Ex. No. 1.A **UNINFORMED SEARCH ALGORITHM - BFS**

Date:

Aim:

To write a Python program to implement Breadth First Search (BFS).

Algorithm:

```
Step 1.  Start
Step 2.  Put any one of the graph's vertices at the back of the queue.
Step 3.  Take the front item of the queue and add it to the visited list.
Step 4.  Create a list of that vertex's adjacent nodes. Add those which are not within the
         visited list to the rear of the queue.
Step 5.  Continue steps 3 and 4 till the queue is empty.
Step 6.  Stop
```

Program:

```
graph = {
  '5' : ['3','7'],
  '3' : ['2', '4'],
  '7' : ['8'],
  '2' : [],
  '4' : ['8'],
  '8' : []
}

visited = [] # List for visited nodes.
queue = []     #Initialize a queue

def bfs(visited, graph, node): #function for BFS
  visited.append(node)
  queue.append(node)

  while queue:          # Creating loop to visit each node
    m = queue.pop(0)
    print (m, end = " ")

    for neighbour in graph[m]:
      if neighbour not in visited:
        visited.append(neighbour)
        queue.append(neighbour)

# Driver Code
print("Following is the Breadth-First Search")
bfs(visited, graph, '5')    # function calling
```

Result:

Thus the Python program to implement Breadth First Search (BFS) was developed successfully.

Ex. No.1.B **UNINFORMED SEARCH ALGORITHM - DFS**

Date:

Aim:

To write a Python program to implement Depth First Search (DFS).

Algorithm:

```
Step 1. Start
Step 2. Put any one of the graph's vertex on top of the stack.
Step 3. After that take the top item of the stack and add it to the visited list of the vertex.
Step 4. Next, create a list of that adjacent node of the vertex. Add the ones which aren't in the
        visited list of vertexes to the top of the stack.
Step 5. Repeat steps 3 and 4 until the stack is empty.
Step 6. Stop
```

Program:

```
graph = {
  '5' : ['3','7'],
  '3' : ['2', '4'],
  '7' : ['8'],
  '2' : [],
  '4' : ['8'],
  '8' : []
}

visited = set() # Set to keep track of visited nodes of graph.

def dfs(visited, graph, node):  #function for dfs
    if node not in visited:
        print (node)
        visited.add(node)
        for neighbour in graph[node]:
            dfs(visited, graph, neighbour)

# Driver Code
print("Following is the Depth-First Search")
dfs(visited, graph, '5')
```

Result:

Thus the Python program to implement Depth First Search (BFS) was developed successfully.