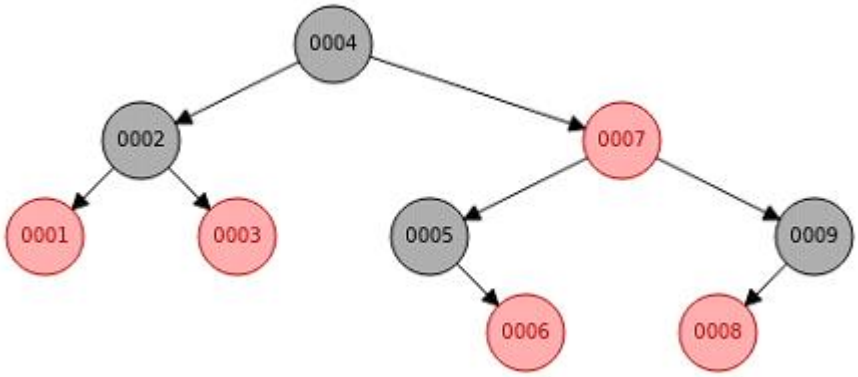
COP 3503 Section 11 Final Exam Review Answers

1)



2)

```
int profit = 0;
for(int i = 1; i < prices.length; i++){
    if(prices[i] > prices[i-1])
       profit += (prices[i] - prices[i-1]);
}
return profit;
```

ALTERNATE SOLUTION

```
int valley = prices[0];
int peak = prices[0];
int profit = 0;
int i = 0;
while (i < prices.length - 1) {
   while (i < prices.length - 1 && prices[i] >= prices[i + 1])
       i++;
    valley = prices[i]
    while (i < prices.length - 1 && prices[i] <= prices[i + 1])
        i++;
    peak = prices[i];
    profit += peak - valley;
}
return profit;
```

3) There are 6 ways for a, b, c at the beginning. There are also an additional 2 ways to sort the graph if x1 or x2 are first. Any of the six orderings of the beginning statements can be paired with the two orderings of the x1, x2 statements. Therefore, there are a total of 6 x 2 = 12 ways to order them.

4) Bits used from Huffman Tree:

Character	Encoding	Total Bits
‘A’	00	2 * 22 = 44
‘B’	10	2 * 36 = 72
‘C’	010	3 * 13 = 39
‘D’	110	3 * 15 = 45
‘E’	1110	4 * 10 = 40
‘F’	011	3 * 14 = 42
‘G’	1111	4 * 10 = 40

Sum the entire table: 44 + 72 + 39 + 45 + 40 + 42 + 40 = 322 bits

Original # of bits = 3(22+13+14+36+15+10+10) = 3(120) = 360 bits

Bits saved = 360 – 322 = 38 bits

5a) The path is alphabetical. The path goes from A to U. [A, B, C, D….U]. The reason is that every time we are at a letter, the subsequent letter is adjacent and unvisited.

5b) 21 – since we add a new recursive call to the stack for every single vertex without finishing any of the calls the max stack depth is 21, the number of vertices in the graph.

5c) P – this has to be traced carefully, but here is the path:

K→B→A→D→C→L→M→N→O→E→F→G→H→I→J→U→T→S→R→Q→P

COP 3503 Section 12 Final Exam Review Answers

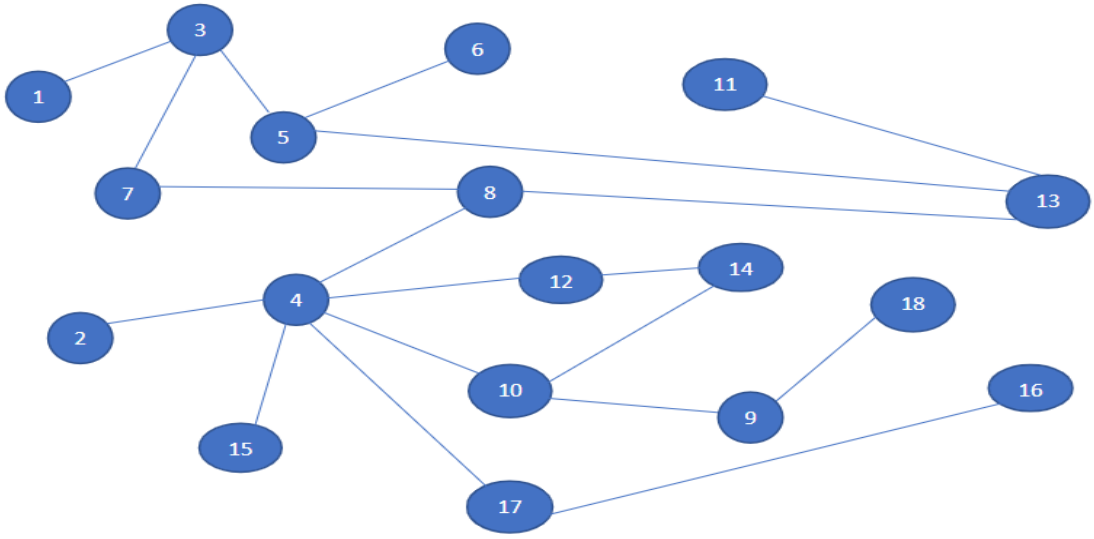
1)

