorithm on an unsorted array to find the k-th smallest element in the array *efficiently*. In this tutorial, we will spend some time discussing the details. Your job before attending this tutorial is to read this algorithm from the Internet: `http://en.wikipedia.org/wiki/Quickselect`, or if you have Competitive Programming 4 (or earlier) book, read about it in Section 2.3.4 (the earlier part of Order Statistics Tree).

Abstract Data Type (ADT)

Q4). Please self-read List ADT and its common operations: `https://visualgo.net/en/list?slide=2-1`. Now compare it with the sample (fixed-size array) implementation discussed back in last Tutorial Tut01 and if we replace that fixed-size array with STL vector implementation instead. What are the concepts of ADT that you have understood by now? We will discuss List ADT in more details in Lecture when we discuss 3 related ADTs: Stack, Queue, Deque.

Hands-on 2

TA will run the second half of this session with another chosen Kattis problem involving sorting.

Problem Set 1

We will end the tutorial with discussion of PS1 B (/magicsequence), up to 79 points, in HIGH level. Recall that PS1 A (/basicprogramming2) is part of Q1 of this week's tutorial.