DSA Lab3

Kshitij R

1BM18CS048

```
int findParent(int x, int parent[])
{
    if parent[x] = x
        return x;
    else
        find the root node || recursive call the function
                              until we find the root node
    return parent[x] = findParent(parent[x], parent);
}
void unionSet(int x, int y, int parent[])
{
    if root node of x is not equal to that of
    y , these are not in same set.
    {
        parent[root_x] = root_y & decrement count.
    }
// count variable stores the total no. of 1's in
grid . It gets decremented each time we
create a set
}
int rootIsland(int grid[][20], int x, int y)
{
    // count no. of 1's i.e no of islands) in grid
    & store it in count.
    // create a parent array of six equal to
    [row * col] where row is no. of rows & col. is
    no of cols in grid. each element of
    grid[i][j] = parent[n * i+j]
    parent[i] = i
```

```
for(int i=0; i<row; i++)
{
    for(int j=0; j<col; j++)
    {
        if (grid[i][j]==1)
        {
            i.e current position is on island, we check
            for all 8 neighbours of current
            island and call union func. on current
            island & the neighbour
        }
    }
}
    return count;
}
```