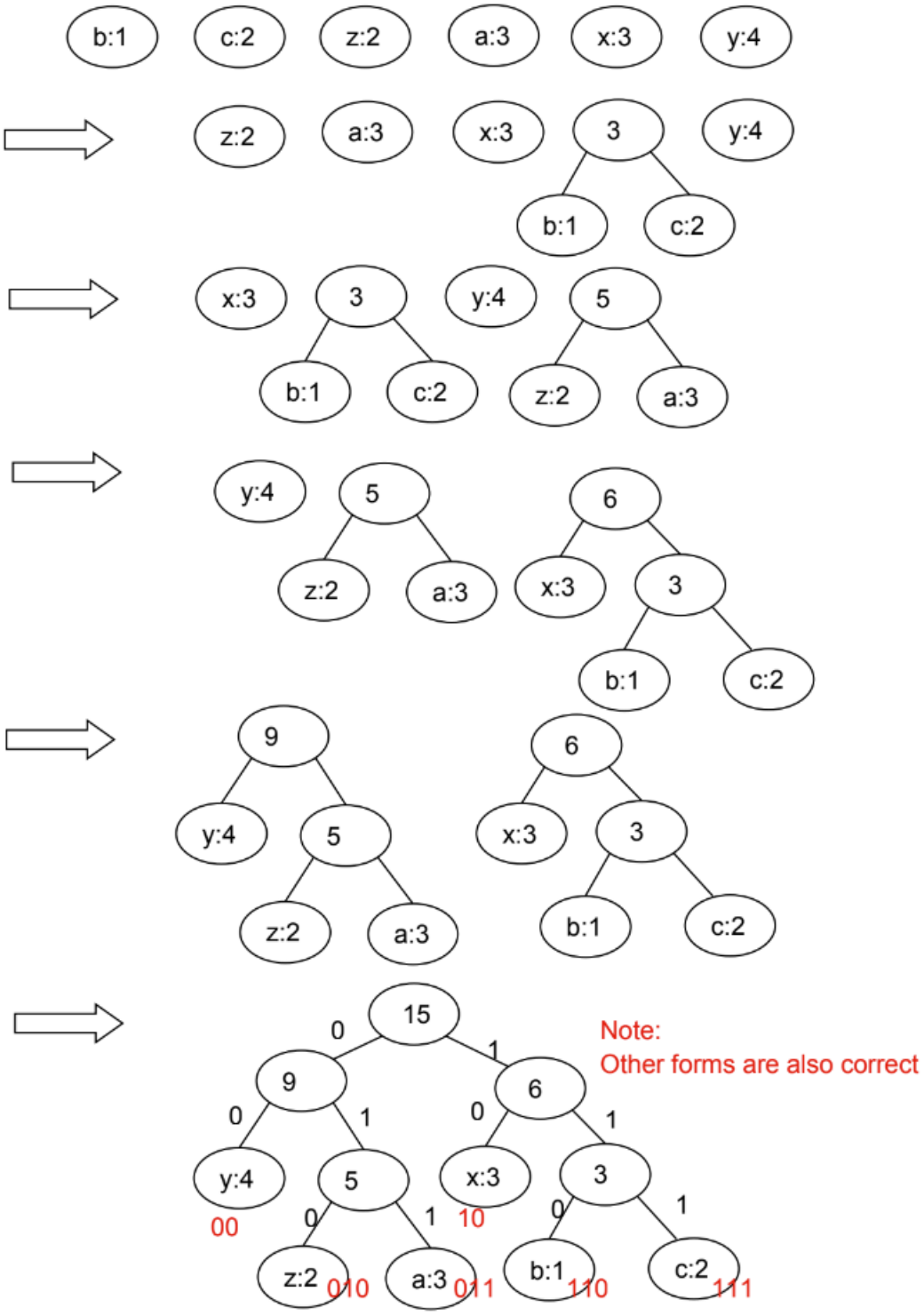
EL9343 Homework 11

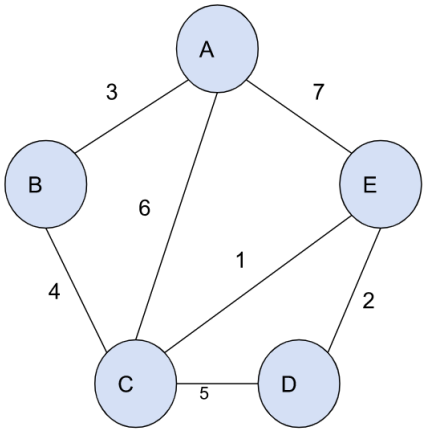
Due: Dec. 8th 11:00 a.m.

