Disjointset

#include <stdio.h>

struct disjset {

int parent[10];

int rank[10];

int n;

}dis;

void makeset()

{

int i;

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

{

dis.parent[i] = i;

dis.rank[i] = 0;

}

}

void displayset()

{

int i;

printf("\nparent array\n");

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

{

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

}

printf("\nrank array\n");

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

{

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

}

printf("\n");

}

int find(int x)

{

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

{

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

}

return dis.parent[x];

}

void uion(int x,int y)

{

int xset = find(x);

int yset = find(y);

if(xset == yset)

return;

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;

}

else

{

dis.parent[yset] = xset;

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

dis.rank[yset] = -1;

}

}

int main()

{

int n,x,y,ch,wish;

printf("how many elements?");

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

makeset();

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 the elements to perform union");

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

uion(x,y);

break;

case 2:printf("enter the elements to check if connected components\n");

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("\n do you wish to continue?(1/10)\n");

scanf("%d",&wish);

}

while(wish==1);

return 0;

}

Output:

