


EXPERIMENT NO.07

(1)BFS

```
C grapghbfs.c > main()
1  #include <stdio.h>
2
3  int a[20][20], q[20], visited[20], n, f = -1, r = -1;
4
5  void bfs(int v) {
6      int i;
7      for (i = 0; i < n; i++) {
8          if (a[v][i] != 0 && visited[i] == 0) {
9              r = r + 1;
10             q[r] = i;
11             visited[i] = 1;
12             printf("%d ", i);
13         }
14     }
15     f = f + 1;
16     if (f <= r) {
17         bfs(q[f]);
18     }
19 }
20
21 int main() {
22     int v, i, j;
23     printf("\nEnter number of vertices: ");
24     scanf("%d", &n);
25     for (i = 0; i < n; i++) {
26         visited[i] = 0;
27     }
28     printf("\nEnter graph data in matrix form:\n");
29     for (i = 0; i < n; i++) {
30         for (j = 0; j < n; j++) {
31             scanf("%d", &a[i][j]);
32         }
33     }
34     printf("\nEnter the starting vertex: ");
35     scanf("%d", &v);
36     f = 0;
37     r = 0;
```

graphbfs.c >  breadth_first_search(int)

```
5 void bfs(int v) {
6     f = f + 1;
7     if (f <= r) {
8         bfs(q[f]);
9     }
10 }
11
12 int main() {
13     int v, i, j;
14     printf("\nEnter number of vertices: ");
15     scanf("%d", &n);
16     for (i = 0; i < n; i++) {
17         visited[i] = 0;
18     }
19     printf("\nEnter graph data in matrix form:\n");
20     for (i = 0; i < n; i++) {
21         for (j = 0; j < n; j++) {
22             scanf("%d", &a[i][j]);
23         }
24     }
25     printf("\nEnter the starting vertex: ");
26     scanf("%d", &v);
27     f = 0;
28     r = 0;
29     q[r] = v;
30     visited[v] = 1;
31     printf("%d ", v);
32     bfs(v);
33     if (f == n - 1) {
34         printf("\nBFS not possible");
35     }
36     printf("\n");
37     return 0;
38 }
```

OUTPUT:

```
Enter number of vertices: 3
Enter graph data in matrix form:
1 0 1
0 1 0
1 1 0
Enter the starting vertex: 2
2 0 1
```

(2)DFS

```

graphDFS.c > DFS(int)
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  int G[20][20], visited[20], v;
5
6  void DFS(int t) {
7      int j;
8      visited[t] = 1;
9      printf(" %d ->", t + 1);
10     for (j = 0; j < v; j++) {
11         if (G[t][j] == 1 && visited[j] == 0) {
12             DFS(j);
13         }
14     }
15 }
16
17 int main() {
18     int i, j, e, v1, v2, source;
19
20     printf("Enter the number of edges: ");
21     scanf("%d", &e);
22
23     printf("Enter the number of vertices: ");
24     scanf("%d", &v);
25
26     for (i = 0; i < v; i++) {
27         for (j = 0; j < v; j++) {
28             G[i][j] = 0;
29         }
30         visited[i] = 0;
31     }
32
33     for (i = 0; i < e; i++) {
34         printf("Enter the edges (format: V1 V2): ");
35         scanf("%d %d", &v1, &v2);
36         G[v1 - 1][v2 - 1] = 1;
37     }

```

graphDFS.c > DFS(int)

```
7 int main() {
8     G[v1][v2] = 0;
9 }
10 visited[i] = 0;
11 }
12
13 for (i = 0; i < e; i++) {
14     printf("Enter the edges (format: V1 V2): ");
15     scanf("%d %d", &v1, &v2);
16     G[v1 - 1][v2 - 1] = 1;
17 }
18
19 printf("\nAdjacency Matrix:\n");
20 for (i = 0; i < v; i++) {
21     for (j = 0; j < v; j++) {
22         printf(" %d", G[i][j]);
23     }
24     printf("\n");
25 }
26
27 printf("\nEnter the source vertex: ");
28 scanf("%d", &source);
29
30 printf("DFS Traversal starting from vertex %d:\n", source);
31 DFS(source - 1);
32
33 return 0;
34 }
35
```

OUTPUT:

```
Enter the number of edges: 3
Enter the number of vertices: 4
Enter the edges (format: V1 V2): 5 7
Enter the edges (format: V1 V2): 2 9
Enter the edges (format: V1 V2): 1 6

Adjacency Matrix:
0 0 0 0
0 0 0 0
0 0 0 0
0 0 0 0

Enter the source vertex: 3
DFS Traversal starting from vertex 3:
3 ->
```

SUBMITTED BY:SHAMAL BHANUDAS DEORE
CLASS/DIV:SY-IT-A
ROLL NO.18