Jai Jain

1BM18CS040

Date __/__/__

```
Insert (heap, value)
{
    Create new node with key as value
    Create temporary heap
    loop over heap until it becomes NULL
            if degree of original tree in heap <
        degree of temporary tree in heap
            Create new heap and add original tree
        else
            add temporary tree to heap
        if original ~~tree~~ heap has left over heap tree
            add them to new heap
    if temporary heap has left over tree
            add all of them to new heap
    if heap size <1
            return heap
    loop over new heap
        if it's end of heap
            one element remains
        else if degree first tree <
            degree of second tree
                merge
    else if degree are same then
        binomial tree are same in heap
    return heap.
get min(heap)
{
    start from first tree in heap & check
    root of tree. find min of all roots &
    return
}
```

```
Extract min (heap)
{   get min value
      start from first tree in heap
        if tree root is not minimum then c
              create new heap and add tree to heap
          remove minimum else from heap & convert
             tree to heap
          Merge newly created heap without mini
          element and heap that was created earlier
          return merged heap
}
```