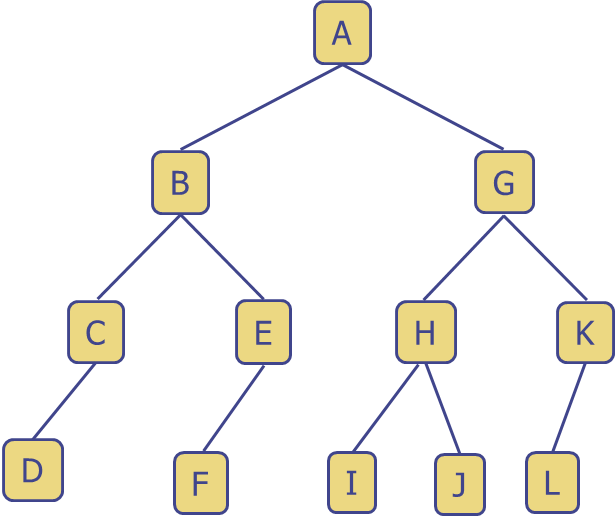
**Tutorial 5**

**Question 1: Trees**

For each of the following trees, fill in its corresponding three tables:

* The first table is about generic properties of the tree.
* The second table is about properties for specific nodes in the tree.
* The third table contains an array. For each node in the tree, you should mark which position in the array it should occupy.

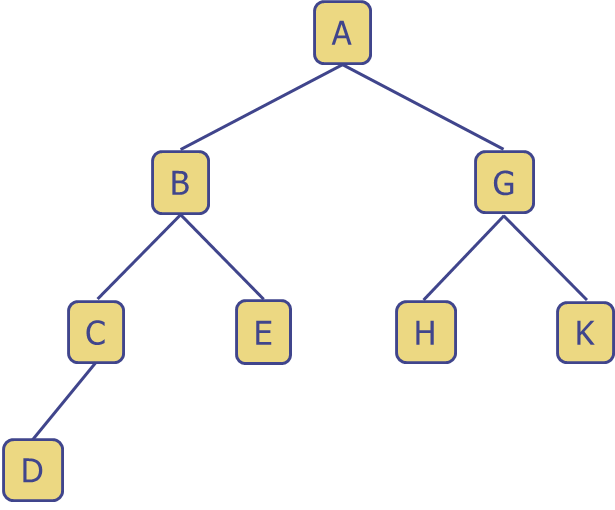
**Tree 1**

****

|  |  |  |  |
| --- | --- | --- | --- |
| Property | | Answer | |
| Root Node | | A | |
| Internal Nodes (内部节点是至少有一个子节点的节点) | | A, B, G, C, E, H, K | |
| External Nodes(外部节点（叶子）是没有子节点的节点) | | D, F, I, J, L | |
| Height of tree(树的高度定义为**从根节点到最深叶子节点的最长路径上的边数)** | | 1. B-C-D=3 2. B-E-F=3 3. G-H-I/J=3 4. G-K-L=3   therefore, Height of Tree is 3 | |
| Inorder traversal(中序遍历规则（Left → Root → Right）：  1. 递归遍历左子树  2. 访问根节点  3. 递归遍历右子树) | | D(left)-A(root)-L(right)  D-C-B-F-E-A-I-H-J-G-L-K | |
| Preorder traversal(先序遍历规则（Root → Left → Right）：  1. 访问根节点  2. 递归遍历左子树  3. 递归遍历右子树) | | A-B-C-D-E-F-G-H-I-J-K-L | |
| Postorder traversal(后序遍历的规则是（Left → Right → Root）：  1. 递归遍历左子树  2. 递归遍历右子树  3. 访问根节点) | | D-C-F-E-B-I-J-H-L-K-G-A | |
|  | **Depth(深度**是指从根节点 A 到该节点所经过的边数**)** | | **Ancestor(祖先**是指从根节点到该节点所经过的所有父节点**)** | | **Descendants(子孙**是指该节点的所有直接或间接子节点。) |
| **A** | **0** | | **-** | | B, C, D, E, F, G, H, I, J, K, L |
| **B** | **1** | | **A** | | C, D, E, F |
| **C** | **2** | | **A, B** | | **D** |
| **D** | **3** | | **A,B,C** | | **-** |
| **E** | **2** | | **A,B** | | **F** |
| **F** | **3** | | **A,B,E** | | **-** |
| **G** | **1** | | **A** | | **H,K,I,J,L** |
| **H** | **2** | | **A,G** | | **I,J** |
| **I** | **3** | | **A,G,H** | | **-** |
| **J** | **3** | | **A,G,H** | | **-** |
| **K** | **2** | | **A,G** | | **L** |
| **L** | **3** | | **A,G,K** | | **-** |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** |
| **-** | **A** | **B** | **G** | **C** | **E** | **H** | **K** | **D** |  | **F** |  | **I** | **J** | **L** |  |