Initial Array	First Sort	Second Sort	Third Sort	Fourth Sort	Fifth Sort	Sorted Array
446123	358231	446123	010099	010099	010099	010099
445123	446123	445123	446123	781543	445123	166787
358231	445123	358231	445123	445123	446123	358231
166787	781543	781543	358231	446123	358231	445123
781543	166787	166787	781543	166787	166787	446123
010099	010099	010099	166787	358231	781543	781543

2)

	C	E	A	R	T	R	A	N	D
C	1	1	1	1	1	1	1	1	1
A	1	1	2	2	2	2	2	2	2
T	1	1	2	2	3	3	3	3	3
E	1	2	2	2	3	3	3	3	3
R	1	2	2	3	3	4	4	4	4

3) Perform a depth first search (DFS) and a breath first search (BFS) with the following constraints and write the order in which the vertices are visited. When given multiple vertices to traverse to, prioritize odd numbered vertices over even numbered vertices. When two or more vertices can be visited after the first constraint, prioritize the smaller vertex value over larger ones. The starting vertex for both searches is vertex 1.

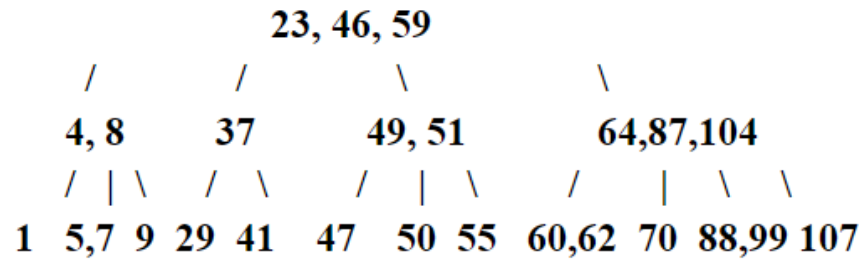


DFS: 1, 3, 5, 13, 11, 8, 7, 4, 15, 17, 16, 2, 10, 9, 18, 14, 12, 6

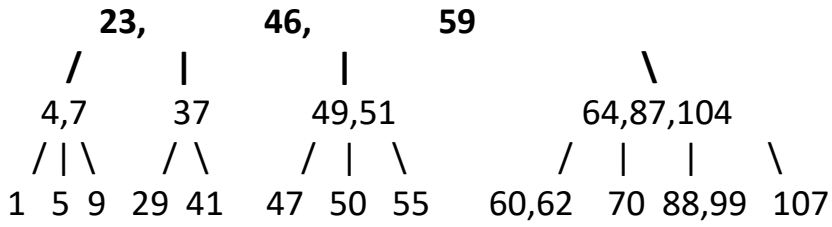
BFS: 1, 3, 5, 7, 13, 6, 8, 11, 4, 15, 17, 2, 10, 12, 16, 9, 14, 18

4)

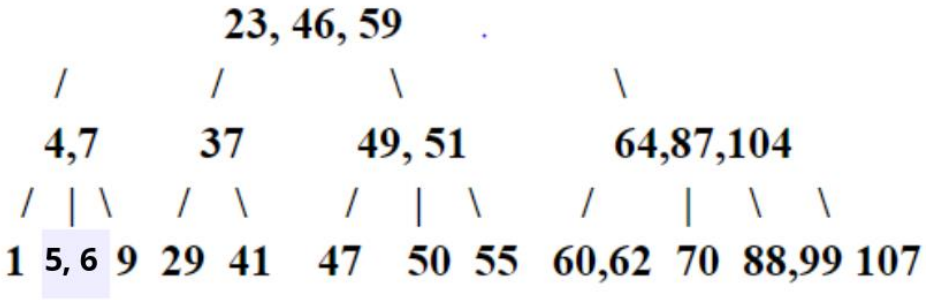
Insert 88:



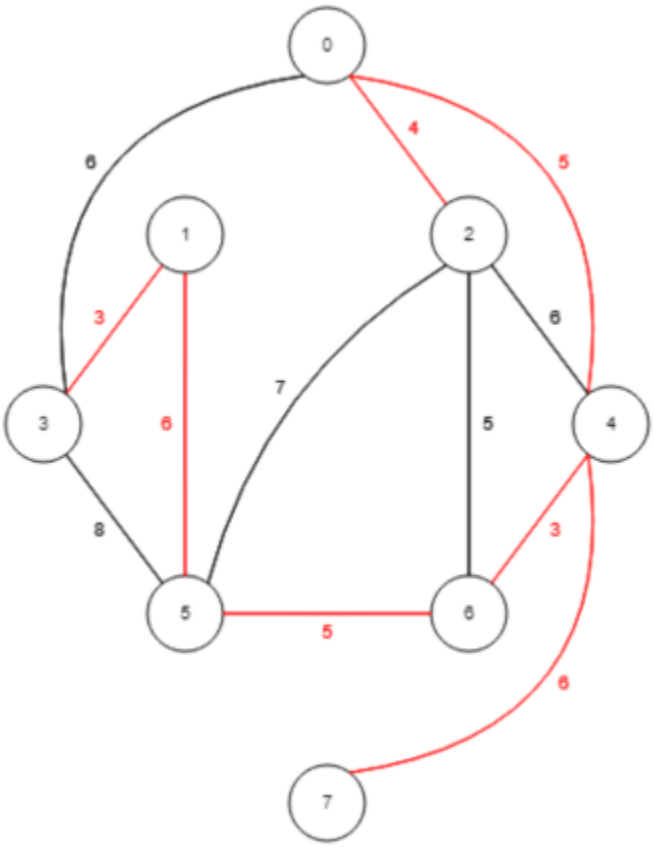
Delete 8:



Insert 6:



5)



Note: The edge (2, 6) can ALSO be selected as the second edge added instead of (0, 4). These are the two possible MST's that Prim's could produce starting at vertex 0.

6)

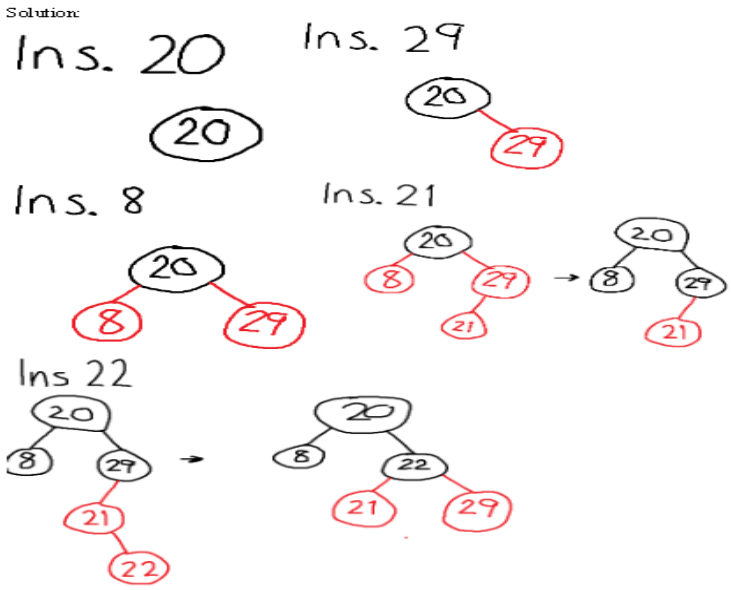
```
private int knapsack(int[] weights, int[] values, int i, int c){
    if(i >= weights.length) return 0;
    if(memo[i][c] != -1) return memo[i][c];

    if(c - weights[i] >= 0) {
        memo[i][c] = Math.max(values[i] + knapsack(weights, values, i + 1, c - weights[i]),
                              knapsack(weights, values, i + 1, c));
        return memo[i][c];
    }

    memo[i][c] = knapsack(weights, values, i + 1, c);
    return memo[i][c];
}
```

Note: In the problem statement, the last formal parameter was called c. In this solution, it's been renamed c.

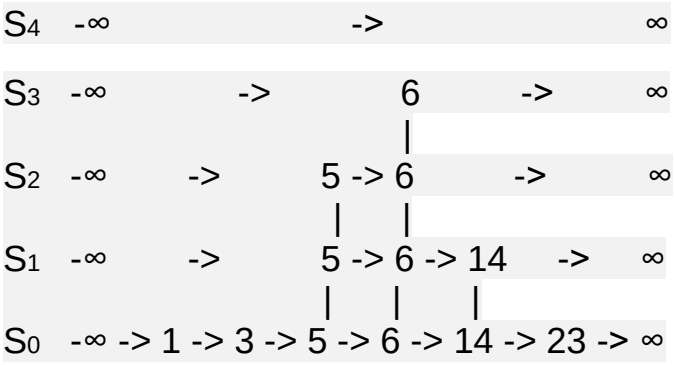
COP 3503 Section 13 Final Exam Review Answers

1) Show the state of a red-black tree after the following items have been inserted to an empty red-black tree, in this order: 20, 29, 8, 21, 22.

