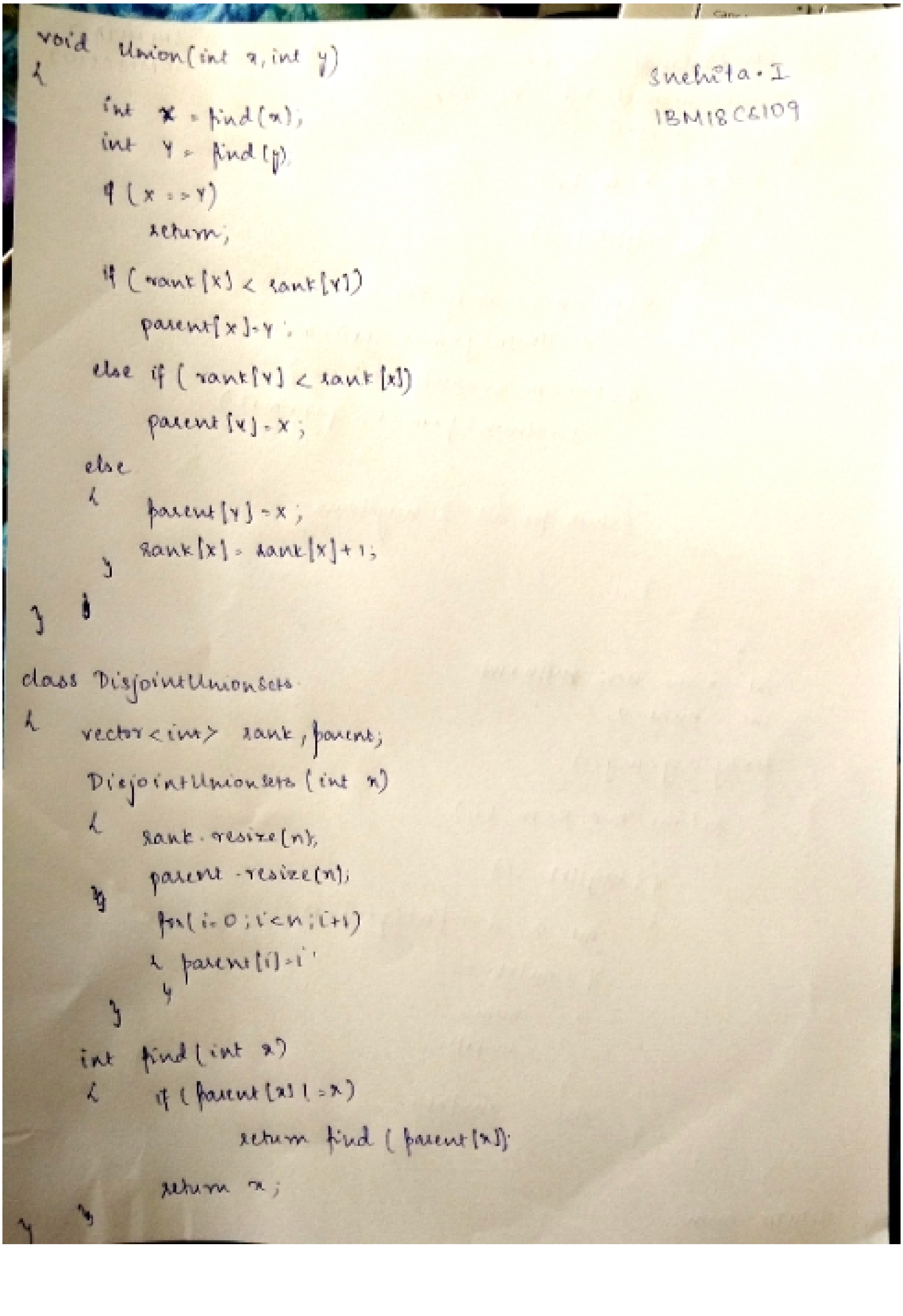

Snehita . I
1BM18CS109

```
void  Union(int x, int y)
{
    int  X = find(x);
    int  Y = find(y)
    if (X == Y)
        return;
    if (rank[X] < rank[Y])
        parent[X]=Y;
   else if ( rank[Y] < rank[X])
        parent[Y] = X ;
   else
   {
        parent[Y] = X ;
       Rank[X] = rank[X]+1;
   }
}  }

class DisjointUnionSets
{   vector<int>  rank, parent;
    DisjointUnionSets (int  n)
    {
        Rank.resize(n);
        parent .resize(n);
    }
        for(i=0;i<n;i++)
        { parent[i]=i
        }
     }
   int  find(int  x)
   {    if (parent[x] != x)
                return find ( parent[x]);
        return  x;
   }
}
```

SNEHITA · I
1BM18CS109

```
int   countNoofIslands ( vector <vector <int >>  arr)
{     DisjointUnionSet * d = ne
      for( j=0; j<n ; j++)
      {
        for( k=0 ; k<m, k++)
        {
            if ( arr[j][k] != 0)
            {.
                  if (j+1 < n && arr[j+1][k] == 1)
                       d-> Union( j* m+k , (j+1) * m + k)
                  if ( k+1 < m  &&  arr[j][k+1] ==1)
                       d-> Union( j*m+k,  (j+1) * m+k ),

                  :
                  // check for all  8  neighbours
                  :
            }.
        }
      }
     int  * ans = new int[n*m];
     int    num=0;
    for (j=0; j<n; j++)
    {
       for( int  k=0; k<m; k++)
       {
            if ( arr[j][k] == 1)
            {
                 int   x = d -> find(j * m + k)
                 if (ans[x] == 0)
                  {     num++;
                        ans[x]+1 ;
                  }
                 else    ans[x]++ ;
            }
       }
    }
}  return  num;
```