Divyank Gupta
1BM18CS030

Write-up

② Binomial Heap :-

```
//Insert Function

public void insert (int value)
{
    if (value >0)
    {
        BinomialHeapNode temp = new BinomialHeapNode (value);
        if (Nodes ==null)
        {
            Nodes = temp ;
            size = 1 ;
        }
        else
        {
            unionNodes (temp) ;
            size ++;
        }
    }
}
```

```
// Extract-Min Function

public int extractMin()
{
    if (Nodes == null)
        return -1 ;

    BinomialHeapNode temp = Nodes , prevTemp=null;
    BinomialHeapNode minNode = Nodes . findMinNode () ;
```

Diyank

Continued . . . .

```
while (temp.key != minNode.key)
{
    prevTemp = temp;
    temp = temp.sibling;
}

if (prevTemp == null)
    Nodes = temp.sibling;
else
    prevTemp.sibling = temp.sibling;

temp = temp.child,
BinomialHeapNode fakeNode = temp;

while (temp != null)
{
    temp.parent = null;
    temp = temp.sibling; }

if (Nodes == null && fakeNode == Null)
    size = 0;

else
{
    if ( Nodes == null && fakeNode != null) {
        Nodes = fakeNode.reverse(null);
        size = Nodes.get size(); }

    Else
    {
        if (Nodes != null && fakeNode == null)
            size = Nodes.get size();

        else
        {
            unionNodes (fakeNode.reverse(null));
            size = Nodes.getsize(); }   }
}
return minNode.key
}
```

Divyank Gupta
1BM18CS030

Continued....

// getMinNode Function

```
public BinomialHeapNode getMinNode()
{
    BinomialHeapNode x = this , y= this ;
    int min = x.Key ;
     while (x!=null) {
       if(x.Key < min) {
           y = x ;
           min = x.Key;
       }
      x = x.sibling ;
    }
    return y ;
}
```