# Trace the following breadth first algorithm

|  |  |
| --- | --- |
| **1**  **2**  **3**  **4**  **5**  **6**  **7**  **8**  **9**  **10**  **11**  **12**  **13**  **14** | def breadth\_first\_traversal(node)  queue = []  queue.append(node)  visited.append(node)    while queue is not empty  node = queue.pop(0)  print (node, end = " ")  for i in graph[node]:  if i not in visited  queue.append(i)  visited.append(i)  graph={'A':['D','B'],  'B':['A','E','C','F'],  'C’: ['B','F’],  'D': ['A','E'],  'E':['D','B'],  'F':['B','C’]}  visited = []  breadth\_first\_traversal("A") |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Node** | **i** | **Output** | **Visited** | **Queue** |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

# Trace the following depth first algorithm

|  |  |
| --- | --- |
| **1**  **2**  **3**  **4**  **5**  **6**  **7**  **8**  **9**  **10** | def depth\_first\_traversal(node):  visited.append(node)  for i in graph[node]:  if i not in visited:  depth\_first\_traversal(i)  return  graph={“A”:[“D”,“B”],  “B”:[“A”,”E”,”C”,”F”],  “C”: [“B”,“F”],  “D”: [“A”,”E”],  ”E”:[“D”,”B”],  ”F”:[“B”,”C”]}  visited = []  depth\_first\_traversal(“A”)  print(visited) |

|  |  |  |  |
| --- | --- | --- | --- |
| **Call** | **Node** | **i** | **Visited** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |