ASSIGNMENT 7

NAME: SHISHU REG:2020CA089

Q1)Given an array A of size N and a number K(where k<N). Find the K-th largest/smallest number in the array, i.e., K-th order statistic.

CODE:

```
#include <iostream>

#include <vector>

#include <queue>

using namespace std;


// Function to find the k'th smallest element in an array using min-heap

int findKthSmallest(vector<int> const &input, int k)

{

    // base case

    if (input.size() < k) {

        exit(-1);

    }


    // create an empty min-heap and initialize it with all elements

    // `use std::greater` as the comparison function for min-heap

    priority_queue<int, vector<int>, greater<int>> pq(input.begin(), input.end());


    // pop from min-heap exactly `k-1` times

    while (--k) {

        pq.pop();

    }
```

```
    // return the root of min-heap
    return pq.top();
}

int main()
{   int N,k;
    cout<<"Enter size of Array: ";
    cin>>N;
    vector<int> input;
    cout<<"enter elements of array: ";
    for(int i=0;i<N;i++){
        int a;
        cin>>a;
        input.push_back(a);
    }
    cout<<"Enter value of k(k<N) ";
    cin>>k;
    cout << "k'th smallest array element is " << findKthSmallest(input, k);

    return 0;
}
```

OUTPUT:-

```
input
Enter size of Array: 6
enter elements of array: 4 6 3 9 12 54
Enter value of k(k<N) 4
k'th smallest array element is 9

...Program finished with exit code 0
Press ENTER to exit console.
```

Q2)Write a C Program to implement BFS.

Code:

```
#include<stdio.h>

int a[20][20], q[20], visited[20], n, i, j, f = 0, r = -1;


void bfs(int v) {

	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++]);

	}

}


void main() {

	int v;

	printf("\n Enter the number of vertices:");
```

```
        scanf("%d", &n);

        for(i=1; i <= n; i++) {
                q[i] = 0;
                visited[i] = 0;
        }

        printf("\n Enter graph data in matrix form:\n");
        for(i=1; i<=n; i++) {
                for(j=1;j<=n;j++) {
                        scanf("%d", &a[i][j]);
                }
        }

        printf("\n Enter the starting vertex:");
        scanf("%d", &v);
        bfs(v);
        printf("\n The node which are reachable are:\n");

        for(i=1; i <= n; i++) {
                if(visited[i])
                        printf("%d\t", i);
                else {
                        printf("\n Bfs is not possible. Not all nodes are reachable");
                        break;
                }
        }
}
```

OUTPUT:-

```
 Enter the number of vertices:3

 Enter graph data in matrix form:
2 4 5
9 6 1
4 9 3

 Enter the starting vertex:3

 The node which are reachable are:
1       2       3

...Program finished with exit code 0
Press ENTER to exit console.
```