# **Experiment No 4**

#### Title: Consider threading a binary tree using preorder threads rather than in-order threads. Design an algorithm for traversal without using stack and analyze its complexity.

#### Objectives:

1. To understand concept of threading a binary tree.
2. To understand concept of representation of threading a binary tree.

#### Outcomes:

* To effectively perform various operations on TBT.
* To implement TBT and traverse it in preorder sequence.
* To use TBT in various application.

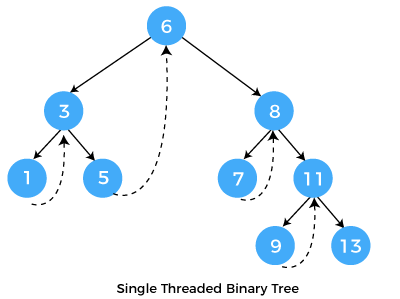
#### Theory:

In the linked representation of binary trees, more than one half of the link fields contain NULL values which results in wastage of storage space. If a binary tree consists of **n** nodes then **n+1** link fields contain NULL values. So in order to effectively manage the space, a method was devised by Perlis and Thornton in which the NULL links are replaced with special links known as threads. Such binary trees with threads are known as **threaded binary trees**. Each node in a threaded binary tree either contains a link to its child node or thread to other nodes in the tree.

**There are two types of threaded binary trees.**

Single Threaded: Where a NULL right pointers is made to point to the inorder successor (if successor exists)

Double Threaded: Where both left and right NULL pointers are made to point to inorder predecessor and inorder successor respectively. The predecessor threads are useful for reverse inorder traversal and postorder traversal.



#### In one-way threaded binary trees, a thread will appear either in the right or left link field of a node. If it appears in the right link field of a node then it will point to the next node that will appear on performing in order traversal. Such trees are called ****Right threaded binary trees****. If thread appears in the left field of a node then it will point to the nodes inorder predecessor. Such trees are called ****Left threaded binary trees.**** Left threaded binary trees are used less often as they don't yield the last advantages of right threaded binary trees. In one-way threaded binary trees, the right link field of last node and left link field of first node contains a NULL. In order to distinguish threads from normal links they are represented by dotted lines.

#### 

#### In two-way threaded Binary trees, the right link field of a node containing NULL values is replaced by a thread that points to nodes inorder successor and left field of a node containing NULL values is replaced by a thread that points to nodes inorder predecessor.

In the above figure of two-way threaded Binary tree, we noticed that no left thread is possible for the first node and no right thread is possible for the last node. This is because they don't have any inorder predecessor and successor respectively. This is indicated by threads pointing nowhere. So in order to maintain the uniformity of threads, we maintain a special node called the **header node**. The header node does not contain any data part and its left link field points to the root node and its right link field points to itself. If this header node is included in the two-way threaded Binary tree then this node becomes the inorder predecessor of the first node and inorder successor of the last node. Now threads of left link fields of the first node and right link fields of the last node will point to the header node.

**Advantages of Threaded Binary Tree:**

* In threaded binary tree, linear and fast traversal of nodes in the tree so there is no requirement of stack. If the stack is used then it consumes a lot of memory and time.
* It is more general as one can efficiently determine the successor and predecessor of any node by simply following the thread and links. It almost behaves like a circular linked list.

**Disadvantages of Threaded Binary Tree:**

* When implemented, the threaded binary tree needs to maintain the extra information for each node to indicate whether the link field of each node points to an ordinary node or the node's successor and predecessor.
* Insertion into and deletion from a threaded binary tree are more time consuming since both threads and ordinary links need to be maintained.

**Software Required:** g++ / gcc compiler- / 64 bit Fedora, eclipse IDE

**Input**: 1.Number of nodes.

2.Node value (Input Data).

**Output:** Threaded Binary tree.

**Conclusion:** This program gives us the knowledge of Threaded Binary tree and all its operations.

**OUTCOME**

**Upon completion Students will be able to:**

* **ELO1:** To effectively perform various operations on TBT.
* **ELO2:** To implement TBT and traverse it in preorder sequence.
* **ELO3:** To use TBT in various application.

**Questions.**

1. What is TBT?
2. What are different operations on TBT?
3. Write a algorithm to travel the TBT using preorder threads.