Problem Set 8, Part I

Problem 1: Hash tables

1-1) linear

0	ant
1	flea
2	bat
3	cat
4	goat
5	dog
6	bird
7	bison

overflow: duck

1-2) quadratic

0	
1	
2	
3	cat
4	goat
5	bird
6	bison
7	dog

ant

1-3) double hashing

0	ant
1	bat
2	flea
3	cat
4	goat
5	bison
6	bird
7	dog

duck

1-4) probe sequence:
3, 6, 1, 4, 7, 2

1-5) table after the insertion:

0	
1	leopard
2	
3	fly
4	toad
5	
6	cat
7	terrier

Problem 2: Comparing data structures

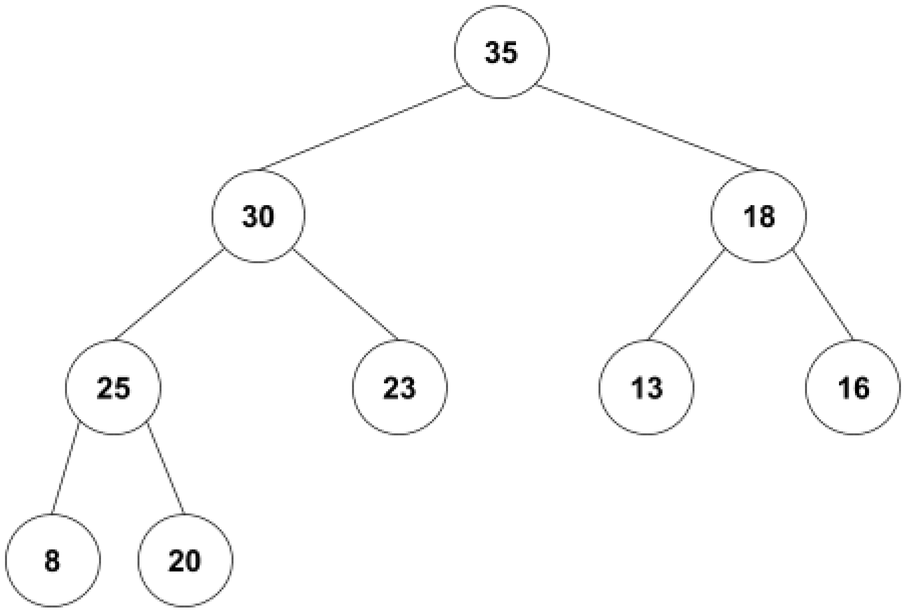
In this situation, both the balanced search tree and hash table can retrieve a record on the order of 20 operations per retrieval in a database of one million records ($\log_2 1{,}000{,}000=20$). Both can identify a product by specifying its name: for a balanced search tree, it can use keys; for a hash table, it can use double hashing, open addressing, or some form of probing. The biggest issue is the last point, which requests that the supermarket database be able to increase the size of the database - adding large sets of new records - without taking the system offline. This is where trees excel. Given a key, a simple insert can create a new node at O(log n) time. Hash tables, however, requires an entire rehashing of all existing values because items may end up in different positions in a larger table. Furthermore, the hash table overflow and lack of a consistent pattern when identifying products are more disadvantages toward this database.

Problem 3: Complete trees and arrays
3-1)
Left child of A is in position 101 = 2*50 + 1
Right child of A is in position 102 = 2*50 + 2
The parent of A is in position 24 = (50-2)/2

