Student Information

Full Name : Burak Metehan Tunçel
Id Number : 2468726

Answer 1

I choose the Prim's algorithm to construct minimum spanning tree.

Answer of a

Choice	Edge	Cost
1	{e,f}	1
2	{e,h}	2
3	{h,g}	2
4	{c,f}	3
5	{d,g}	3
6	{a,d}	2
7	{b,d}	3
8	{h,i}	4

Table 1: The order in which the edges are added to the tree.

Answer of b

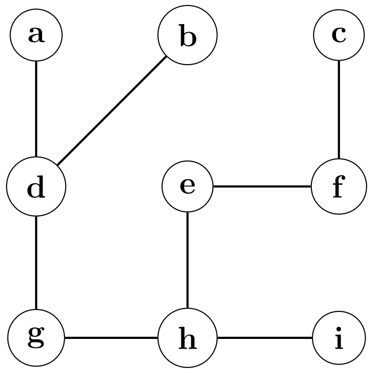
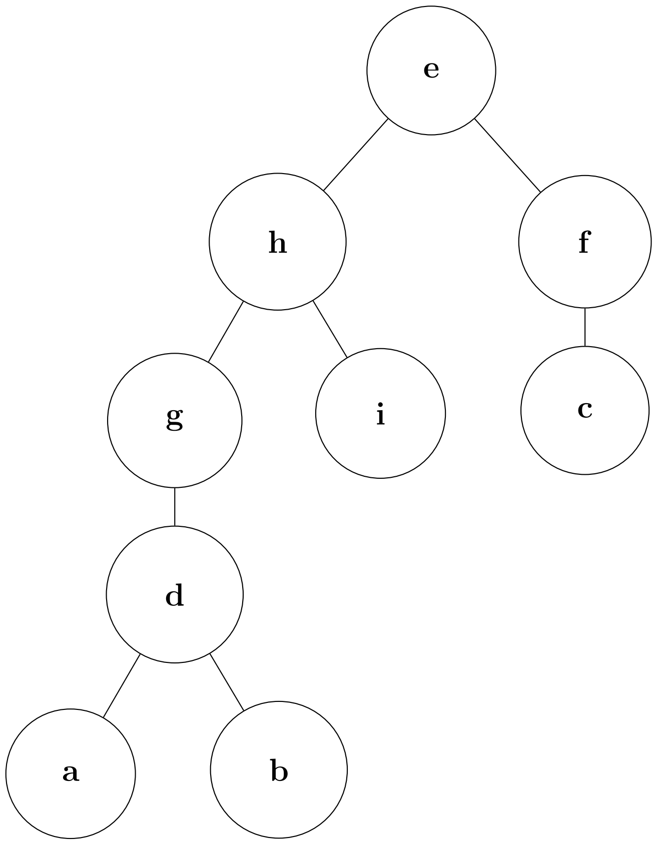


Figure 1: The minimum spanning tree of the Graph G in Figure 1.

The more common tree representation is the following



Answer of c

The minimum spanning tree (MST), which is found the part b, is unique for the graph G in Figure 1. However, in general, there is not absolute one MST for any connected edge-weighted undirected graph. That is, there can be more than 1 MST for a connected edge-weighted undirected graph.

As an example, there can be another MST if the edge (b,c) has weight 3 in graph G in Figure 1. When it happens, b can also be connected from c instead of d.

Answer of d

There are two mainly used algorithms to construct minimum spanning tree (MST), namely Prim's algorithm and Kruskal's algorithm.
Both algorithm starts by choosing the minimum-weight/smallest-weight edge of the graph. If there is only one, unique, minimum-weight/smallest-weight edge, this edge has to be chosen by both algorithm at start. Therefore, this edge has to be included in the MST if there is a MST.

Answer 2

G and H are *isomorphic*. G and H have the same number of vertices, number of edges, and number of vertices of each degree.

- Both G and H have $\underline{6}$ vertices and $\underline{8}$ edges.
- Both have $\underline{3}$ vertices of degree $\underline{2}$, $\underline{2}$ vertices of degree $\underline{3}$, and $\underline{1}$ vertex of degree $\underline{4}$.

Since G and H agree with respect to these invariants, it is reasonable to try to find an *isomorphism* f.

The following one-to-one correspondence between G and H is the our *isomorphism* f.

- $f(a) = m$
- $f(b) = q$
- $f(c) = p$
- $f(d) = r$
- $f(e) = n$
- $f(f) = o$

So, G and H are *isomorphic*.

Answer 3

Answer of a

- **Number of vertices:** 7
- **Number of edges:** 6
- **Height of T:** 3

Answer of b

- **Postorder:** (q:13), (s:19), (u:23), (v:58), (t:43), (r:24), (p:17)
- **Inorder:** (q:13), (p:17), (s:19), (r:24), (u:23), (t:43), (v:58)
- **Preorder:** (p:17), (q:13), (r:24), (s:19), (t:43), (u:23), (v:58)

Answer of c

T is *a full binary tree.*

Explanation *Full Binary Tree*

A full binary tree is a tree in which every node other than the leaves has two children.

In other words, a full binary tree is a tree which every node has 0 or 2 children.

Since the every node of the T has zero or two children, it can be told that T is *a full binary tree.*

Answer of d

T is *not a complete binary tree.*

Explanation *Complete Binary Tree*

A complete binary tree is a binary tree in which every level, except possibly the last, is completely filled, and all nodes are as far left as possible.