Solution:



2) After eating a questionable Oreo that you found in a CS1 classroom, you discover that you can see into the future, so of course you decide to test it out by making a skip list. Using Java's random class, you foretell that the next 12 numbers between 0 and 1 are [.32, .73, .65, .93, .21, .53, .64, .26, .50, .41, .77, .48]. What will the skip list look like if you are storing the elements 3, 6, 5, 1, 23, 14 in order, and to ascend an element the random number must be greater than .50?



3) Which representation faster for inserting a new vertex into a graph, adjacency list or adjacency matrix? Why?

Adjacency list. An adjacency list uses linked lists and has two pointers (to front and back nodes), Thus the insertion can be done in O(1) run time. Whereas with an adjacency matrix you need to increase its storage by copying the entire matrix. Giving it a time complexity of O(n*n), where n is the set of vertices.

4)

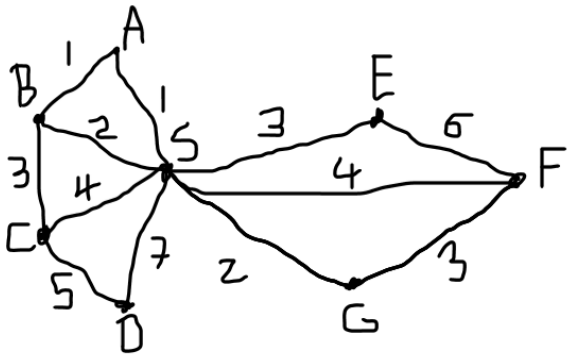
Initial Array	First Sort	Second Sort	Third Sort	Fourth Sort	Fifth Sort	Sorted Array
446123	358231	446123	010099	010099	010099	010099
445123	446123	445123	446123	781543	445123	166787
358231	445123	358231	445123	445123	446123	358231
166787	781543	781543	358231	446123	358231	445123
781543	166787	166787	781543	166787	166787	446123
010099	010099	010099	166787	358231	781543	781543

5)

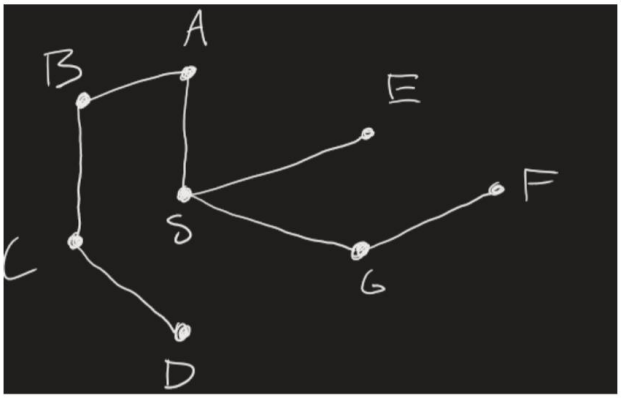
	C	E	A	R	T	R	A	N	D
C	1	1	1	1	1	1	1	1	1
A	1	1	2	2	2	2	2	2	2
T	1	1	2	2	3	3	3	3	3
E	1	2	2	2	3	3	3	3	3
R	1	2	2	3	3	4	4	4	4

COP 3503 Section 14 Final Exam Review Solutions

1) Draw the Minimum Spanning Tree for the following tree by using Prim's algorithm starting from node S.



Answer:



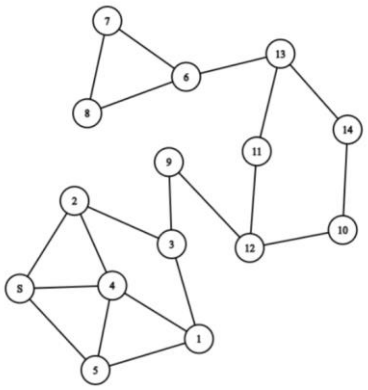
2) This question was omitted.

3) Perform a Radix sort on the following set of values:

123 321 781 555 902 319

```
V   V   V
321 902 123
781 319 319
902 321 321
123 123 555
555 555 781
419 781 902
FINAL SORTED LIST: 123, 319, 321, 555, 781, 902
```

4) Given the following graph, show the order the vertices get visited in a BFS starting at the vertex labeled ‘S’. When enqueuing the neighboring vertices of a vertex, do so in numerical order.



BFS Order: S, 2, 4, 5, 3, 1, 9, 12, 10, 11, 14, 13, 6, 7, 8

5) Convert the following 2-4 tree into a red-black tree