Data Structures Test Bank 2015

Data Structures Tree Traversals

Question 1

Following function is supposed to calculate the maximum depth or height of a Binary tree -- the number of nodes along the longest path from the root node down to the farthest leaf node.

```
int maxDepth(struct node* node)
{
   if (node==NULL)
       return 0;
   else
   {
       /* compute the depth of each subtree */
       int lDepth = maxDepth(node->left);
       int rDepth = maxDepth(node->right);

       /* use the larger one */
       if (lDepth > rDepth)
           return X;
       else return Y;
   }
}
```

What should be the values of X and Y so that the function works correctly?

A X = lDepth, Y = rDepth

B X = lDepth + 1, Y = rDepth + 1

C X = lDepth - 1, Y = rDepth -1

D None of the above

Question 1 Explanation:

If a tree is not empty, height of tree is MAX(Height of Left Subtree, Height of Right Subtree) + 1
See program to Find the Maximum Depth or Height of a Tree for more details.

Data Structures Test Bank 2015

Question 2

What is common in three different types of traversals (Inorder, Preorder and Postorder)?

A Root is visited before right subtree

B Left subtree is always visited before right subtree

C Root is visited after left subtree

D All of the above

E None of the above

Question 2 Explanation:

The order of inorder traversal is LEFT ROOT RIGHT The order of preorder traversal is ROOT LEFT RIGHT The order of postorder traversal is LEFT RIGHT ROOT In all three traversals, LEFT is traversed before RIGHT

Question 3

The inorder and preorder traversal of a binary tree are d b e a f c g and a b d e c f g, respectively. The postorder traversal of the binary tree is:

A d e b f g c a

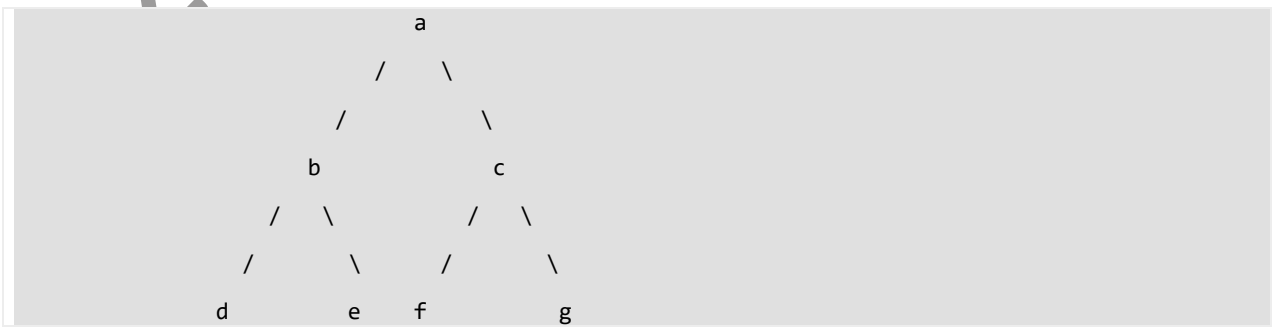
B e d b g f c a

C e d b f g c a

D d e f g b c a

Question 3 Explanation:

Below is the given tree.



Data Structures Test Bank 2015

Question 4

What does the following function do for a given binary tree?

```
int fun(struct node *root)
{
   if (root == NULL)
      return 0;
   if (root->left == NULL && root->right == NULL)
      return 0;
   return 1 + fun(root->left) + fun(root->right);
}
```

A Counts leaf nodes

B Counts internal nodes

C Returns height where height is defined as number of edges on the path from root to deepest node

D Return diameter where diameter is number of edges on the longest path between any two nodes.

Question 4 Explanation:

The function counts internal nodes. 1) If root is NULL or a leaf node, it returns 0. 2) Otherwise returns, 1 plus count of internal nodes in left subtree, plus count of internal nodes in right subtree. See the following complete program. 1

Question 5

Which of the following pairs of traversals is not sufficient to build a binary tree from the given traversals?

A Preorder and Inorder

B Preorder and Postorder

C Inorder and Postorder

D None of the Above

Question 5 Explanation:

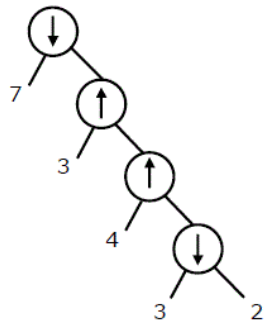
See http://www.geeksforgeeks.org/if-you-are-given-two-traversal-sequences-can-you-construct-the-binary-tree/ for details.

