Date

Expt. No. ABHAYA V 1BM18CS001 Page No.

Q Implement Binomial Heap with functions insert(H,k), getMin(H), extractMin(H)

Pseudo code:

```
struct Node
{ ...
};
Node* mergeBinTrees(Node *b1, Node *b2)
{
        // Function to merge two binomial trees
}
list<Node*> unionBinomialHeap(list<Node*> l1, list<Node*> l2)
{
        // This function performs union operation on
           two binomial heaps
}
list<Node*> adjust(list<Node*> h)
{
        // This function rearranges the heap so that
           heap is in increasing order of degree
           and no two binomial trees have
           same degree in this heap.
}
list<Node*> insert(list<Node*> h, int key)
{       Node *temp = new Node(key);
        return InsertTreeInHeap(h, temp);
}
```

```
Node * getMin (list <Node *> h)
{
    list <Node *> :: iterator it = h.begin();
    Node *temp = *it;
    while (it != h.end())
    {
        if ((*it) -> data < temp -> data)
            temp = *it;
        it++;
    }
    return temp;
}
list <Node *> extractMin ( list <Node *> h )
{
    list <Node *> new_heap, [redacted] x;
    Node *temp;
    temp = getMin (heap);
    list <Node *> :: iterator it;
    it = h.begin();
    while (it != h.end())
    {   if ( *it != temp )
        {
            new_heap.push_back (*it);
        } it++;
    }
    x = removeMinFromTree ([redacted] temp) ;
    new_heap = unionBinomialHeap (new_heap, lo);
    new_heap = adjust (new_heap);
    return new_heap;
}
```

Teacher's Signature :