Name: Dhruv Bheda **SAPID:** 60004200102
DIV: B/B1

DMW
Exp9

Aim: To implement the HITS algorithm.

Theory:

Hyperlink Induced Topic Search (HITS) Algorithm is a Link Analysis Algorithm that rates webpages, developed by Jon Kleinberg. This algorithm is used for the web link structures to discover and rank the webpages relevant for a particular search. HITS uses hubs and authorities to define a recursive relationship between web pages. Before understanding the HITS Algorithm, we first need to know about Hubs and Authorities.

Given a query to a Search Engine, the set of highly relevant web pages is called Roots.
They are potential Authorities.
Pages that are not very relevant but point to pages in the Root are called Hubs. Thus, an Authority is a page that many hubs link to whereas a Hub is a page that links to many authorities.

Algorithm:

1. Let the number of iterations be k.
2. Each node is assigned a Hub score = 1 and an Authority score = 1.
3. Repeat k times:

   Hub update: Each node's Hub score = Σ (Authority score of each node it points to).

   Authority update: Each node's Authority score = Σ (Hub score of each node pointing to it).

   Normalize the scores by dividing each Hub score by the square root of the sum of the squares of all Hub scores, and dividing each Authority score by the square root of the sum of the squares of all Authority scores. (optional)

Code:

```
import networkx as nx import
matplotlib.pyplot as plt

G = nx.DiGraph()

connections = []
edges = int(input('Enter the number of edges: '))
print('Enter the connection between the nodes: ')
for i in range(edges):
   print(f'Edge {i + 1}: ')
   start = input('Enter start node: ')
end = input('Enter end node: ')
 connections.append((start, end))
print()

G.add_edges_from(connections)

plt.figure(figsize = (10, 10))

nx.draw_networkx(G, with_labels = True)

hubs, authorities = nx.hits(G, max_iter = 50, normalized = True)

print("Hub Scores: ")
for key, value in hubs.items():
  print(f'{key}: {value}')
print()

print("Authority Scores: ")
for key, value in authorities.items():
    print(f'{key}: {value}')
```

Output:

```
Enter the number of edges: 14
Enter the connection between the nodes:
Edge 1:
Enter start node: A
Enter end node: D
Edge 2:
Enter start node: B
Enter end node: C
Edge 3:
Enter start node: B
Enter end node: E
Edge 4:
Enter start node: C
Enter end node: A
Edge 5:
Enter start node: D
Enter end node: C
Edge 6:
Enter start node: E
Enter end node: D
Edge 7:
Enter start node: E
Enter end node: B
Edge 8:
Enter start node: E
Enter end node: F
Edge 9:
Enter start node: E
Enter end node: C
Edge 10:
Enter start node: F
Enter end node: C
Edge 11:
Enter start node: F
Enter end node: H
Edge 12:
Enter start node: G
Enter end node: A
Edge 13:
Enter start node: G
Enter end node: C
Edge 14:
Enter start node: H
Enter end node: A

Hub Scores:
A: 0.04642540403219995
D: 0.13366037526115382
B: 0.15763599442967322
C: 0.03738913224642654
E: 0.25881445984686646
F: 0.15763599442967322
H: 0.03738913224642654
G: 0.17104950750758036

Authority Scores:
A: 0.10864044011724344
D: 0.13489685434358
B: 0.11437974073336446
C: 0.38837280038761807
E: 0.06966521184241477
F: 0.11437974073336446
H: 0.06966521184241475
G: 0.0
```

Conclusion:

Learnt about hits algorithm in web structure mining and implemented it in python.