Data Structures Test Bank 2015

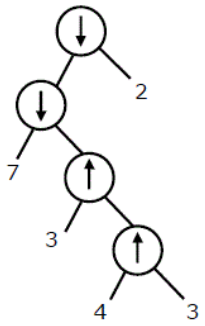
Question 6

Consider two binary operators ' ' and ' ' with the precedence of operator being lower than that of the operator. Operator is right associative while operator is left associative. Which one of the following represents the parse tree for expression (7 3 4 3 2)? (GATE CS 2011)

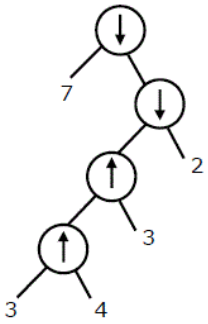
(A)



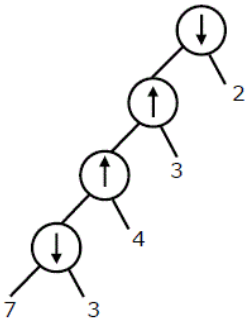
(B)



(C)



(D)



A A

B B

C C

D D

Question 6 Explanation:

Let us consider the given expression ([Tex]7 \downarrow 3 \uparrow 4 \uparrow 3 \downarrow 2[/Tex]). Since the precedence of [Tex]\uparrow[/Tex] is higher, the sub-expression ([Tex]3 \uparrow 4 \uparrow 3[/Tex]) will be evaluated first. In this sub-expression, [Tex]4 \uparrow 3[/Tex] would be evaluated first because [Tex]\uparrow[/Tex] is right to left associative. So the expression is evaluated as [Tex]((7 \downarrow (3 \uparrow (4 \uparrow 3))) \downarrow 2)[/Tex]. Also, note that among the two [Tex]\downarrow [/Tex] operators, first one is evaluated before the second one because the associativity of [Tex]\downarrow[/Tex] is left to right.

Data Structures Test Bank 2015

Question 7

Which traversal of tree resembles the breadth first search of the graph?

A Preorder

B Inorder

C Postorder

D Level order

Question 7 Explanation:

Breadth first search visits all the neighbors first and then deepens into each neighbor one by one.
The level order traversal of the tree also visits nodes on the current level and then goes to the next level.

Question 8

Which of the following tree traversal uses a queue data structure?

A Preorder

B Inorder

C Postorder

D Level order

Question 8 Explanation:

Level order traversal uses a queue data structure to visit the nodes level by level.

Data Structures Test Bank 2015

Question 9

Which of the following cannot generate the full binary tree?

A Inorder and Preorder

B Inorder and Postorder

C Preorder and Postorder

D None of the above

Question 9 Explanation:

To generate a binary tree, two traversals are necessary and one of them must be inorder. But, a full binary tree can be generated from preorder and postorder traversals. Read the algorithm **here**. Read Can tree be constructed from given traversals?

Question 10

Consider the following C program segment

```
struct CellNode
{
  struct CelINode *leftchild;
  int element;
  struct CelINode *rightChild;
}

int Dosomething(struct CelINode *ptr)
{
    int value = 0;
    if (ptr != NULL)
    {
      if (ptr->leftChild != NULL)
        value = 1 + DoSomething(ptr->leftChild);
      if (ptr->rightChild != NULL)
        value = max(value, 1 + DoSomething(ptr->rightChild));
    }
    return (value);
}
```

The value returned by the function DoSomething when a pointer to the root of a non-empty tree is passed as argument is (GATE CS 2004)

Data Structures Test Bank 2015

A The number of leaf nodes in the tree

B The number of nodes in the tree

C The number of internal nodes in the tree

D The height of the tree

Question 10 Explanation:

Explanation: DoSomething() returns max(height of left child + 1, height of left child + 1). So given that pointer to root of tree is passed to DoSomething(), it will return height of the tree. Note that this implementation follows the convention where height of a single node is 0.

Question 11

Let LASTPOST, LASTIN and LASTPRE denote the last vertex visited in a postorder, inorder and preorder traversal. Respectively, of a complete binary tree. Which of the following is always true? (GATE CS 2000)

A LASTIN = LASTPOST

B LASTIN = LASTPRE

C LASTPRE = LASTPOST

D None of the above

Question 11 Explanation:

It is given that the given tree is complete binary tree. For a complete binary tree, the last visited node will always be same for inorder and preorder traversal. None of the above is true even for a complete binary tree. The option (a) is incorrect because the last node visited in Inorder traversal is right child and last node visited in Postorder traversal is root. The option (c) is incorrect because the last node visited in Preorder traversal is right child and last node visited in Postorder traversal is root. For option (b), see the following counter example. Thanks to Hunaif Muhammed for providing the correct explanation.

```
     1
   /   \
  2     3
 / \   /
4   5 6
```

Data Structures Test Bank 2015

```
Inorder traversal is 4 2 5 1 6 3

Preorder traversal is 1 2 4 5 3 6
```

Question 12

The array representation of a complete binary tree contains the data in sorted order. Which traversal of the tree will produce the data in sorted form?

