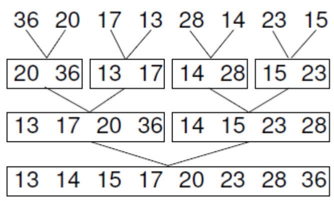
Study Questions for Final, Data Structures & Algorithms

You are encouraged to discuss questions and solutions with your classmates or others. I have no solutions to provide but can answer questions you may have. The questions here focus on material covered after midterm #2, but questions on the final could appear on topics from midterms 1 and 2 (but will emphasize the new material).

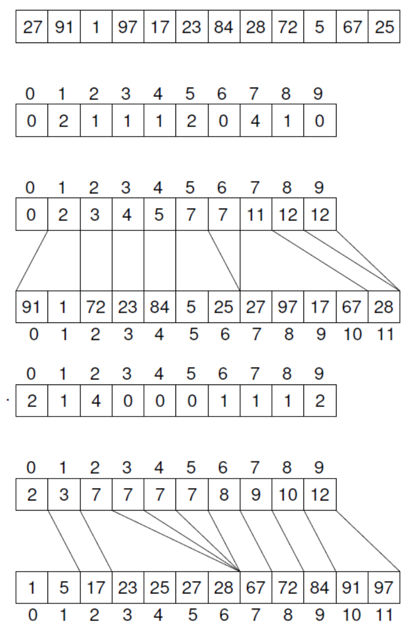
1. Sorting

a) Identify the sorting algorithms depicted below.

1)



2)



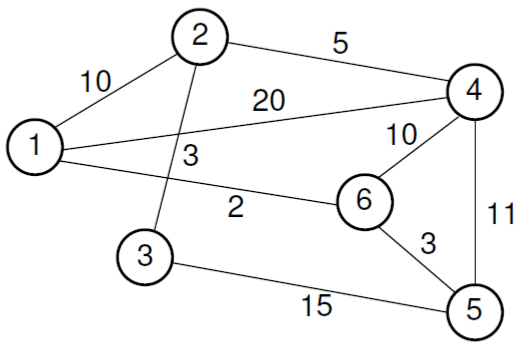
b) When implementing Insertion Sort, a binary search could be used to locate the position within the first i-1 elements of the array into which element i should be inserted. How would this affect the runtime of the algorithm?

c) You have a set of numbers with values in the range 0 to 30,000. There are no duplicates. Devise an efficient algorithn to sort these numbers that keeps memory requirements to a minimum.

d) Consider the version of quicksort that uses the first element in its range of values to sort as the pivot/partition element. Give an array of 8 elements that results in the worst-case performance for quicksort.

2. Graph Algorithms

Given the following graph:



a) Show the result of running BFS from vertex 2 (BFS ignores the weights)
b) Show the result of running DFS from vertex 2 (DFS ignores the weights)
c) Run Kruskal or Prim's MST algorithm (source = 2) on the graph and show the final MST
    a. Show the steps of the algorithm as opposed to just producing the final MST
d) Run Dijkstra's Algorithm with vertex 2 as the source and show the final parent/distance values for each vertex
    a. Show the steps of the algorithm as opposed to just producing the final result

e) Consider if a linked list is used to implement a set. For example, if Set 1 contains a,b,c and Set 2 contains d,e then the linked list could be: (1,a) → (1,b) → (2,d) → (2,e) → (1,c)

    There is no particular order among the entries, they are just there in the order they were inserted. We would need to search the linked list to perform set lookups, deletions, unions, etc.

    Give the runtime of Kruskal's algorithm if this data representation is used for implementing a set.

3. Hash Tables

a) Professor Krunk hypothesizes that he can obtain substantial performance gains by modifying the chaining scheme of a hash table so that each list is kept in sorted order. How does this modification affect the running time for successful searches, unsuccessful searches, insertions, and deletions?

b) Consider a hash function that multiplies the ASCII value of each character in a string together, then takes the modulus to fit the result into a hash table. Discuss if this is a good hash function or not, and if not, propose a better hash function.

c) If we use a hash table with linear probing to resolve collisions, what problems arise if we want to delete entries in the hash table?

4. Recurrence Relations

a) Give the recurrence relation for this function:

```
int algo2(int n)
{
        if (n == 0)
                return 1;
        return algo2(n-1) + algo2(n-1);
}
```

b) Solve the recurrence relation: T(n) = 2T(n/2) + n   where T(1) = 1