

DAA Assignment No 06

Q) Write a C/C++ program to implement

1) Insertion Sort

2) DFS

3) BFS

1) Insertion Sort:

-- > Program Code:

```
Go Run Terminal Help insertionSort.cpp - DAA - Visual Studio Code
insertionSort.cpp X bfs.cpp
p6 > insertionSort.cpp > main()
1  #include <iostream>
2  using namespace std;
3
4  void insertionSort(int arr[], int n) {
5      for (int i = n - 1; i > 0; i--) {
6          int j = i - 1, key = arr[i];
7          while (j >= 0 && arr[j] > key) {
8              arr[j+1] = arr[j];
9              j--;
10         }
11         arr[j+1] = key;
12     }
13 }
14
15 int main() {
16     int arr[] = { 10,30,15,5,25,20 };
17     int n = sizeof(arr) / sizeof(arr[0]);
18
19     insertionSort(arr, n);
20
21     for (int i = 0; i < n; i++)
22         cout << arr[i] << " ";
23     cout << endl;
24
25     return 0;
26 }
```

Output:

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
● PS C:\Users\DELL\Desktop\DAA> cd 'c:\Users\DELL\Desktop\DAA\p6\output'
● PS C:\Users\DELL\Desktop\DAA\p6\output> & .\insertionSort.exe
5 10 30 15 20 25
○ PS C:\Users\DELL\Desktop\DAA\p6\output> █
```

2)DFS

--> Program Code:

```
Go Run Terminal Help dfs.cpp - DAA - Visual Studio Code
insertionSort.cpp bfs.cpp dfs.cpp x
p6 > dfs.cpp > DFS(int, vector<int> [], vector<bool>&v)
1  #include<bits/stdc++.h>
2  using namespace std;
3  // This function implements DFS using Decrease and Conquer approach
4  void DFS(int u, vector<int> adj[], vector<bool> &visited){
5      visited[u] = true;
6      cout << u << " ";
7      // traverse all adjacent vertices of vertex u
8      for(int v : adj[u]){
9          if(!visited[v]){
10             DFS(v, adj, visited); // recursively call DFS on the adjacent vertex
11         }
12     }
13 }
14 int main(){
15     int V, E; // input the number of vertices and edges
16     cout << "Enter the number of vertices: ";
17     cin >> V;
18     cout << "Enter the number of edges: ";
19     cin >> E;
20
21     // create an adjacency list representation of the graph
22     vector<int> adj[V];
23     cout << "Enter the edges (u v) :> << endl;
24     for(int i = 0; i < E; i++){
25         int u, v;
26         cin >> u >> v;
27         adj[u].push_back(v);
28         adj[v].push_back(u);
29     }
30     vector<bool> visited(V, false); // mark all vertices as unvisited
31     for(int u = 0; u < V; u++){ // perform DFS from all unvisited vertices
32         if(!visited[u]){
33             DFS(u, adj, visited);
34         }
35     }
36     return 0;
37 }
```

3)BFS

--> Program Code:

Go Run Terminal Help bfs.cpp - DAA - Visual Studio Code

insertionSort.cpp

bfs.cpp

```
p6 > bfs.cpp > main()
1  #include <iostream>
2  #include <queue>
3  #include <vector>
4  #include <algorithm>
5
6  using namespace std;
7
8  void BFS(vector<int> adj[], int start)
9  {
10     queue<int> q;
11     vector<bool> visited(adj->size(), false);
12
13     visited[start] = true;
14     q.push(start);
15
16     while (!q.empty())
17     {
18         int current = q.front();
19         cout << current << " ";
20         q.pop();
21
22         for (int i = 0; i < adj[current].size(); i++)
23         {
24             int neighbor = adj[current][i];
25             if (!visited[neighbor])
26             {
27                 visited[neighbor] = true;
28                 q.push(neighbor);
29             }
30         }
31     }
32 }
```

```
33
34 int main()
35 {
36     int vertices, edges;
37     cout << "Enter the number of vertices: ";
38     cin >> vertices;
39     cout << "Enter the number of edges: ";
40     cin >> edges;
41
42     vector<int> adj[vertices + 1];
43
44     cout << "Enter the edges: " << endl;
45     for (int i = 0; i < edges; i++)
46     {
47         int u, v;
48         cin >> u >> v;
49         adj[u].push_back(v);
50         adj[v].push_back(u);
51     }
52
53     cout << "Enter the starting vertex: ";
54     int start;
55     cin >> start;
56
57     BFS(adj, start);
58
59     return 0;
60 }
```