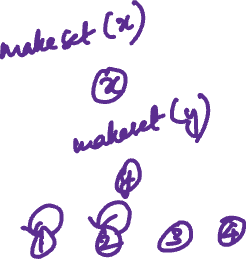
#include <stdio.h>

struct DisjSet {



int parent[10];



int rank[10]; //rank[i] is the height of the tree representing the set



int n;

}dis;



// Creates n single item sets



void makeSet()



{



for (int i = 0; i < dis.n; i++) {



dis.parent[i] = i;



dis.rank[i]=0;



}



}

//Displays Disjoint set



void displaySet()

{ printf("\nParent Array\n");



for (int i = 0; i < dis.n; i++) {

printf("%d ",dis.parent[i]); }



printf("\nRank Array\n");



for (int i = 0; i < dis.n; i++)



{

printf("%d ",dis.rank[i]);



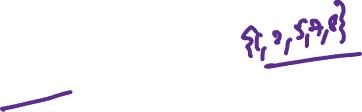
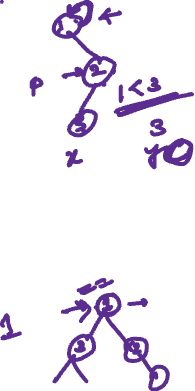
}

printf("\n");

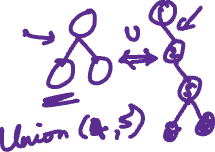
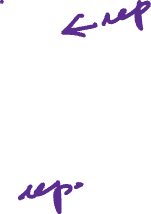


}

// Finds set of given item x



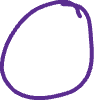
int find(int x)



{

// Finds the representative of the set

// that x is an element of



if (dis.parent[x] != x) {



// if x is not the parent of itself

// Then x is not the representative of

// his set,



dis.parent[x] = find(dis.parent[x]);



// so we recursively call Find on its parent



// and move i's node directly under the

// representative of this set

}

return dis.parent[x];



}

// Do union of two sets represented

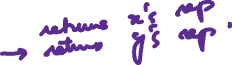
// by x and y.

void Union(int x, int y)

{

// Find current sets of x and y

int xset = find(x);



int yset = find(y);

// If they are already in same set

if (xset == yset)

return;

// Put smaller ranked item under

// bigger ranked item if ranks are

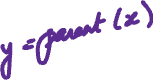
// different



if (dis.rank[xset] < dis.rank[yset]) {



dis.parent[xset] = yset;



dis.rank[xset]=-1;



}

else if (dis.rank[xset] > dis.rank[yset]) {

dis.parent[yset] = xset;



dis.rank[yset]=-1;

}

// If ranks are same, then increment

// rank.

else {

dis.parent[yset] = xset;



dis.rank[xset] = dis.rank[xset] + 1;



dis.rank[yset]=-1;



}

}

int main()

{ int n,x,y;

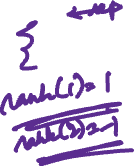
printf("How many elements ?");



scanf("%d",&dis.n);



makeSet();



int ch,wish;



do

{

printf("\n\_\_\_\_MENU\_\_\_\_\n");



printf("1. Union \n2.Find\n3.Display\n");

printf("enter choice\n");

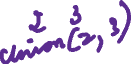
scanf("%d",&ch);

switch(ch)

{

case 1: printf("Enter elements to perform union");

scanf("%d %d",&x,&y);



Union(x, y);



break;

case 2: printf("Enter elements to check if connected components");

scanf("%d %d",&x,&y);



if (find(x) == find(y))



printf("Connected components\n") ;

else

printf("Not connected components \n") ;



break;

case 3: displaySet();



break;

}

printf("\nDo you wish to continue ?(1/0)\n");

scanf("%d",&wish);

}while(wish==1);

return 0;

}