

# ADS Lab-1

## Program-2 find group of islands using find and union operation

Code:

```
#include<iostream>
#include<vector>
using namespace std;

class DisjointUnionSets
{
    vector<int> rank, parent;
    int n;

public:
    DisjointUnionSets(int n)
    {
        rank.resize(n);
        parent.resize(n);
        this->n = n;
        makeSet();
    }
```

```
void makeSet()
{
    for (int i = 0; i < n; i++)
        parent[i] = i;
}
```

```
int find(int x)
{
    if (parent[x] != x)
    {
        parent[x]=find(parent[x]);
    }

    return parent[x];
}
```

```
void Union(int x, int y)
{
    int xRoot = find(x);
    int yRoot = find(y);

    if (xRoot == yRoot)
```

```
if (rank[xRoot] < rank[yRoot])
```

```
    parent[xRoot] = yRoot;
```

```
else if (rank[yRoot] < rank[xRoot])
```

```
    parent[yRoot] = xRoot;
```

```
else
```

```
{
```

```
    parent[yRoot] = xRoot;
```

```
    rank[xRoot] = rank[xRoot] + 1;
```

```
}
```

```
}
```

```
};
```

```
int countIslands(vector<vector<int>>>a)
```

```
{
```

```
    int n = a.size();
```

```
    int m = a[0].size();
```

```
    DisjointUnionSets *dus = new
```

```
    DisjointUnionSets(n * m);
```

```

for (int j = 0; j < n; j++)
{
    for (int k = 0; k < m; k++)
    {

        if (a[j][k] == 0)
            continue;

        if (j + 1 < n && a[j + 1][k] == 1)
            dus->Union(j * (m) + k,
                       (j + 1) * (m) + k);
        if (j - 1 >= 0 && a[j - 1][k] == 1)
            dus->Union(j * (m) + k,
                       (j - 1) * (m) + k);
        if (k + 1 < m && a[j][k + 1] == 1)
            dus->Union(j * (m) + k,
                       (j) * (m) + k + 1);
        if (k - 1 >= 0 && a[j][k - 1] == 1)
            dus->Union(j * (m) + k,
                       (j) * (m) + k - 1);
        if (j + 1 < n && k + 1 < m &&
            a[j + 1][k + 1] == 1)
            dus->Union(j * (m) + k,
                       (j + 1) * (m) + k + 1);
        if (j + 1 < n && k - 1 >= 0 &&

```

```

        a[j - 1][k + 1] == 1)
        dus->Union(j * m + k,
                    (j - 1) * m + k + 1);
    if (j - 1 >= 0 && k - 1 >= 0 &&
        a[j - 1][k - 1] == 1)
        dus->Union(j * m + k,
                    (j - 1) * m + k - 1);
    }
}

```

```

int *c = new int[n * m];
int numberOfIslands = 0;
for (int j = 0; j < n; j++)
{
    for (int k = 0; k < m; k++)
    {
        if (a[j][k] == 1)
        {
            int x = dus->find(j * m + k);

            if (c[x] == 0)
            {
                numberOfIslands++;
                c[x]++;
            }
        }
    }
}

```

```

        else
            c[x]++;
    }
}
}
return numberOfIslands;
}

```

```

int main(void)
{
    vector<vector<int>>a = {{1, 1, 0, 0, 0},
                           {0, 1, 0, 0, 1},
                           {1, 0, 0, 1, 1},
                           {0, 0, 0, 0, 0},
                           {1, 0, 1, 0, 1}};
    cout << "Number of Islands is: "
          << countIslands(a) << endl;
}

```

Result

compiled and executed in 1.43 sec(s)

**Number of Islands is: 5**