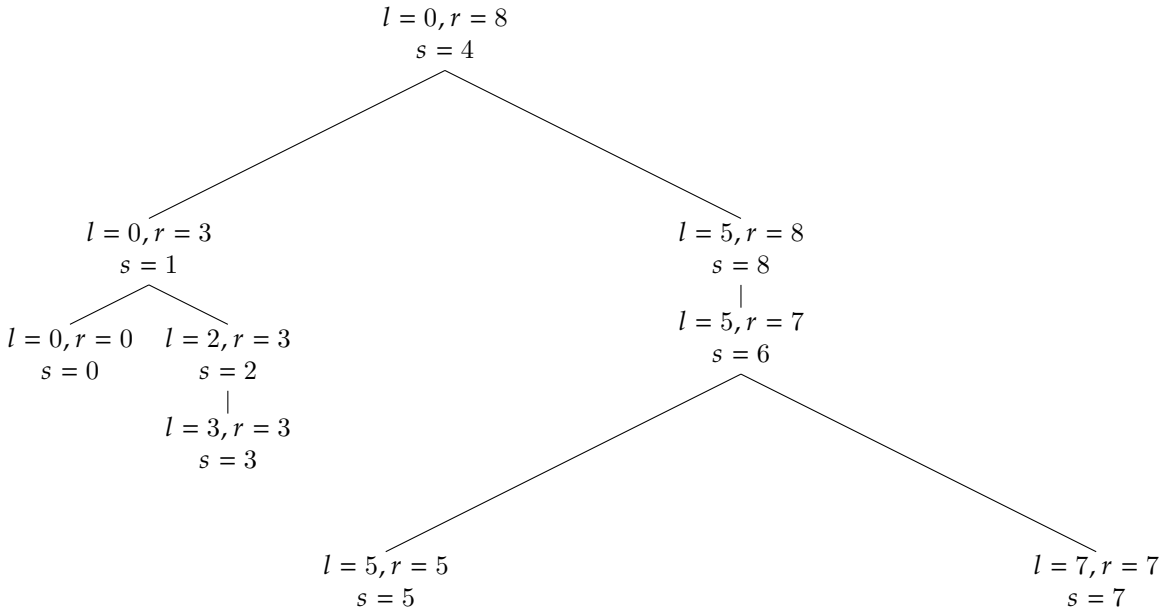
Assignment 3

Madiba Hudson-Quansah

Question 1

Apply quicksort to sort the list Q,U,I,C,K,S,O,R,T in alphabetical order. Draw the tree of the recursive calls made.

Solution:



Where the first element in the subarray is chosen as the pivot and where s and r are the start and end index of the partitioned subarray respectively, and s is the index of the pivot in the final sorted list. So $s = 5, r = 8, s = 8$ means for the subarray starting from index 5 to 8 inclusive the pivot ends up at the index 8 in the final list.

Question 2

1. Apply the bottom-up dynamic programming algorithm to the following instance of the knapsack problem
2. How many different optimal subsets does the instance of part (a) have?
3. In general, how can we use the table generated by the dynamic programming algorithm to tell whether there is more than one optimal subset for the knapsack problem's instance?
4. Instead of computing values in a bottom-up way, suppose we apply the memory function method to the instance of the knapsack problem given in part (a). Indicate the entries of the dynamic programming table that are (i) never computed by the memory function method, (ii) retrieved without a re-computation.

Solution:

1. The recurrence relation of the knapsack problem is:

$$F(i,j) = \begin{cases} \max\left\{F(i-1,j), v_i + F(i-1, j-w_i)\right\} & \text{if } j - w_i \geqslant 0 \\ F(i-1,j) & \text{if } j - w_i < 0 \end{cases}$$

Where i is the index of the ith item in consideration, j is the capacity of the knapsack, v_i is the value of the ith item, and w_i is the weight of the ith item. For the base cases $F(0,j) = 0$ for $j \geqslant 0$ and $F(i,0) = 0$ for $i \geqslant 0$ This gives us the table

i	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0
1	0	0	0	25	25	25	25
2	0	0	20	25	25	45	45
3	0	15	20	35	40	45	60
4	0	15	20	35	40	55	60
5	0	15	20	35	40	55	**65**

2. There is only one optimal subset.
3. By tracing back the table from the bottom right corner to the top left corner, if there is more than one path to the top left of the table, then there is more than one optimal subset.
4.

Question 3

Apply Warshall's algorithm to find the transitive closure of the digraph defined by the following adjacency matrix:

$$\begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

Solution:

$$R^{(0)} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$R^{(1)} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$R^{(2)} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$R^{(3)} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$R^{(4)} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Question 4

Design a divide-and-conquer algorithm for computing the number of levels in a binary tree. (In particular, the algorithm must return 0 and 1 for the empty and single-node trees, respectively.) What is the time efficiency class of your algorithm?

Solution:

Algorithm 1 NumLevels (R)

▷ Calculates the number of levels in a binary tree with root R
▷ Input: The root of the binary tree R
▷ Output: The number of levels in the binary tree

```
1: h ← 0
2: for each child c of R do
3:     h ← max (h, NumLevels (c) + 1)
4: end for
5: return h
```

The time efficiency class of the algorithm is $O(n)$ where n is the number of nodes in the binary tree.