ARYAMAN MISHRA

19BCE1027

Aim: Implement HITS Algorithm.

Data Structures Used:2D Arrays.

Algorithm:

First, we have to explain what is authority and hub. HITS uses hubs and authorities to define a recursive relationship between webpages.

- Authority: A node is high-quality if many high-quality nodes link to it
- Hub: A node is high-quality if it links to many high-quality nodes

Algorithm Steps

- Initialize the hub and authority of each node with a value of 1
- For each iteration, update the hub and authority of every node in the graph
- The new authority is the **sum of the hub** of its parents
- The new hub is the **sum of the authority** of its children
- Normalize the new authority and hub

Function to calculate the authority and hub score of all the nodes in the

network.

Parameters:

outlinks: (n, n) int matrix where 1 represents the presence of a link and 0

represents absence of a link

Returns: hub_score: nd-array, containing the hub scores of the nodes

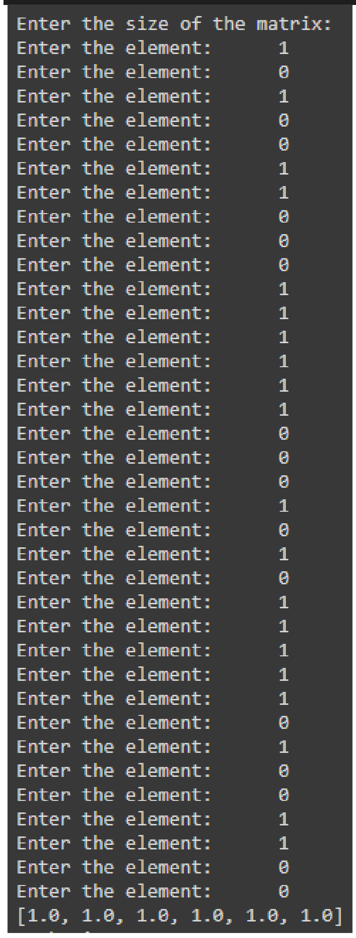
authority_score: nd-array, containing the authority scores of the nodes

1. Write a python program that takes the outlink data of a subsection of the web and that computes the Normalized Hub and authority score. The program should take the following parameters :

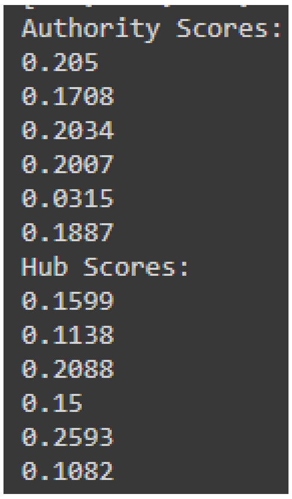
   a. Number of sites in the sub-section of the web
   b. Outlink data of the sites

IMPLEMENTATION AND RESULTS:

```python
# Implementing HITS Algorithm

import numpy as np

# Function to calculate the authority and hub score of the nodes
def authority_hub_score(outlinks):


  # size of the matrix
  size = outlinks.shape[0]

  # Initializing the lists
  hub_scores = [1.0 for i in range(size)]
  authority_scores = [1.0 for i in range(size)]

  # Printing initial Hub scores
  print(hub_scores)

  for _ in range(100):
    # Calculating the authority scores of the nodes
    for j in range(size):
      temp_auth = 0.0
      for i in range(size):
        if outlinks[i][j] == 1:
          temp_auth += hub_scores[i]
      authority_scores[j] = temp_auth

    # Normalizing the authority scores
    auth_sum = sum(authority_scores)
    # print(auth_sum)
    for i in range(len(authority_scores)):
      authority_scores[i] /= auth_sum

    # Calculating the hub scores of the nodes
    for i in range(size):
      temp_hub = 0.0
      for j in range(size):
        if outlinks[i][j] == 1:
          temp_hub += authority_scores[j]
      hub_scores[i] = temp_hub
    # Normalizing the hub scores
    hub_sum = sum(hub_scores)
    # print(hub_sum)
    for i in range(len(hub_scores)):
      hub_scores[i] /= hub_sum

  return authority_scores, hub_scores


n = int(input('Enter the size of the matrix:\t'))
outlinks = []
for i in range(n*n):
  temp = int(input('Enter the element:\t'))
  outlinks.append(temp)
outlinks = np.reshape(outlinks, (n, n))
authority_scores, hub_scores = authority_hub_score(outlinks)
print("Authority Scores:")
for i in (authority_scores):
  print(round(i, 4))
print("Hub Scores:")
for i in (hub_scores):
  print(round(i, 4))
```



```
Enter the size of the matrix:
Enter the element:      1
Enter the element:      0
Enter the element:      1
Enter the element:      0
Enter the element:      0
Enter the element:      1
Enter the element:      1
Enter the element:      0
Enter the element:      0
Enter the element:      0
Enter the element:      1
Enter the element:      1
Enter the element:      1
Enter the element:      1
Enter the element:      1
Enter the element:      1
Enter the element:      0
Enter the element:      0
Enter the element:      0
Enter the element:      1
Enter the element:      0
Enter the element:      1
Enter the element:      0
Enter the element:      1
Enter the element:      1
Enter the element:      1
Enter the element:      1
Enter the element:      1
Enter the element:      0
Enter the element:      1
Enter the element:      0
Enter the element:      0
Enter the element:      1
Enter the element:      1
Enter the element:      0
Enter the element:      0
[1.0, 1.0, 1.0, 1.0, 1.0, 1.0]
```



