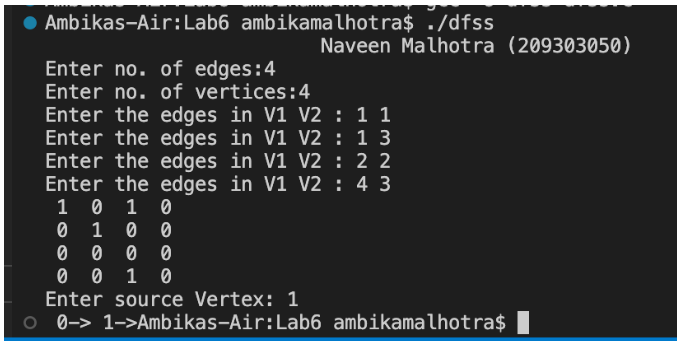
LAB 9:

1. DFS Implementation

```
#include <stdio.h>
#include <stdlib.h>
int sourceV, Vertex, Edge, time, visited[10], Graph[10][10];
void DepthFirstSearch(int i)
{
   int j;
   visited[i] = 1;
   printf(" %d->", i++);
   for (j = 0; j < Vertex; j++)
   {
      if (Graph[i][j] == 1 && visited[j] == 0)
         DepthFirstSearch(j);
   }
}
int main()
{   int i, j, vertex1, vertex2;
   printf("\t\t\tGraphs\n");
   printf("Enter no. of edges:");
   scanf("%d", &Edge);
   printf("Enter no. of vertices:");
   scanf("%d", &Vertex);
   for (i = 0; i < Vertex; i++)
   {
      for (j = 0; j < Vertex; j++)
         Graph[i][j] = 0;
   }
   for (i = 0; i < Edge; i++)
   {
      printf("Enter the edges in V1 V2 : ");
      scanf("%d%d", &vertex1, &vertex2);
      Graph[vertex1 - 1][vertex2 - 1] = 1;
   }
   for (i = 0; i < Vertex; i++)
   {
      for (j = 0; j < Vertex; j++)
         printf(" %d ", Graph[i][j]);
           printf("\n");
   }
   printf("Enter source Vertex: ");
   scanf("%d", &sourceV);
   DepthFirstSearch(sourceV - 1);
return 0; }
```

OUTPUT:

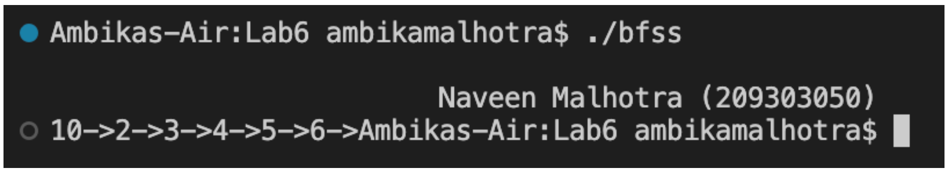


```
Ambikas-Air:Lab6 ambikamalhotra$ ./dfss
                        Naveen Malhotra (209303050)
Enter no. of edges:4
Enter no. of vertices:4
Enter the edges in V1 V2 : 1 1
Enter the edges in V1 V2 : 1 3
Enter the edges in V1 V2 : 2 2
Enter the edges in V1 V2 : 4 3
 1  0  1  0
 0  1  0  0
 0  0  0  0
 0  0  1  0
Enter source Vertex: 1
 0-> 1->Ambikas-Air:Lab6 ambikamalhotra$ 
```

2. BFS Implementation

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

struct queue
{
   int size;
   int f;
   int r;
   int *arr;
};

int isEmpty(struct queue *q)
{
   if (q->r == q->f)
   {
      return 1;
   }
   return 0;
}

int isFull(struct queue *q)
{
   if (q->r == q->size - 1)
   {
      return 1;
   }
   return 0;
}

void enqueue(struct queue *q, int val)
{
   if (isFull(q))
   {
      printf("This Queue is full\n");
   }
   else
   {
      q->r++;
      q->arr[q->r] = val;
      // printf("Enqued element: %d\n", val);
   }
}

int dequeue(struct queue *q)
{
   int a = -1;
   if (isEmpty(q))
   {
        printf("This Queue is empty\n");
    }
    else
    {
        q->f++;
        a = q->arr[q->f];
    }
    return a;
}

int main()
{
    // Initializing Queue (Array Implementation)
    printf("\n\t\t\tNaveen Malhotra (209303050)\n");
    struct queue q;
    q.size = 400;
    q.f = q.r = 0;
    q.arr = (int *)malloc(q.size * sizeof(int));

    // BFS Implementation
    int node;
    int i = 1;
    int visited[7] = {0, 0, 0, 0, 0, 0, 0};
    int a[7][7] = {
        {0, 1, 1, 1, 0, 0, 0},
        {1, 0, 1, 0, 0, 0, 0},
        {1, 1, 0, 1, 1, 0, 0},
        {1, 0, 1, 0, 1, 0, 0},
        {0, 0, 1, 1, 0, 1, 1},
        {0, 0, 0, 0, 1, 0, 0},
        {0, 0, 0, 0, 1, 0, 0}};
    printf("%d", i);
    visited[i] = 1;
    enqueue(&q, i); // Enqueue i for exploration
    while (!isEmpty(&q))
    {
        int node = dequeue(&q);
        for (int j = 0; j < 7; j++)
        {
            if (a[node][j] == 1 && visited[j] == 0)
            {
                printf("%d", j);

                printf("->");
                visited[j] = 1;
                enqueue(&q, j);
            }
        }
        }
    return 0;
}
```

OUTPUT:



```
Ambikas-Air:Lab6 ambikamalhotra$ ./bfss

                        Naveen Malhotra (209303050)
10->2->3->4->5->6->Ambikas-Air:Lab6 ambikamalhotra$
```