Exam 2 Notes

Matthew Sabo

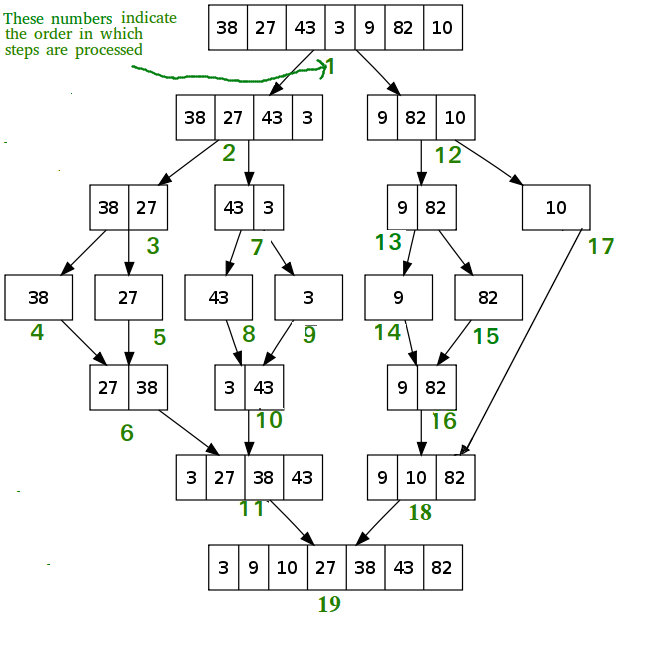
2/22/23

**Sorting Algorithms**

**Quick Sort:**

* A divide and conquer algorithm.
* Picks element as a pivot and partitions the given array around the pivot.
  + Can pick the first element, last element, random element, or median as pivot.
* 
* The efficiency of quick sort is O(n log n) in best case
  + Worst case is O(n^2) if bad pivot is chosen.

**Merge Sort:**

* Recursively divides the array into 2 halves, sort each half, then merge the sorted halves back together.
* Time complexity of O(n log n)
* 
* Stable
* Not adaptive

**Shell Sort:**

* Developed by Donald Shell in 1959
* Variation of Insertion Sort
* Able to swap items in list that are far away from each other rather than only being able to swap to the items right next to the element
* Step 1 − Start  
  Step 2 − Initialize the value of gap size. Example: h  
  Step 3 − Divide the list into smaller sub-part. Each must have equal intervals to h  
  Step 4 − Sort these sub-lists using insertion sort  
  Step 5 – Repeat this step 2 until the list is sorted.  
  Step 6 – Print a sorted list.  
  Step 7 – Stop.
* Best case time complexity is Ω (n log(n))
  + Average case is O(n\*log n)~O(n^1.25)
  + Worst Case is O(n^2)

**Heap Sort:**

* Sorts a set of elements by adding each one to a heap, then removing them one at a time
  + Smallest element comes off first so the sequence will be in ascending order.
* Time complexity: O(n log n)
* Not stable

**Recursion**

* Recursion is when something is called on itself
* Is recursive if:
  + Has a simple base case
    - A terminating scenario that does not use recursion to produce an answer
  + A recursive step
    - A set of rules that reduces all successive cases toward the base case.