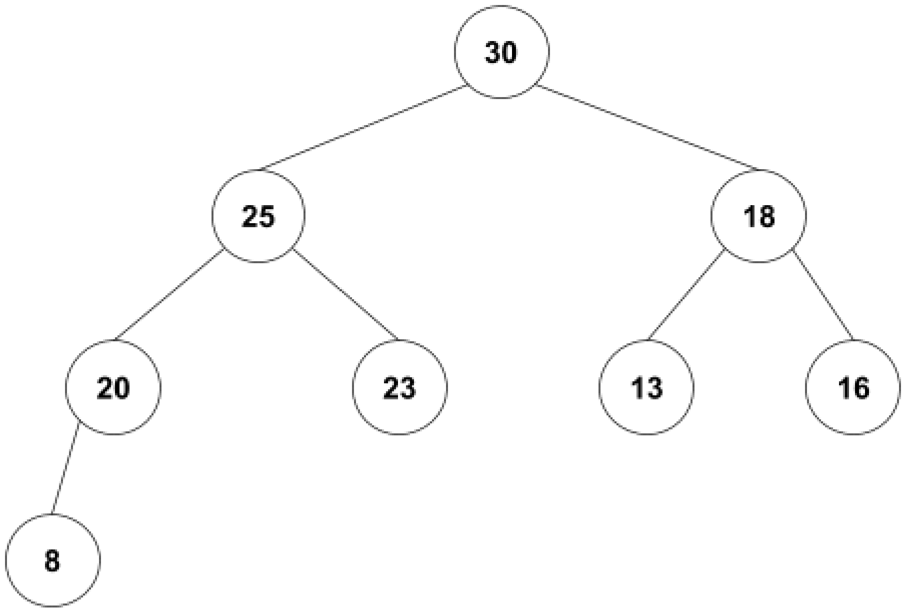
3-2)
A complete tree with 200 nodes has a height of 7. This is because the height of a complete tree can be found by $\log_2(n+1) - 1$.

3-3)
If we have 200 nodes, we have to realize that node 200 is actually in position 199 because the root node has an index of 0 instead of 1. Given that the last node is in 199, we know that all odd-numbered nodes are a left-child node, so the rightmost bottommost node is a left child of its parent.

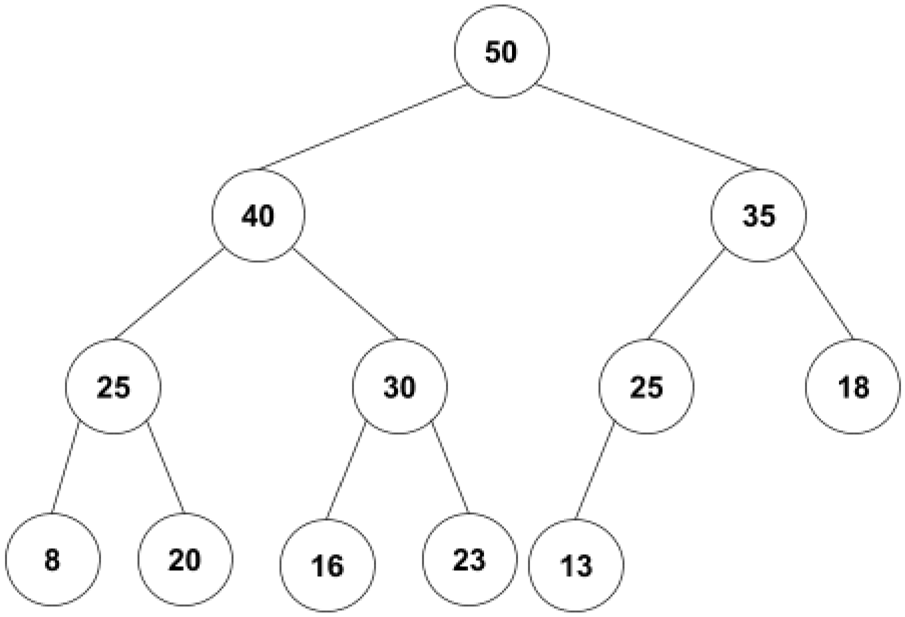
Problem 4: Heaps
4-1)
after one removal



after a second removal
(copy your revised diagram from part 1 here, and edit it to show the result of the second removal)



4-2)


50
40
35
25
30
25
18
8
20
16
23
13