1. Design a greedy algorithm for arranging the queuing order in a supermarket. Suppose there are n customers come to the counter at the same time, noted as $c_1, c_2, \ldots, c_n$, the time to service i-th customer is s_i, $i = 1, 2, \ldots, n$, and the absolute time to finish i-th customer is T_i, $i = 1, 2, \ldots, n$. Your goal is to decide a queuing order of n customers to minimize the accumulated completion time (waiting time + service time) of all n customers, that is, to minimize $\sum_{i=1}^{n} T_i$.
   (a) Provide an algorithm to solve this issue;
   (b) Prove the correctness of your algorithm by showing the greedy choice property and optimal substructure;
   (c) Justify the running time of your algorithm.

   Solutions:
   (a) Sort the customers by serving time from smallest to largest and run them in that order.
   (b) Optimal Substructure:
   Suppose c is the first customer served in an optimal solution, then we can remove him, and the order we serve the remaining customers is also optimal in terms of minimizing the accumulated completion time.
   Greedy-choice property:
   Let O be an optimal solution. Let c be the customer which has the smallest serving time and let $c' \neq c$ be the first customer serve in O. Let O' be the solution obtained by switching the order in which we serve c and c' in O. Then we can prove that O' has a smaller accumulated completion time than O, which means O is not optimal.
   After this operation, the completion time of c' in O' equals to the completion time of c in O. However, the new completion time of c is smaller than the old time of c', and for everyone between c and c', the completion time is reduced since the waiting time is reduced. The completion times of all others remain the same. Therefore, the accumulated completion time is reduced by changing from O' to O, proving that the greedy solution gives an optimal solution.
   (c) The running time is $O(n \log n)$ because we need to first do a sorting.
2. How many bits are required to encode the message "aaabccxxxyyyyzz" using Huffman Codes? Please show the coding tree you build.
   Solutions: We can build the coding tree as in the following figure. The total number of bits is 38.