A Preorder

B Inorder

C Postorder

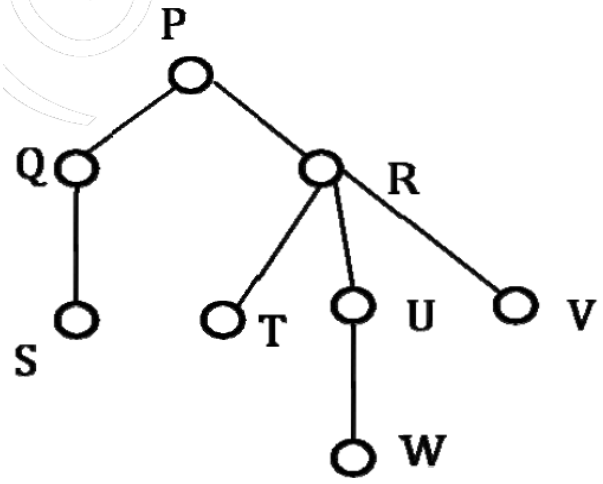
D Level order

Question 12 Explanation:

The level order traversal of a binary tree prints the data in the same order as it is stored in the array representation of a complete binary tree.

Question 13

Consider the following rooted tree with the vertex P labeled as root



The order in which the nodes are visited during in-order traversal is

Data Structures Test Bank 2015

A SQPTRWUV

B SQPTURWV

C SQPTWUVR

D SQPTRUWV

Question 13 Explanation:

```
Algorithm Inorder(tree) - Use of Recursion
Steps:
1. Traverse the left subtree,
   i.e., call Inorder(left-subtree)
2. Visit the root.
3. Traverse the right subtree,
   i.e., call Inorder(right-subtree)

Understanding this algorithm requires the basic
understanding of Recursion

Therefore, We begin in the above tree with root as
the starting point, which is P.

# Step 1( for node P) :
Traverse the left subtree of node or root P.
So we have node Q on left of P.

-> Step 1( for node Q)
Traverse the left subtree of node Q.
So we have node S on left of Q.

* Step 1 (for node S)
Now again traverse the left subtree of node S which is
NULL here.

* Step 2(for node S)
```

```
Visit the node S, i.e print node S as the 1st element of
inorder traversal.

* Step 3(for node S)
Traverse the right subtree of node S.
Which is NULL here.

Now move up in the tree to Q which is parent
of S.( Recursion, function of Q called for function of S).
Hence we go back to Q.

-> Step 2( for node Q):
Visit the node Q, i.e print node Q as the 2nd
element of inorder traversal.

-> Step 3 (for node Q)
Traverse the right subtree of node Q.
Which is NULL here.

Now move up in the tree to P which is parent
of Q.( Recursion, function of P called for function of Q).
Hence we go back to P.

# Step 2(for node P)
Visit the node P, i.e print node S as the 3rd
element of inorder traversal.

# Step 3 (for node P)
Traverse the right subtree of node P.
Node R is at the right of P.

Till now we have printed SQP as the inorder of the tree.
Similarly other elements can be obtained by traversing
the right subtree of P.

The final correct order of Inorder traversal would
```

```
be SQPTRWUV.
```

Question 14

Consider the pseudocode given below. The function DoSomething() takes as argument a pointer to the root of an arbitrary tree represented by the leftMostChild-rightSibling representation. Each node of the tree is of type treeNode.

```
typedef struct treeNode* treeptr;
struct treeNode
{
    treeptr leftMostChild, rightSibling;
};
int DoSomething (treeptr tree)
{
    int value=0;
    if (tree != NULL)
    {
        if (tree->leftMostChild == NULL)
            value = 1;
        else
            value = DoSomething(tree->leftMostChild);
        value = value + DoSomething(tree->rightSibling);
    }
    return(value);
}
```

When the pointer to the root of a tree is passed as the argument to DoSomething, the value returned by the function corresponds to the

A number of internal nodes in the tree.

B height of the tree.

C number of nodes without a right sibling in the tree.

D number of leaf nodes in the tree.

Question 14 Explanation:

The function counts leaf nodes for a tree represented using leftMostChild-rightSibling representation. Below is function with comments added to demonstrate how function works. 1

Data Structures Test Bank 2015

Question 15

Level order traversal of a rooted tree can be done by starting from the root and performing

A preorder traversal

B inorder traversal

C depth first search

D breadth first search

Question 15 Explanation:

See this post for details

Question 16

Consider the label sequences obtained by the following pairs of traversals on a labeled binary tree. Which of these pairs identify a tree uniquely ?

```
(i)     preorder and postorder
(ii)    inorder and postorder
(iii)   preorder and inorder
(iv)   level order and postorder
```

A (i) only

B (ii), (iii)

C (iii) only

D (iv) only

Question 16 Explanation:

See http://www.geeksforgeeks.org/if-you-are-given-two-traversal-sequences-can-you-construct-the-binary-tree/ for details

Data Structures Test Bank 2015

Question 17

Let LASTPOST, LASTIN and LASTPRE denote the last vertex visited in a postorder, inorder and preorder traversal, respectively, of a complete binary tree. Which of the following is always true?

A LASTIN = LASTPOST

B LASTIN = LASTPRE

C LASTPRE = LASTPOST

D None of the above

Question 17 Explanation:

See question 1 of http://www.geeksforgeeks.org/data-structures-and-algorithms-set-1/

مع تحيات مبرمجون بلا حدود 2015

تامر أبو الزينات 0786358466

محمود عماد 0785780440