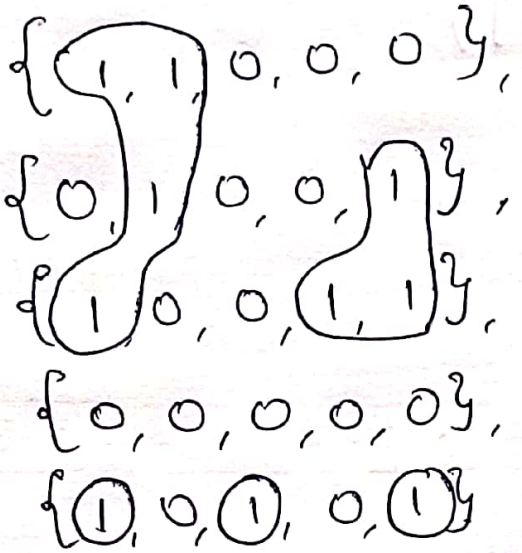
Pseudocode

Sahana L
1BM18CS089



Total 5 islands.

```
// Implementation using Disjoint Sets.
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)
  {
    return find (parent[x]);
  }
  return x;
}

void Union (int x, int y)
{
  int xRoot = find(x);
  int yRoot = find(y);
  if (xRoot == yRoot)
    return;
  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;
            // Check all 8 neighbours and do a Union with neighbour
            set if neighbour is also 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;
```