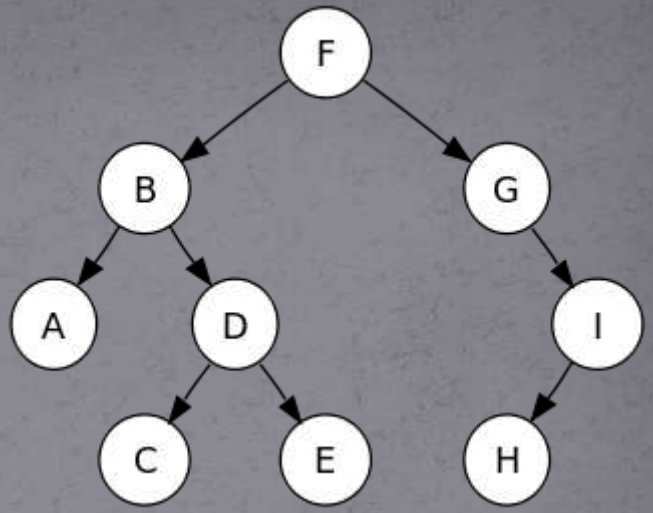
Topic



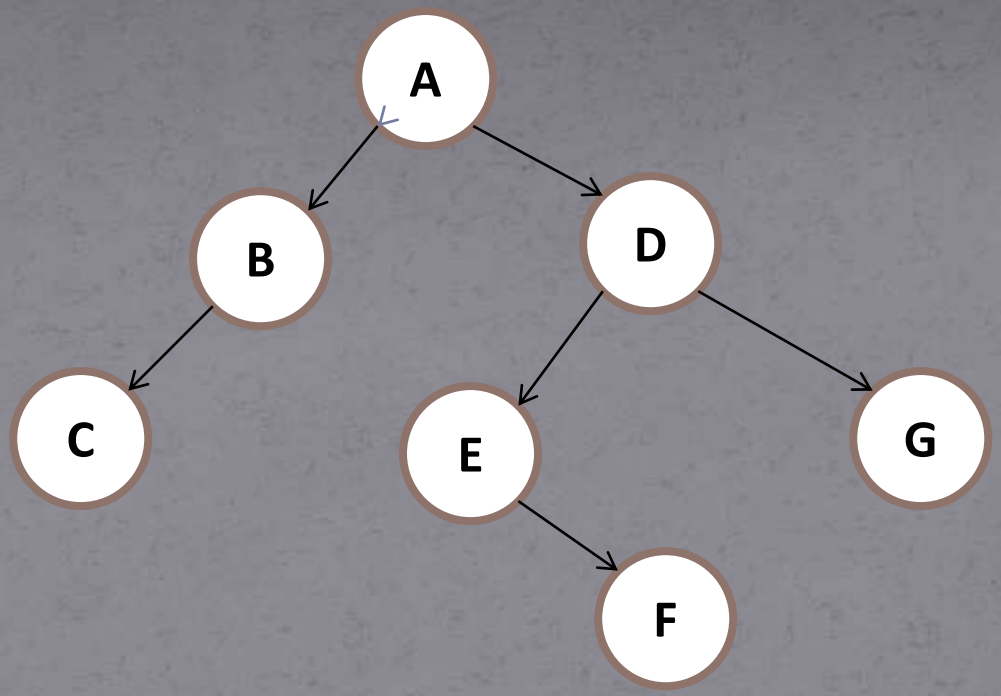
Tree Traversal

Contain

- Binary Tree Traversal
- Preorder
- Inorder
- Postorder
- Example

Binary Tree Traversal

- The most common operations performed on tree structure is that of traversal. This is a procedure by which each node in the tree is processed exactly once in a systematic manner.
- There are three ways of traversing a binary tree.
- 1. Preorder Traversal
- 2. Inorder Traversal
- 3. Postorder Traversal



- Preorder traversal : A B C D E F G
- Inorder traversal : C B A E F D G
- Postorder traversal : C B F E G D A
- Converse Preorder traversal : A D G E F B C
- Converse Inorder traversal : G D F E A B C
- Converse Postorder traversal : G F E D C B A

Preorder

- Preorder traversal of a binary tree is defined as follow
  - Process the root node
  - Traverse the left subtree in preorder
  - Traverse the right subtree in preorder
- If particular subtree is empty (i.e., node has no left or right descendant) the traversal is performed by doing nothing, In other words, a null subtree is considered to be fully traversed when it is encountered.
- The preorder traversal of a tree is given by
- A B C D E F G

Inorder

- The Inorder traversal of a binary tree is given by following steps,
  - Traverse the left subtree in Inorder
  - Process the root node
  - Traverse the right subtree in Inorder
- The Inorder traversal of a tree is given by
- C B A E F D G

Postorder

- The postorder traversal is given by
  - Traverse the left subtree in postorder
  - Traverse the right subtree in postorder
  - Process the root node
- The Postorder traversal of a tree is given by
- C B F E G D A

Converse…

- If we interchange left and right words in the preceding definitions, we obtain three new traversal orders which are called
  - Converse Preorder (A D G E F B C)
  - Converse Inorder (G D F E A B C)
  - Converse Postorder (G F E D C B A)

Procedure : preorder(root)

- Given a binary tree whose root node address is given by pointer variable root and whose node structure is same as described below. This procedure traverses the tree in preorder, in a recursive manner.

```
void preorder (root)
{
   if (root==NULL)
   return;
   printf (“ %d\n ”,  root->info);
   preorder ( root-> left);
   preorder ( root-> right);
}
```

Procedure : inorder(root->left)

- Given a binary tree whose root node address is given by pointer variable “root” and whose node structure is same as described below. This procedure traverses the tree in inorder, in a recursive manner.

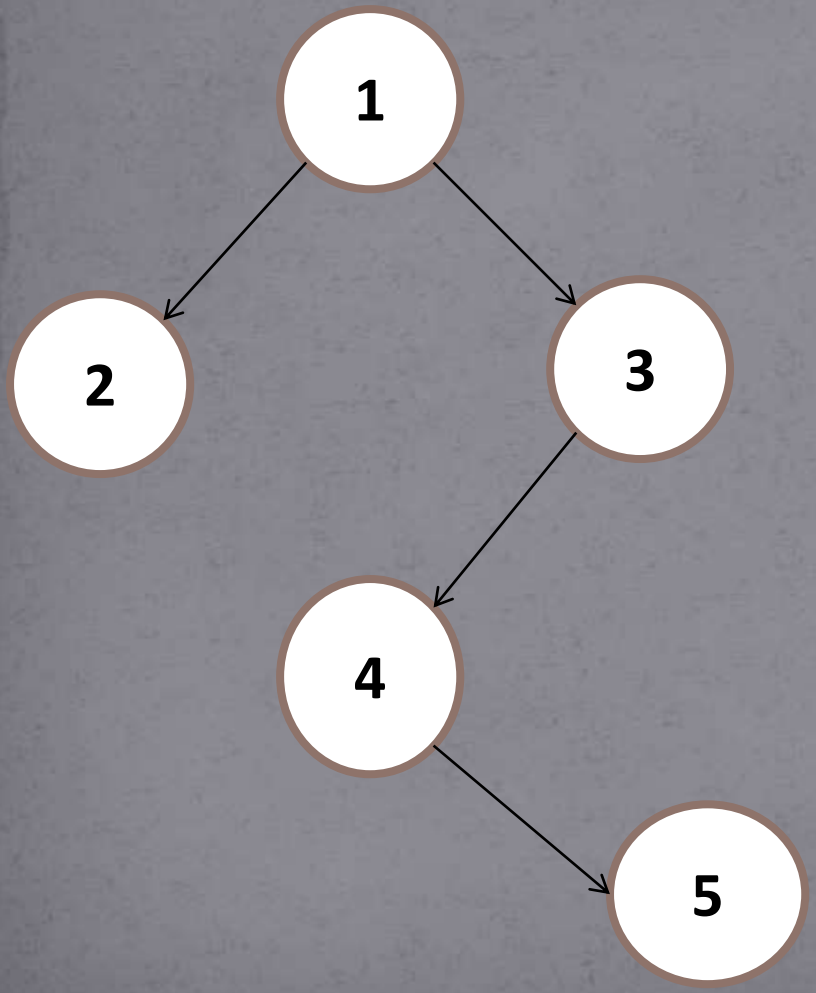
```
void inorder (root->left)
{
   if (root==NULL)
   return;
   inorder ( root);
   printf (“ %d\n ”,  root->info);
   inorder ( root-> right);
}
```

Procedure : postorder(root->left)

- Given a binary tree whose root node address is given by pointer variable “root” and whose node structure is same as described below. This procedure traverses the tree in postorder, in a recursive manner.

```
void postorder (root->left)
{
   if (root==NULL)
   return;
   postorder (root-> right);
   postorder (root);
   printf (“ %d\n ”,  root->info);
}
```

Give traversal order of following tree into Inorder, Preorder and Postorder.



- Inorder: 2 1 4 5 3
- Preorder: 1 2 3 4 5
- Post order: 2 5 4 3 1

References

- Inspiration from Prof. Keyur Suthar and Prof. Rashmin Prajapati
- Notes of DS
- Textbook of DS
- Image from Google images
- Some my own knowledge

Thank You

- #include<stdio.h>
- int main()
- {
- printf(“Thank You”);
- return 0;
- }