13-5-21

Q=> BFS Traversal

```
# include <stdio.h>
# include <time.h>
int a[10][10], q[10], visited[10], n,
f = 0, r = -1;
void bfs(int v){
    int i;
    for (i=1; i<=n; i++)
        if (a[v][i] && !visited[i])
            q[++r] = i;
    if(f<=r) {
        visited [q[f]] = 1;
        bfs (q[f++]);
    }

}
int main(){
    int v, i, j;
    printf("\n Enter the no. of vertices:");
    scanf("%d", &n);
    printf("\n Enter matrix : \n");
    for (i=1; i<=n; j++)
        printf("Enter row %d : \n", i);
        for (j=1; j<=n; j++)
            scanf ("%d", &a[i][j]);
    }
```

```
for (i=1; i<=n; i++) {
    q[i] = 0;
    visited[i]=0;
}
printf("Enter beginning vertex :");
scanf("%d", &v);
double bfs_time = 0.0;
clock_t begin = clock();
bfs(v);
clock_t end = clock();
bfs_time += (double)(end - begin) / CLOCKS_
                            PER_SEC;
printf("Visitable Nodes are : \n");
for(i=1; i<=n; i++) {
    if(visited[i])
        printf("%d", i);
    else
        printf("\n BFS isnt possible");
printf("\n n= %d \n Time : %f\n", n,
            bfs_time);
return 0;
}
```

d> Insertion Sort

```
#include<stdio.h>
```

```
# include<time.h>
void intersection (int a[], int n){
    for(int i=1; i<n; i++){
        int j, key;
        key = a[i];
        j= i-1;
        while (key < a[j] && j>=0){
            a[j+1] = a[j];
            --j;
        }
        a[j+1] = Key;
    }
}
void display(int a[], int n){
    for( int i=0; i<n; i++){
        printf(" %d ", a[i]);
    }
}
int main(){
    int n, i, a[10];
    printf(" Enter size of array : ");
    scanf(" %d ", &n);
    printf(" Enter values of array\n");
    for(i=0; i<n; i++){
        scanf(" %d ", &a[i]);
    }
double is_time = 0.0;
clock_t begin = clock();
insertion (a, n);
clock_t end = clock();
is_time += (double)(end - begin)/
                        Clocks_PER_SEC;
display (a, n);
printf("\n n = %d \t Time : %f \n",
                    n, is_time);
return n;
}
```