In other words, a complete binary tree is a tree which all the levels are completely filled except possibly the last level and the last level has all keys as left as possible.

In T, the level 2 is not fully completed and the vertices in the last level are located in right instead of left.

Since T does not satisfy the above explanation, it can be told that T is *not a complete binary tree.*

Answer of e

T is *not a balanced binary tree.*

Explanation

If the subtrees at each vertex contain paths of approximately the same length, it can be told that the tree is balanced.

In other words, a balanced binary tree is a binary tree in which the height of the left and right subtree of any node differ by not more than 1 (0 or 1).

In T, the height of the left subtree of (p:17) is 0 whereas the height of the right subtree of (p:17) is 2. This indicates that it is not balanced.

Since T does not satisfy the above explanation, it can be told that T is *not a complete binary tree.*

Answer of f

T is *not a binary search tree.*

Explanation

A binary search tree is a rooted binary tree whose internal nodes each store a key greater than all the keys in the node's left subtree and less than those in its right subtree.

In T, the right subtree of the (r:24) includes (u:23) whose key is 23. 23 is less than 24 ($23 \leq 24$); therefore, (u:23) should be located in the left subtree of (r:24) if T is a binary search tree.

Since T does not satisfy the above explanaiton, it can be told that T is not *not a binary search tree.*

Answer of g

The answer is **11** according to following theorem.

Theorem *The minimum number of the nodes for a full binary tree with height λ*

The minimum number of the nodes for a full binary tree with height λ is $2\lambda + 1$

Proof. We can use induction to prove the above theorem.

Basis Step: If the height is 0 ($\lambda = 0$), there should be 1 node ($2 \times 0 + 1 = 1$). There will be only root node.
Inductive Step: Assume that the minimum number of nodes when height is k ($\lambda = k$) is $2k + 1$ is N.

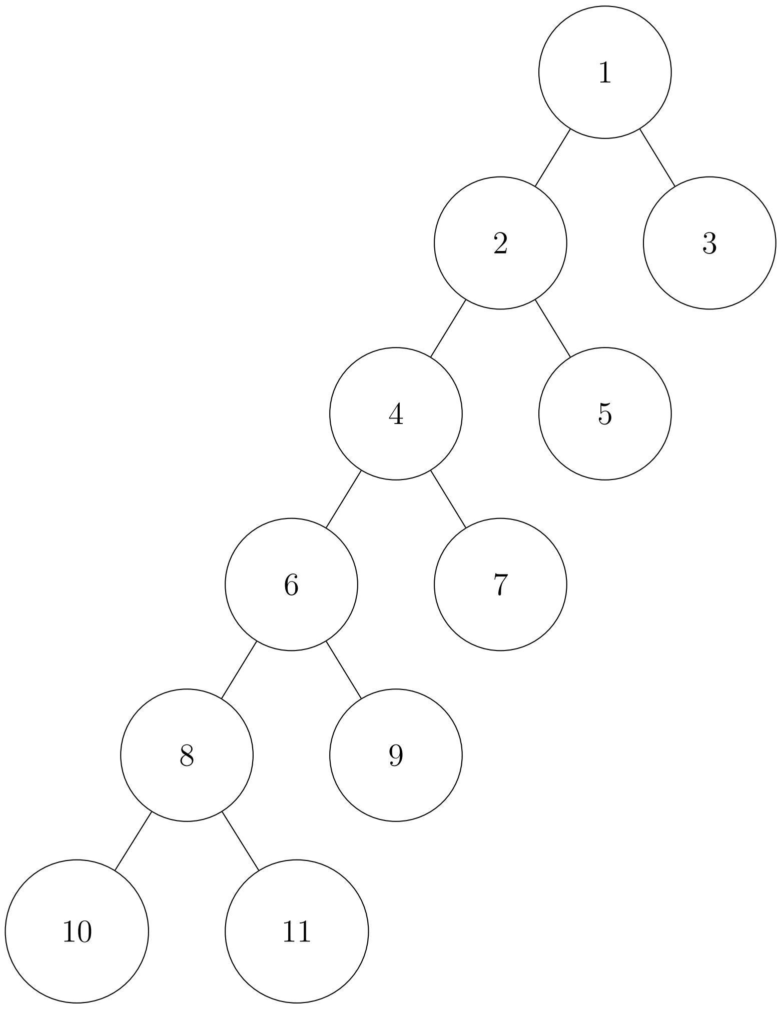
When we add 1 node to the a leaf located at the deepest position (has the maximum height and the level count), the height will increase 1 and become $k+1$. To maintain a full binary tree, it will be necessary to add another node. In other words, the new level will need at least two more nodes. So, the number of nodes will be $N+2$.

Then, by our induction hypothesis $N+2 = (2k+1)+2 = 2(k+1)+1$, which is what we wanted.

□

In other words, basically, to achieve the full binary tree, there should be at least 2 nodes at each level except level 0 where root is located. So, for height 5, there should be 5 levels, each of which contains 2 nodes, and 1 root. $5 \times 2 + 1 = 11$.

One of the example full binary trees with height 5 and have the minimum number of the nodes is given below.


1
2
3
4
5
6
7
8
9
10
11