

CS2040C Semester 1 2018/2019
Data Structures and Algorithms

Tutorial 02 - Sorting, ADT, List

For Week 4 (Week Starting 3rd September 2018)

Document was last modified on: August 28, 2018

1 Introduction and Objectives

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

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> (that slide 2 only, i.e. 2, 2-1, to 2.8) for this tutorial.

2 Tutorial 02 Questions

Sorting (part 2)











Q1). At <https://visualgo.net/en/sorting?slide=1-2>, Visualgo describes a few array applications that become easier/more efficient if the underlying array is sorted. In this tutorial, we will quickly discuss application 1-4 (if not completed during the last tutorial) and focus on application 5-6.

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 algorithm 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 $O(N)$.

You may assume ”nearly sorted” is a sorted input array with only a few elements that are not in their correct positions. (Eg: 1, 2, 3, 9, 5, 6, 7, 4, 8, 10)

Input type → ↓ Algorithm	Random	Sorted		Nearly Sorted	
		Ascending	Descending	Ascending	Descending
(Opt) Bubble Sort	$O(N^2)$	$O(N)$ - best			
(Min) Selection Sort					$O(N^2)$
Insertion Sort			$O(N^2)$		
 Merge Sort				$O(N \log N)$	
 Quick Sort		$O(N^2)$			
(Rand) Quick Sort	$O(N \log N)$				
 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 3 (or 3.17/b) book, read about it in Chapter 9.

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 tutorial Tut01. 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.

Problem Set 1

We will end the tutorial with discussion of PS1. Just a reminder, PS1 is due on 8 September 2018, 8am.