

1) Show the Lo/Hi/Mid Record for Binary Searching this array for the number 14.

0	1	2	3	4	5	6	7	8	9
0	2	3	5	8	8	10	12	14	20

2) Show the stages of selection, insertion, and merge sort for this array.

0	1	2	3	4	5	6	7	8	9
8	2	9	7	1	3	5	2	0	10

3) Run the Sorts Recognizer Program (http://fahrenbacher.com/webpage/ap-a/demos/old-demos/sort_recognizer/sorter.html) on my website under "Ancient Demos." See if you can figure out which sort is being demonstrated (merge sort will not appear, and the one that sorts in a way you've not learned is called Bubble Sort - you will not be quizzed on this).

4) Which sorts...

(a) Are stable?

(b) Have a consistent running time (don't vary much based on the numbers in the array)?

(c) Take a lot of memory?

(d) Are generally the fastest? (try this demo out to convince yourself: http://fahrenbacher.com/webpage/ap-a/demos/old-demos/sorting_demo/SortItem.html)

(e) Have $n-1$ stages?

(f) Have $\log_2(n)$ stages?

5) When should you use linear (sequential) search instead of binary search?

6) Which searching technique is best for sorted information (Not sure - try this out: http://fahrenbacher.com/webpage/ap-a/demos/old-demos/search_test/searcher.html)

7) If you need more practice with the sort algorithms, try out this applet: (<http://fahrenbacher.com/webpage/ap-a/demos/old-demos/fsort/fsort.html>) It asks you to drag numbers around to demonstrate each of the sorting algorithms (it will tell you if you moved the right thing - it's a little difficult to understand...).

8) Done with all this? Then I'm sure there's a project or two you need to be finishing up right now...