

## DS WEEK-10

```
Start here x bfs.c x *dfs.c x hash.c x BST.c x
1  #include <stdio.h>
2  int queue[20], front=-1, rear=-1;
3  int visited[20];
4  int n, graph [20][20];
5
6  void enqueue(int v)
7  {
8      if(rear==n-1)
9          return ;
10     if (front==-1)
11         front=0;
12     queue[++rear]=v;
13 }
14 int dequeue()
15 {
16     return queue[front++];
17 }
18 void bfs(int start)
19 {
20     int i, v;
21     enqueue(start);
22     visited[start]=1;
23
24     while(front<=rear)
25     {
26         v=dequeue();
27         printf("%d", v);
28
29         for(i=0; i<n; i++)
30         {
31             if(graph[v][i]==1 && visited[i]==0)
32             {
33
34                 enqueue(i);
35                 visited[i]=1;
36             }
37         }
38     }
39 }
```

```

40     int main()
41     {
42         int i, j, start;
43         printf("enter number of vertices:");
44         scanf("%d", &n);
45
46         printf("enter adjacency matrix:\n");
47         for(i=0; i<n; i++)
48             for(j=0; j<n; j++)
49                 scanf("%d", &graph[i][j]);
50
51         for(i=0; i<n; i++)
52
53             visited[i]=0;
54
55             printf("enter starting vertex:");
56             scanf("%d", &start);
57
58             printf("BFS traversal:");
59
60             bfs(start);
61
62         return 0;
63     }
64

```

```

"C:\Users\BMSCE\Documents" x + v
enter number of vertices:4
enter adjacency matrix:
1 0 1 1
1 0 0 0
0 1 0 1
1 1 1 1
enter starting vertex:0
BFS traversal:0231
Process returned 0 (0x0)   execution time : 27.363 s
Press any key to continue.

```

```
1  #include <stdio.h>
2
3  int graph[20][20], visited[20];
4  int n;
5
6  void DFS(int v)
7  {
8      int i;
9      visited[v] = 1;
10
11     for(i = 0; i < n; i++)
12     {
13         if (graph[v][i] == 1 && visited[i] == 0)
14             DFS(i);
15     }
16 }
17
18 int main()
19 {
20     int i, j;
21     printf("Enter number of vertices: ");
22     scanf("%d", &n);
23
24     printf("Enter adjacency matrix:\n");
25     for(i = 0; i < n; i++)
26         for(j = 0; j < n; j++)
27             scanf("%d", &graph[i][j]);
28
29     for(i = 0; i < n; i++)
30         visited[i] = 0;
31
32     DFS(0);
33
34     for(i = 0; i < n; i++)
35     {
36         if (visited[i] == 0)
37         {
38             printf("Graph is not connected\n");
39             return 0;
40         }
41     }
42
43     printf("Graph is connected\n");
44     return 0;
45 }
```



"C:\Users\BMSCE\Documents" X



Enter number of vertices: 4

Enter adjacency matrix:

1 0 1 1

1 0 0 0

0 1 0 1

1 1 1 1

Graph is connected

Process returned 0 (0x0) execution time : 27.848 s

Press any key to continue.

|

Start here X bfs.c X dfs.c X \*hash.c X BST.c X

```
1  #include <stdio.h>
2  #define SIZE 10
3
4  int hashTable[SIZE];
5
6  void insert(int key)
7  {
8      int index = key % SIZE;
9
10     // Linear probing
11     while (hashTable[index] != -1)
12     {
13         index = (index + 1) % SIZE;
14     }
15     hashTable[index] = key;
16 }
17
18 void display()
19 {
20     int i;
21     printf("\nHash Table:\n");
22     for(i = 0; i < SIZE; i++)
23     {
24         if(hashTable[i] == -1)
25             printf("%d: empty\n", i);
26         else
27             printf("%d: %d\n", i, hashTable[i]);
28     }
29 }
30
31 int main()
32 {
33     int n, key, i;
34
35     // Initialize table with -1 (empty)
36     for(i = 0; i < SIZE; i++)
37         hashTable[i] = -1;
38
39     printf("Enter number of employee records: ");
40     scanf("%d", &n);
41
42     printf("Enter 4-digit employee keys:\n");
43     for(i = 0; i < n; i++)
44     {
45         scanf("%d", &key);
46         insert(key);
47     }
48     display();
49     return 0;
50 }
```



"C:\Users\BMSCE\Documents" X



Enter number of employee records: 5

Enter 4-digit employee keys:

1011

1050

9999

2201

1001

Hash Table:

0: 1050

1: 1011

2: 2201

3: 1001

4: empty

5: empty

6: empty

7: empty

8: empty

9: 9999

Process returned 0 (0x0) execution time : 33.538 s

Press any key to continue.

|