ILAHIA COLLEGE OF ENGINEERING AND TECHNOLOGY

DEPARTMENT OF MCA

FIRST SEMESTER MCA (2020 SCHEME) PRACTICAL

EXAMINATION JUNE 2021

20MCA135 DATA STRUCTURE LAB

Date : 30-6-2021 Time : 1 Pm-

Submitted by,

Gopika Sabu

ICE20MCA2023

Batch : 1

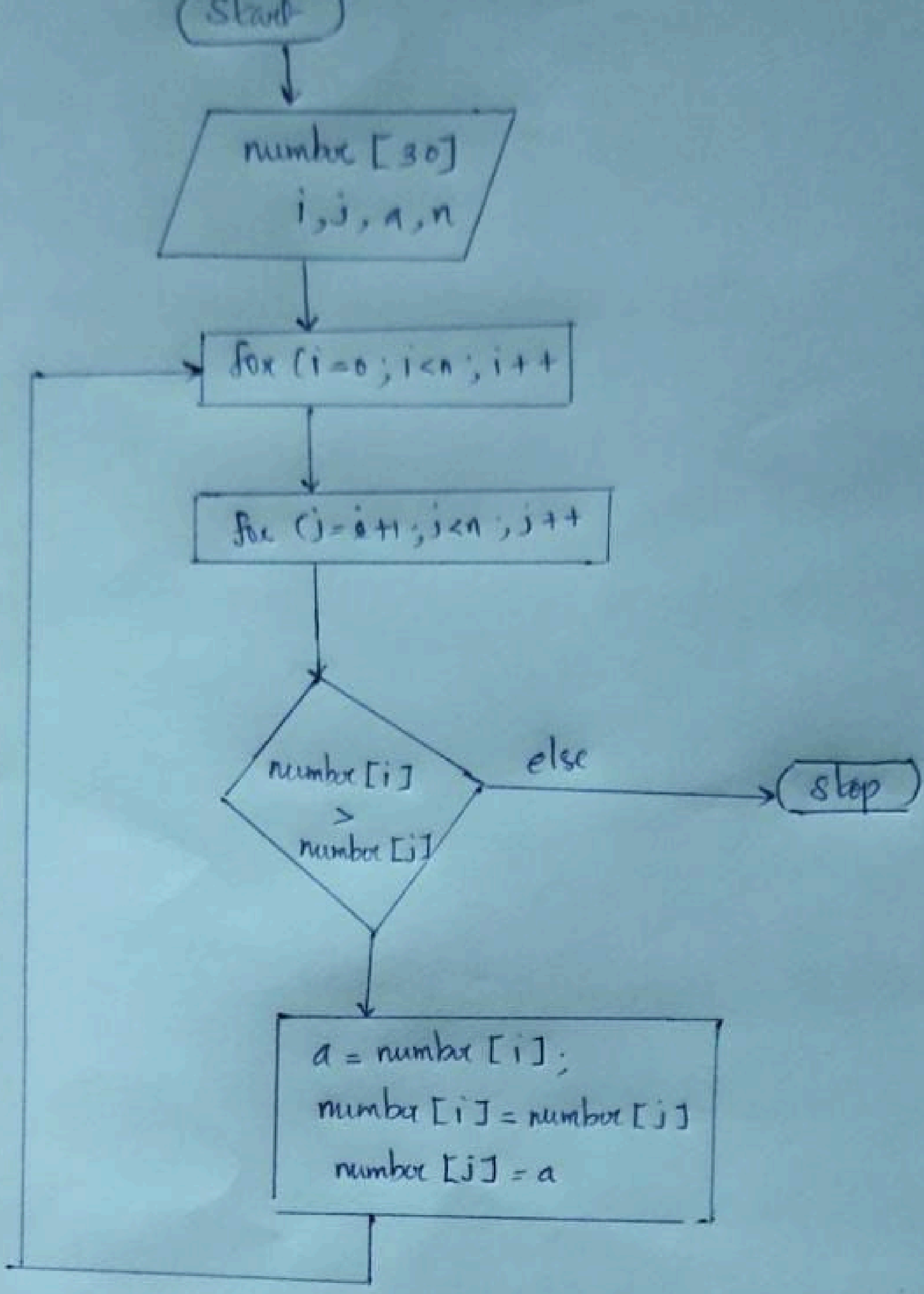
1) SORTING OF AN INTEGER ARRAY ?

```
# include <stdio.h>
# include <conio.h>
void main ()
 {
 int i, j, a, n, number [30];
  clrscr();
  Printf (" Enter the value of N\n");
   Scanf (" %d", &n);
  Printf (" Enter the numbers \n");
  for (i = 0; i<n ;++i)
  Scanf (" %d ", & number [i]);
  for (i=0; i<n ; ++i)
   {
  for (j=i+1 ; j<n ;++j)
  {
  if (number [i] > number [j] )
   {
   a = number [i];
  number [i] = number [j];
   number [j] = a;
Printf (" the numbers arranged in ascending order are
            given below \n");
for (i=0; i<n; ++i)
Printf ("%d\n", number [i]);
getch();
}
```

__Output__

```
Enter the value of N
 4
Enter the Numbers
 78    45    10    3
the numbers arranged in ascending order are given below
          3
          10
          45
          78
```

Flow chart


Start
number [30]
i, j, a, n
for (i=0; i<n; i++
for (j=i+1; j<n; j++
number [i] > number [j]
else
stop
a = number [i];
number [i] = number [j]
number [j] = a

2) Implement Disjoint set operations ?

```
# include <stdio.h>
# include <conio.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 ("\n Parent 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 Union (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 x, y, n, ch, wish ;
clrscr ();
Printf ( " How many elements? ");
Scanf (" %d ", & dis. n);
makeset ();
do
{
Printf ("\n__menu __\n");
Printf (" 1. union \n 2. Find \n 3. Display \n");
Printf (" enter choice \n");
Scanf(" %d ", &ch);
Switch (ch)
{
```

```
Case 1 :
Printf (" Enter elements to preform 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");
break;
case 3:
displaySet ();
break;
}
Printf ("\n Do you wish to continue ? (1/0)\n");
Scanf (" %d", &wish);
}
while (wish == 1);
return 0;
}
```

Output

```
How many elements ? 4
  — Menu —
1. union
2. Find
3. Display
```

```
enter choice
 1
Enter elements to perform union : 2 3
 Do you wish to continue ? (1/0)
  1
 - Menu -
  1. union
  2. Find
  3. Display
  enter choice
   2
 Enter elements to check if connected components : 1 4
  Not connected components
 Do you wish to continue ? (1/0)
   1
-Menu -
1. union
2. find
3. Display
enter choice
  3
Parent Array
   0122
Rank Array
   001-1
Do you wish to continue ? (1/0)
   0
```

Algorithm

```
Algorithm Weight union (i,j)
{
P[i] = - count [i] & p[j] = - count [j]
 {
 Temp = -p[i] + p[j]
  If (P[i] > p[j]
   {
 P[i] = j;
  P[j] = temp;
  }
 Simple find (1)
 {
 While (P[1]>0)
 Do
 1 = P[i];
  Return I;
  }
 Algorithm collapsing find (i)
 {
 r = 1;
 While (P[r] > 0) do
    r = p[r];
   While (i≠r) do
    {
   s = P[i];
   P[i] = r;
   i = s; }
   return r; }
```