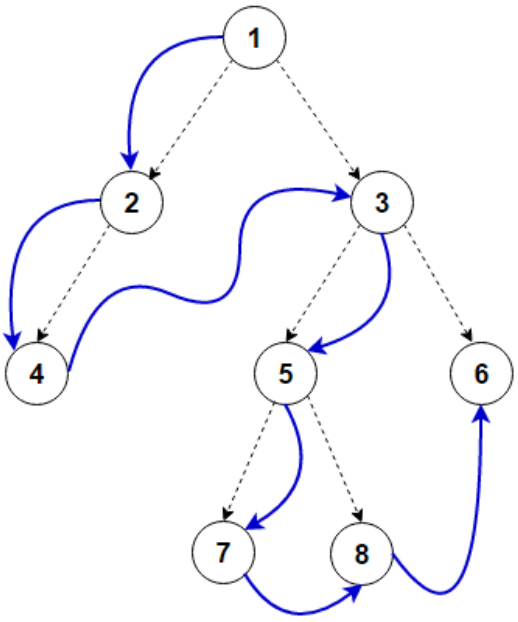
INORDER TRAVERSAL

```
iterativeInorder(node)

s -> empty stack
while (not s.isEmpty() or node != null)
   if (node != null)
      s.push(node)
      node -> node.left
   else
      node -> s.pop()
      visit(node)
      node -> node.right
```

PREORDER TRAVERSAL

In normal pre-order traversal we visit left subtree before the right subtree. If we visit right subtree before visiting the left subtree, it is referred as reverse pre-order traversal.



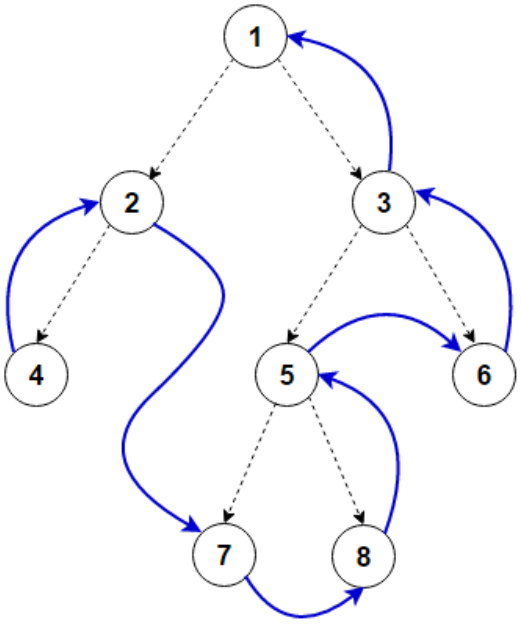
Preorder: 1, 2, 4, 3, 5, 7, 8, 6

```
iterativePreorder(node)

  if (node = null)
    return
  s -> empty stack
  s.push(node)
  while (not s.isEmpty())
    node -> s.pop()
    visit(node)
    if (node.right != null)
       s.push(node.right)
    if (node.left != null)
       s.push(node.left)
```

POSTORDER TRAVERSAL

In normal post-order traversal we visit left subtree before the right subtree. If we visit right subtree before visiting the left subtree, it is referred as reverse post-order traversal.



Postorder: 4, 2, 7, 8, 5, 6, 3, 1

```
iterativePostorder(node)

  s -> empty stack
  t -> output stack
  while (not s.isEmpty())
     node -> s.pop()
     t.push(node)

     if (node.left <> null)
        s.push(node.left)

     if (node.right <> null)
        s.push(node.right)

  while (not t.isEmpty())
     node -> t.pop()
     visit(node)
```