This is an answer guideline, not the actual answer key. If this document is leaked to other people without my consent, no further guidelines will be posted from the next DG onwards.
- created by Yu Jie

CS2040S: Data Structures and Algorithms

Discussion Group Problems for Week 3

For: Aug 26–Aug. 30, 2019

Problem 1. Sorting Jumble.

The first column in the table below contains an unsorted list of countries (which may or may not exist). The last column contains a sorted list of countries. Each intermediate column contains a partially sorted list of countries. Each intermediate column was constructed by beginning with the unsorted list at the left and running one of the sorting algorithms (BubbleSort, InsertionSort, or MergeSort), stopping at some point before it finishes. One column has been sorted using a sorting algorithm that you have not seen in class.

Identify which column was (partially) sorted with which algorithm. *Hint: Do not just execute each sorting algorithm, step-by-step, until it matches one of the columns. Instead, think about the invariants that are true at every step of the sorting algorithm.*

Unsorted	**A**	**B**	**C**	**D**	**Sorted**
Estonia	Australia	Australia	Estonia	China	Australia
China	China	China	China	Estonia	China
Oman	Estonia	Estonia	Australia	Australia	Estonia
Singapore	Oman	Laos	Germany	Laos	Finland
Zambia	Singapore	Oman	Hungary	Malaysia	Germany
Vietnam	Vietnam	Singapore	Finland	Finland	Hungary
Australia	Zambia	Vietnam	Oman	Oman	Iceland
Laos	Laos	Zambia	Iceland	Germany	Laos
Malaysia	Malaysia	Finland	Malaysia	Philippines	Malaysia
Finland	Finland	Germany	Vietnam	Hungary	Oman
Philippines	Philippines	Hungary	Philippines	Qatar	Philippines
Germany	Germany	Iceland	Laos	Iceland	Qatar
Thailand	Thailand	Malaysia	Thailand	Singapore	Singapore
Hungary	Hungary	Philippines	Zambia	Thailand	Thailand
Qatar	Qatar	Qatar	Qatar	Vietnam	Vietnam
Iceland	Iceland	Thailand	Singapore	Zambia	Zambia
Unsorted	**A**	**B**	**C**	**D**	**Sorted**

Please write the proper number in the blank space beside the letter:

A _ 2

B _ 3

C _ 4

D _ 1

1. BubbleSort Bubble-up
2. InsertionSort Relative Positioning is the same
3. MergeSort (top down, sorts top half before bottom half) Sorts by category
4. None of the above

Problem 2. Guess the Number
Next, we will attempt the *Guess the Number* problem on Kattis:
`https://nus.kattis.com/problems/guess`.
Replicated here:

Guess the Number

I am thinking of a number between 1 and 1000, can you guess what number it is? Given a guess, I will tell you whether the guess is too low, too high, or correct. But I will only give you 10 guesses, so use them wisely!

Interaction

Your program should output guesses for the correct number, in the form of an integer between 1 and 1000 on a line on its own. After making each guess, you need to make sure to flush standard out.

After each guess, there will be a response to be read from standard in. This response is a line with one of the following three words:

- "`lower`" if the number I am thinking of is lower than your guess
- "`higher`" if the number I am thinking of is higher than your guess
- "`correct`" if your guess is correct

After having guessed the right answer your program should exit. If you guess incorrectly 10 times, you wont get any more chances and your program will be terminated.

Hint : Binary Search :)

If still cannot solve, text me k.

Problem 3. Extra: Sorting Dlorah's movie collection

Dlorah Hos (the evil CS2040S lecturer from the mirror universe) has a lot of (pirated) movies on her computer. She wants to write a sorting algorithm to sort the movies by name. Since the movies are very large (8K Ultra HD!), moving the movies on disk is expensive. By contrast, comparing two movies to determine which should come first is relatively cheap, since it only requires examining the title of each movie.

Which sorting algorithm should Dlorah use? Justify your answer.

As mentioned during DG, Comparisons != Swaps.

In this case, comparing is much "cheaper" than swapping, so the "best "algorithm to use is Selection Sort.

Post Answer Questions :
1) What is the worst case swap scenario of other algorithms such as MergeSort? n^2?
2) Why do computers actually use QuickSort though?