CODE:

```
def movegen(current, graph):
      return [[n, current] for n in graph[current]]


def goal_test(current, goal):
      return current in goal


def transversal(closed, goal_node):
      print("Transversal: ", end = "")
      for i in closed:
      print(f"{i[0]} --> ", end =" ")
      if goal_node:print(goal_node)


def bfs(graph):
      open = []
      closed = []

      start_node = input("\n\nEnter the start node: ")
      goal_node = input("Enter the goal node or nodes seperated by
spaces: ").split()

      open.append([start_node,None])

      while open:
      current = open[0][0]
      if goal_test(current, goal_node):
            print("\n\nGoal found\n")
            transversal(closed, current)
            return
      else:
            child_list = movegen(current, graph)
            for child in child_list:
                  if any(child[0] == o[0] for o in open) or
any(child[0] == c[0] for c in closed):
                  continue
                  open.append(child)

            closed.append(open[0])
            del open[0]
```

```
      if not goal_node:
      print("\n\nGoal node not specified")
      transversal(closed, goal_node)
      else:print("\n\nGoal not found")


def input_graph():
      graph = {}

      n = int(input("Enter the number of nodes: "))

      for i in range(n):
      node = input("Enter a node: ")
      neighbours = input(f"Enter the neighbours of {node} seperated
by spaces: ").split()
      graph[node] = neighbours

      return graph


if __name__ == "__main__":
      graph = input_graph()
      bfs(graph)
```

OUTPUT:

1. Initial state is the goal state

```
Enter the number of nodes: 7
Enter a node: S
Enter the neighbours of S seperated by spaces: A B C
Enter a node: A
Enter the neighbours of A seperated by spaces: S B E
Enter a node: B
Enter the neighbours of B seperated by spaces: S A D
Enter a node: C
Enter the neighbours of C seperated by spaces: S D G
Enter a node: E
Enter the neighbours of E seperated by spaces: A D G
Enter a node: D
Enter the neighbours of D seperated by spaces: B E C G
Enter a node: G
Enter the neighbours of G seperated by spaces: E D C


Enter the start node: S
Enter the goal node or nodes seperated by spaces: S


Goal found

Transversal: S
```

2. Goal state is not specified

```
Enter the number of nodes: 7
Enter a node: S
Enter the neighbours of S seperated by spaces: A B C
Enter a node: A
Enter the neighbours of A seperated by spaces: S B E
Enter a node: B
Enter the neighbours of B seperated by spaces: S A D
Enter a node: C
Enter the neighbours of C seperated by spaces: S D G
Enter a node: E
Enter the neighbours of E seperated by spaces: A D G
Enter a node: D
Enter the neighbours of D seperated by spaces: B E C G
Enter a node: G
Enter the neighbours of G seperated by spaces: C D E


Enter the start node: S
Enter the goal node or nodes seperated by spaces:


Goal node not specified
Transversal: S -->  A -->  B -->  C -->  E -->  D -->  G
```

3. 1 goal state

```
Enter the number of nodes: 7
Enter a node: S
Enter the neighbours of S seperated by spaces: A B C
Enter a node: A
Enter the neighbours of A seperated by spaces: S B E
Enter a node: B
Enter the neighbours of B seperated by spaces: S A D
Enter a node: C
Enter the neighbours of C seperated by spaces: S D G
Enter a node: E
Enter the neighbours of E seperated by spaces: A D G
Enter a node: D
Enter the neighbours of D seperated by spaces: B C E G
Enter a node: G
Enter the neighbours of G seperated by spaces: E D C


Enter the start node: S
Enter the goal node: G


Goal found

Transversal: S -->  A -->  B -->  C -->  E -->  D -->  G
```

4. More than 1 goal state

```
Enter the number of nodes: 7
Enter a node: S
Enter the neighbours of S seperated by spaces: A B C
Enter a node: A
Enter the neighbours of A seperated by spaces: S B E
Enter a node: B
Enter the neighbours of B seperated by spaces: S A D
Enter a node: C
Enter the neighbours of C seperated by spaces: S D G
Enter a node: E
Enter the neighbours of E seperated by spaces: A D G
Enter a node: D
Enter the neighbours of D seperated by spaces: B E C G
Enter a node: G
Enter the neighbours of G seperated by spaces: E D C


Enter the start node: S
Enter the goal node or nodes seperated by spaces: B G


Goal found

Transversal: S -->  A -->  B
```