Vinay Kumar 1BM18CS153

DATE: PAGE:

Lab-3

Disjoint set implementation of boolean 2D Matrix

```
Static int countIsland(int a[][])
{
   int  n = a.length;
   int  m = a[0].length;

   DisjointUnionSet  dus = new
   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 <m && 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)
             { numberOf~~ed~~Islands++;
                c[x]++;
             }
             else
                c[x]++;
           }
   } }
     return numberOfIslands;
 }
}
```