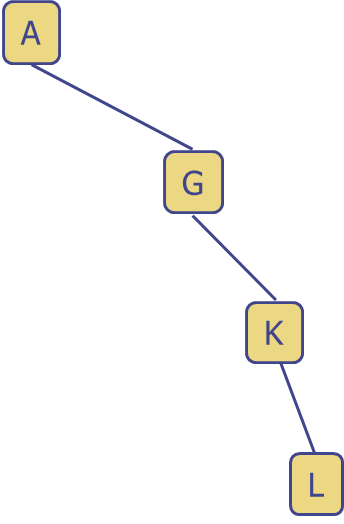
**Tree 2**

****

|  |  |  |  |
| --- | --- | --- | --- |
| Property | | Answer | |
| Root Node | | A | |
| Internal Nodes | | B,G,C | |
| External Nodes | | D,E,H,K | |
| Height of tree | | 1. B-C-D=3 2. B-E=2 3. G-H//K=2   Therefore, 3 | |
| Inorder traversal | | D-C-B-E-A-H-G-K | |
| Preorder traversal | | A-B-C-D-E-G-H-K | |
| Postorder traversal | | D-C-E-B-H-K-G-A | |
|  | **Depth** | | **Ancestor** | | **Descendants** |
| **A** |  | |  | |  |
| **B** |  | |  | |  |
| **C** |  | |  | |  |
| **D** |  | |  | |  |
| **E** |  | |  | |  |
| **G** |  | |  | |  |
| **H** |  | |  | |  |
| **K** |  | |  | |  |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** |
| **-** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

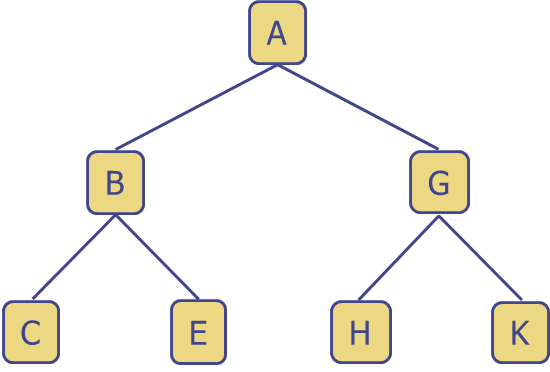
**Tree 3**

****

|  |  |  |  |
| --- | --- | --- | --- |
| Property | | Answer | |
| Root Node | | A | |
| Internal Nodes | | G,K | |
| External Nodes | | L | |
| Height of tree | | A-G-K-L=3 | |
| Inorder traversal | |  | |
| Preorder traversal | |  | |
| Postorder traversal | |  | |
|  | **Depth** | | **Ancestor** | | **Descendants** |
| **A** |  | |  | |  |
| **G** |  | |  | |  |
| **K** |  | |  | |  |
| **L** |  | |  | |  |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** |
| **-** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

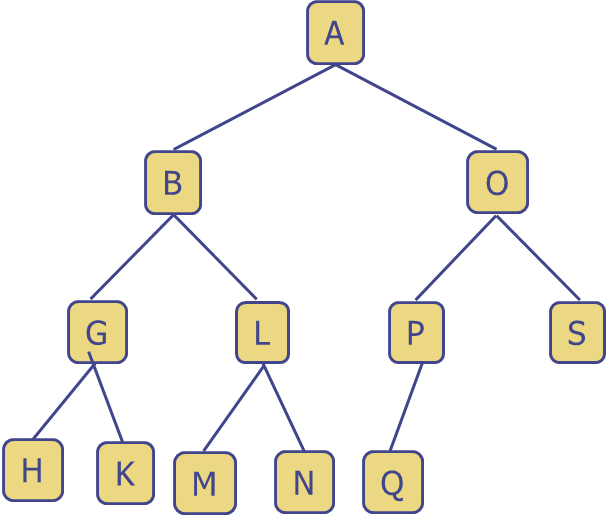
**Tree 4**

****

|  |  |  |  |
| --- | --- | --- | --- |
| Property | | Answer | |
| Root Node | |  | |
| Internal Nodes | |  | |
| External Nodes | |  | |
| Height of tree | |  | |
| Inorder traversal | |  | |
| Preorder traversal | |  | |
| Postorder traversal | |  | |
|  | **Depth** | | **Ancestor** | | **Descendants** |
| **A** |  | |  | |  |
| **B** |  | |  | |  |
| **C** |  | |  | |  |
| **E** |  | |  | |  |
| **G** |  | |  | |  |
| **H** |  | |  | |  |
| **K** |  | |  | |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** |
| **-** |  |  |  |  |  |  |  |

**Tree 5**

****

|  |  |  |  |
| --- | --- | --- | --- |
| Property | | Answer | |
| Root Node | |  | |
| Internal Nodes | |  | |
| External Nodes | |  | |
| Height of tree | |  | |
| Inorder traversal | |  | |
| Preorder traversal | |  | |
| Postorder traversal | |  | |
|  | **Depth** | | **Ancestor** | | **Descendants** |
| **A** |  | |  | |  |
| **B** |  | |  | |  |
| **G** |  | |  | |  |
| **H** |  | |  | |  |
| **K** |  | |  | |  |
| **L** |  | |  | |  |
| **M** |  | |  | |  |
| **N** |  | |  | |  |
| **O** |  | |  | |  |
| **P** |  | |  | |  |
| **Q** |  | |  | |  |
| **S** |  | |  | |  |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** |
| **-** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

Additional questions:

1. Which of the above trees, if any, are proper binary trees?
2. How big of an array do we need to store an arbitrary binary tree of height *h*?
3. We have shown how to use an array representation for binary trees. How would we extend this to work on ternary trees?