LAB-10

Naman Agarwal
1BM18CS057

```
Node
{
    int data, degree
    Node* child, sibling, parent;
};

merge (Node n1, n2)
    if n1 -> data > n2 -> data.
        swap (n1, n2)

    n2 -> parent = n1
    n2 -> sibling => n1 -> child
    n1 -> child = n2
    n1 -> degree ++;

union ( list <> l1, list <> l2)
    // create a new heap.
    while ( iterator = l1.end & iterator = l2.end)
        if D(L1) <= D(L2)
            // push the value of iterator. L1
        else.
            // push the value of iterator. L2
    // Then push rest of the elements.
```

A. Agarwal

Lab-9

Naman Agarwal
1BM18CS057

```
adjust ( list <> heap)

    if heap.size <=1
        return.

    if  size == 2

        it2 = it1
        it2 ++;
        it3 = heap.end().

    while (it1 -> xend)

        if it2 == heapend()

            it1 ++;

        else if (*it1 -> degree < *it2 -> deg

            it1 ++
            it2 ++
            if (it3 != heapend())

                it3 ++

        ~~else if~~

    return heap.
```