```
from collections import defaultdict

class·Graph:
····def·__init__(self,vertices):
········self.V·=·vertices
········self.graph·=·defaultdict(list)
····def·addEdge(self,u,v):
········self.graph[u].append(v)
····def·DLS(self,source,target,maxDepth):
········if·source·==·target·:·return·True
········if·maxDepth·<=·0·:·return·False
········#·recursively·traversing·the·graph·while·searching
········for·i·in·self.graph[source]:
················if(self.DLS(i,·target,·maxDepth-1)):
····················return·True
········return·False
g·=·Graph(9)#·creating·the·graph
g.addEdge(0,·1)
g.addEdge(0,·2)
g.addEdge(1,·3)
g.addEdge(1,·4)
g.addEdge(2,·5)
g.addEdge(2,·6)
g.addEdge(3,7)
g.addEdge(3,8)
target·=·3
maxDepth·=·3
source·=·0

if·g.DLS(source,·target,·maxDepth)·==·True:
····print(f"Target·{target}·is·reachable·from·source·{source}·within·max·d
else:
····print(f"Target·{target}·is·NOT·reachable·from·source·{source}·within·m
```

```
Target 3 is reachable from source 0 within max depth 3
```