3. Consider the following graph.



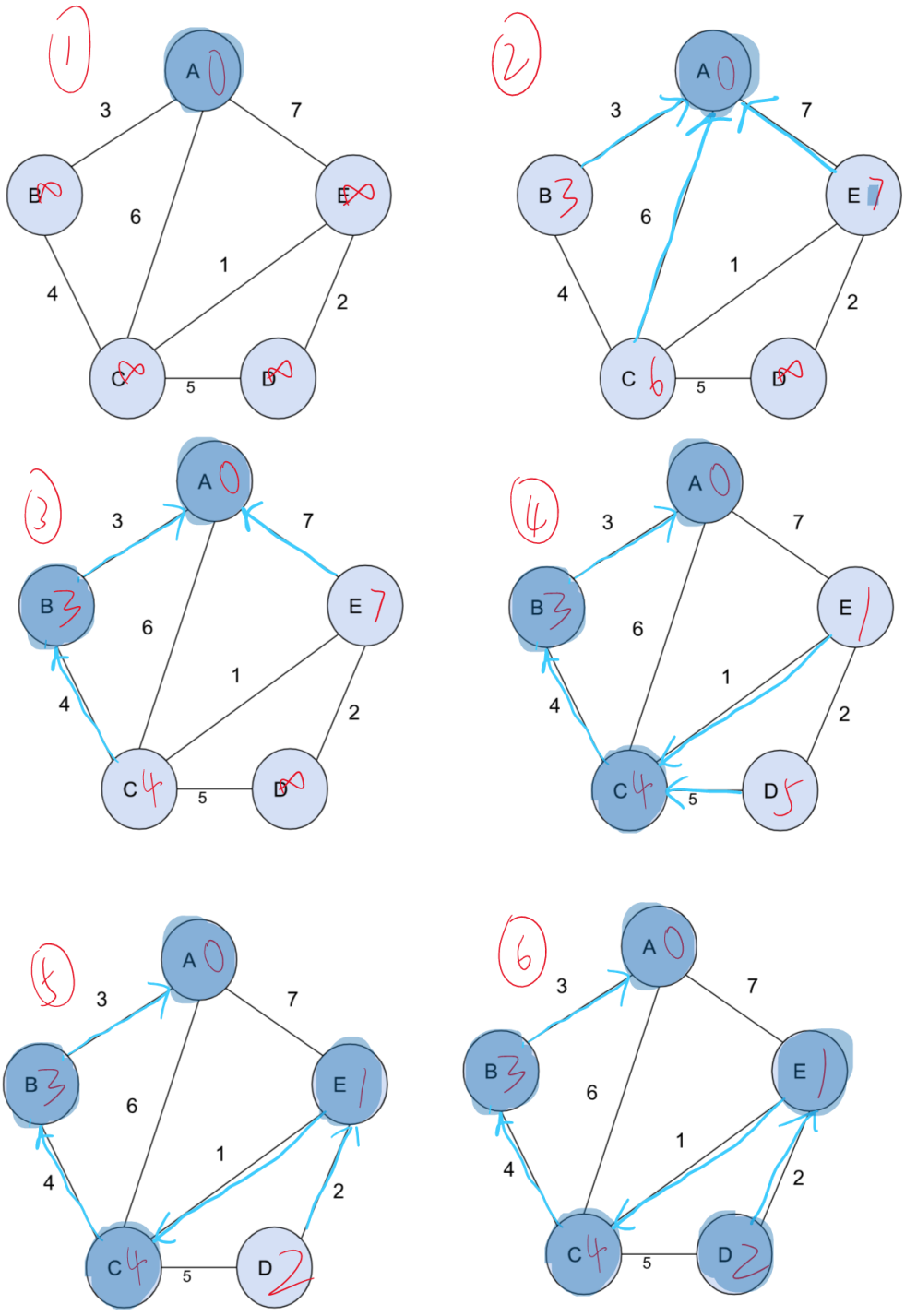
(a) If we run Kruskal's algorithm in the graph, what will be the sequence in which edges are added to the MST?

(b) Demonstrate Prim's algorithm in the graph with A as the source.

Solutions:

(a) The sequence is $\{(C, E), (D, E), (A, B), (B, C)\}$.

(b) Shows as below:



4. Let G be an n-vertex connected undirected graph with costs on the edges. Assume that all the edge costs are distinct.

   (a) Prove that G has a unique MST;

   (b) Give an algorithm to find a cycle in G such that the maximum cost of edge in the cycle is minimum among all possible cycles. The time complexity of your algorithm should be $O(|E| \log |E|)$. Assume

that the graph has at least one cycle, and finding MST takes $O(|E| \log |V|)$ time.

Solutions:

(a) Assuming there are two different MST, T_1 and T_2. There must be at least one edge $x \in T_1$ that does not belong to T_2. Let the C_1 and C_2 be the two sub-tree connected by x in T_1.
Adding x to T_2 will result in a cycle. There must be an edge $y \in T_2$ in this cycle that doesn't belong to T_1, and it connects C_1 and C_2 (otherwise C_1 and C_2 will be disconnected in T_2, of course impossible).
Since the edge costs are distinct, if x costs more, we can remove x from T_1 and add y to make a new tree whose total cost is smaller, contradiction to the MST assumption (similar when y costs more). Therefore, the MST in this graph must be unique.

(b) First find the unique MST, the time is $O(|E| \log |V|)$.
Choose the smallest edge that is not in the MST, add it to the MST will result in a cycle. This cycle is the one that is asked for, because the added edge has the maximum cost among the edges in that cycle (otherwise we can substitute the added edge to get a smaller-cost MST).
The time of finding the edge is $O(|E|)$. **Getting here is enough for the full score.** To further get the entire cycle, we can do BFS or DFS in the MST, since there is only one path between pair of nodes in the tree, and the cost is low as $O(|V|)$. Therefore the total cost is $O(|E| \log |V|)$.
Note: In the common simple graphs, $|E| \approx |V|^2$, so $O(|E| \log |V|)$ and $O(|E| \log |E|)$ is the same here.