Date : Oct 7, 20

ADS - Lab 3.

Akanksha Haddha
1BM18CS007
Date:
P. No:

Batch - B1.

ques : find no of connected islands in given 2D matrix.

```
int findParent ( int x, int parent [] )
{
    // finds parent of int x in parent arr.
    if  parent [x] = x  ; i.e
            // root node is found
        return x ;

    else
    //  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 [] )
{
    // find root node of x and y.
    if root node of x is not equal to
    that of y  it , these are not in
    same set
        ∴ we make
            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
```

Akanksha Naddha
1BM18CS007.
Date:
P. No:

a set ..

}

int noofIsland (int grid [][20], int x, int y)
{
    // count no of 1's (i.e no of islands) in grid & store it in var count.

    // create a parent array of size 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]
    // initially each element is parent of itself
    ∴ iterate from 0 to [row*col] & initialise

        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 an island, we check for all 8 neighbours of current island and call union func^n on current island & the neighbour.

Akanksha Haddha
1BM18CS007.
Date:
P. No:

```
        // to check if they belong to same
           set or not.
        }
    }
}
  return count.

}.
```