**CSCI C343 - Data Structures**

**Summer 2017**

# Homework 06

## Task B Sample Solutions

### Work Policy

Homework 06 is to be done **individually** - no group solutions or cooperative work.

# Tasks

In a text editor, prepare a plain-text file named `hw06`*`username`*`TaskA.txt`, where you will annotate your work for this Homework 06.
Note: for editing plain-text files, we recommend that you use `Notepad++` on Windows systems, `TextWrangler` on Mac systems, and emacs or vim on Linux systems.

B. **Written Tasks**
[50 points]
- Read about the Mergesort algorithm in the textbook Chapter 7.4 (pages 233-236).
- [20 points]
  Sort the number sequence [8, 1, 4, 1, 5, 9, 2, 6, 5] by using:
  a. Insertion sort
  b. Mergesort
  for full points, you need to write down the number sequence after each step in the algorithm, separately for (a.) and (b.) above

  **Answer**:

  *Insertion sort* steps:

```
8, 1, 4, 1, 5, 9, 2, 6, 5
1, 8, 4, 1, 5, 9, 2, 6, 5
1, 4, 8, 1, 5, 9, 2, 6, 5
1, 1, 4, 8, 5, 9, 2, 6, 5
1, 1, 4, 5, 8, 9, 2, 6, 5
1, 1, 4, 5, 8, 9, 2, 6, 5
1, 1, 2, 4, 5, 8, 9, 6, 5
1, 1, 2, 4, 5, 6, 8, 9, 5
1, 1, 2, 4, 5, 5, 6, 8, 9
```

  *Mergesort* steps:

First the number sub-sequence 8, 1, 4, 1 is *recursively sorted* as 1, 1, 4, 8.
(etc.)
Then the number sub-sequence 5, 9, 2, 6, 5 is *recursively sorted* as 2, 5, 5, 6, 9.
(etc.)
Then the results from sorting the two sub-sequences is *merged* into the output (sorted) number sequence.

- [10 points]
  What does it mean for a sorting algorithm to be *stable*?

  **Answer**: A sorting algorithm is defined as *stable*, if any two elements with equal keys will maintain their original order (from the input sequence) after sorting is completed (in the output sequence).

- [20 points]
  For each sorting algorithm in the list, write down whether is it stable, and why it is/isn't stable:
  a. Insertion sort
  b. Bubble sort
  c. Mergesort
  d. Heapsort

  **Answer**: We assume that comparison operations used in the sorting algorithm (e.g. *testing for equality*) don't destroy the order.
  *Insertion sort*, *bubble sort*, and *mergesort* are therefore **stable**, because they rely on comparisons for reordering elements.

  *Heapsort* is **not stable**:
  for example, assume two items in the heapsort input data have identical keys. After heapsort is completed, those two items may end up in a different mutual order than the input sequence's original order.
  This may happen when one of those two identical-key items is compared against a third item with a different key: heapsort therefore does not maintain original ordering for identical keys, and is therefore not stable.

Please provide your answers to all Task A questions and exercises in a plain-text file named `hw06`*`username`*`TaskB.txt`, and push it to your personal IU GitHub C343 Summer 2017 repository under `hw/hw06/`.