**[18CSC305J(Artificial Intelligence)](https://classroom.google.com/c/MjU3Mjk3ODY4MDA2)**

VIKASH MISHRA

RA1811003010344

TECH CSE 3RD YR

F1 SCETION

18CSC305J - Artificial Intelligence lab

EXPERIMENT : Breadth-First Search

ALGORITHM :

1. Add the vertex to start the breadth-first search to the empty queue.
2. Extract a vertex from the queue and its neighbors to the queue of that isn’t marked visited.
3. Repeat step 2 until the queue is empty.

CODE :

#BFS

graph = {

'0': ['1', '2'],

'1': ['0', '3', '4'],

'2': ['0', '3'],

'3': ['1', '2', '4'],

'4': ['1', '3']

}

def bfs(graph, initial):

visited = []

queue = [initial]

while queue:

node = queue.pop(0)

if node not in visited:

visited.append(node)

neighbours = graph[node]

for neighbour in neighbours:

queue.append(neighbour)

return visited

print(bfs(graph,'0'))

OUTPUT :

1

EXPERIMENT : Depth-First Search

ALGORITHM :

1.Start by putting any one of the graph's vertices on top of a stack.

2.Take the top item of the stack and add it to the visited list.

3.Create a list of that vertex's adjacent nodes. Add the ones which aren't in the visited list to the top of the stack.

4.Keep repeating steps 2 and 3 until the stack is empty.

CODE :

#dfs

def dfs(graph, start, visited=None):

if visited is None:

visited = set()

visited.add(start)

print(start)

for next in graph[start] - visited:

dfs(graph, next, visited)

return visited

graph = {'0': set(['1', '2']),

'1': set(['0', '3', '4']),

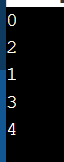
'2': set(['0']),

'3': set(['1']),

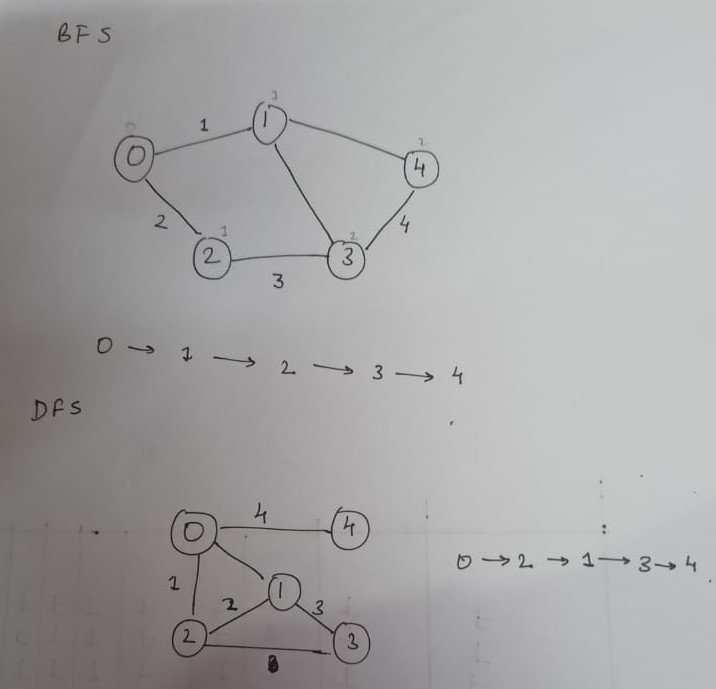
'4': set(['2', '3'])}

dfs(graph, '0')

OUTPUT :



GRAPH :



RESULT :

The given graph were traversed according to the algorithm which were implemented successfully.