San José State University
College of Science / Department of Computer Science
CS 146 Data Structures and Algorithms
Section 4/7, Spring 2015
Instructor: Dr. Angus Yeung

Assignment 4
Due Date: Wednesday, May 6, 11:59 pm

Submission:
1. Create a folder called `hw4` on your computer.
2. Create a WORD document (or any similar word processing document) and put all of your answers in the same document. Your word processing software must allow you to enter math equations.
3. Save the file as `hw4`, e.g., `hw4.doc` or `hw4.docx`. Export the Word document (or similar word processing document) to PDF file. For example, File -> Save As -> choose Format as "PDF".
4. Alternatively, you can write your answers on paper, scan and save your work as a PDF file.
5. Copy or move your PDF file, `hw4.pdf`, to folder `hw4`
6. For programming assignment, follow the instruction in the question to submit your `.java` files.
7. When you are ready for submitting your work, zip up the `hw4` folder and upload it to Assignment 4 on Canvas.

Only softcopy is acceptable. Do not hand in your solutions in hardcopy. Do not encrypt your zip file with a password.

Problems (Total: 100 Points):

Part A contains written assignment questions. Follow the instruction in each question carefully and show all of your work. (40 Points)

Part B contains both written and programming assignment questions. Follow the guidelines in each assignment for programming and submission requirements. (60 Points)

PART A Non-programming Questions (40 Points)

4.1 (5 Points) Given input $\{3, 1, 4, 1, 5, 9, 2, 6, 5\}$, sort the sequence using insertion sort and show each step in your answer.

4.2 (10 Points) Show the result of running Shellsort on the input $\{9, 8, 7, 6, 5, 4, 3, 2, 1\}$ using the increments $\{1, 3, 7\}$.

4.3 (10 Points) Sort $\{3, 1, 4, 1, 5, 9, 2, 6, 5, 3, 5\}$ using quicksort with median-of-three partitioning and a cutoff of 3.

4.4 (5 Points) Show the result of inserting keys 1 to 15 in order into an initially empty leftist heap.

4.5 (5 Points) Show the result of inserting keys 1 to 15 in order into an initially empty skew heap.

4.6 (5 Points) Merge the two skew heaps.
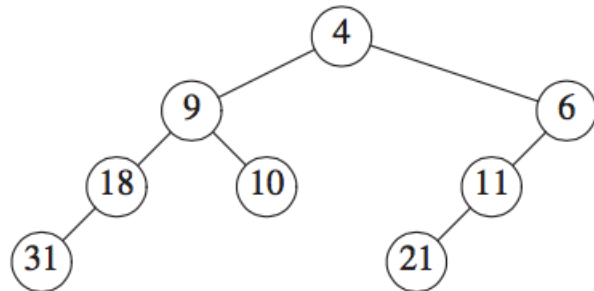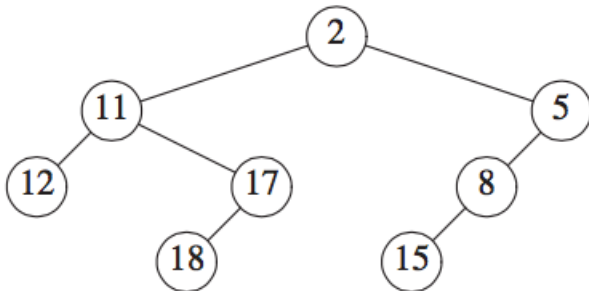
PART B Programming Questions. (60 Points)

4.7 (60 Points) Write a program that will sort arrays of 100,000 integers using the following six algorithms. Since all algorithms shall sort the same array, you are asked to make a copy of **the same array** each time before sorting. Also, your program shall randomize the input each time the program is executed.

There are three types of arrays for each sorting:
1. An randomized and unsorted array of integer numbers.
2. An sorted array of integer numbers, i.e., increasing order: 1, 2, 3, …, 998, 999.
3. An sorted array of integer numbers with reversed order, i.e., decreasing order: 999, 998, .., 3, 2, 1.

The sorting algorithms you are going to implement are:
1. **Quicksort**
2. **Mergesort**
3. **Heapsort**
4. **Shellsort** with suboptimal sequence,
   i.e., h starts at half the length of the array and is halved for each pass. Refer to Figure 7.4 in the textbook.
5. **Shellsort** with optimal sequence proposed by Sedgewick,
   i.e., $\{1, 5, 19, 41, 109,...\}$ Refer to page 278 of the textbook.
6. **Insertion Sort**

There will be 18 sorting results (3 types of input x 6 types of sorting algorithm). For each sorting result, your program should output:
1. How much time it took;
2. How many comparisons it made between two values;
3. How many moves it made of the values. (Whenever a value changes position, it counts as a move. For example, if you swap the positions of two values, it counts as two moves.

Your program is required to output the above 18 sorting results in a single table. As such, you may want to execute the 18 sorting runs sequentially in the main method. Do not include the operations involved in creating sorted input arrays in sorting results.

The output of sorting results shall be in this order: Quicksort, Mergesort, Heapsort, Shellsort, Insertion Sort. The result shall be displayed immediately once one algorithm has finished sorting one input array.

You are also required to document 2 runs of your program (2 x 18 sorting results in two tables) and submit your answer separately in PDF document.

How to Submit:
(1) Create the folder "sortAll" that contains all required .java file(s).
(2) The folder should be part of the hw4.zip file that you upload to Canvas.
(3) Do not include any other file types inside the sortAll folder except .java file(s).
(4) Do not declare and use any "package" in your .java file(s).
(5) Use "sortAll" as the class name – so you can run as %java sortAll. It is unacceptable if your program can run using IDE (e.g., Eclipse) but not from command line as required here.

There is one point penalty for each requirement that a student fails to follow (total penalty: 5 points).
Submit the two tables of sorting results together with your answers of written questions. (5 Points).