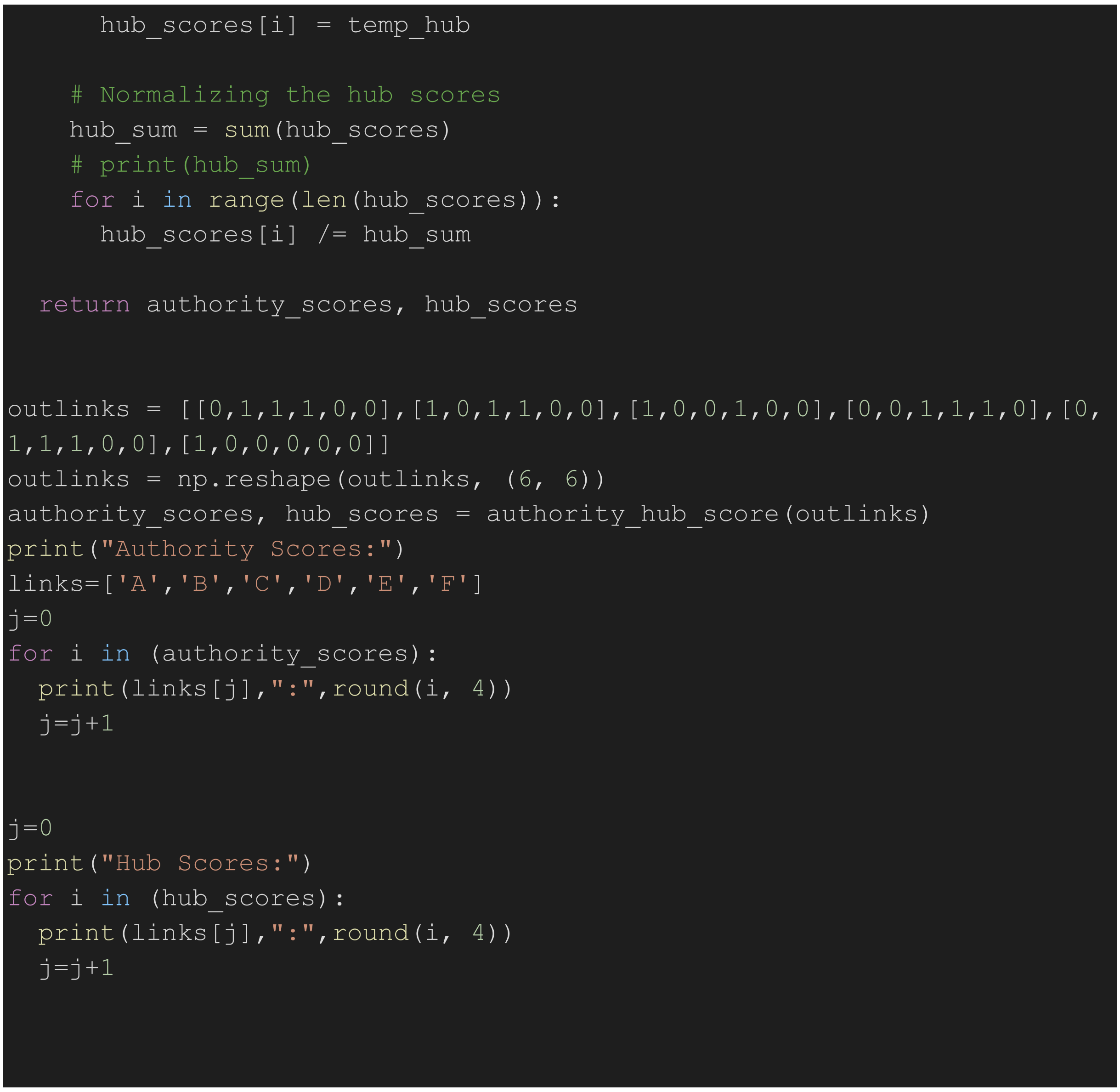
```
Authority Scores:
0.205
0.1708
0.2034
0.2007
0.0315
0.1887
Hub Scores:
0.1599
0.1138
0.2088
0.15
0.2593
0.1082
```

2. Test your program using the following linkage data. Assume each site consists of only one text page.

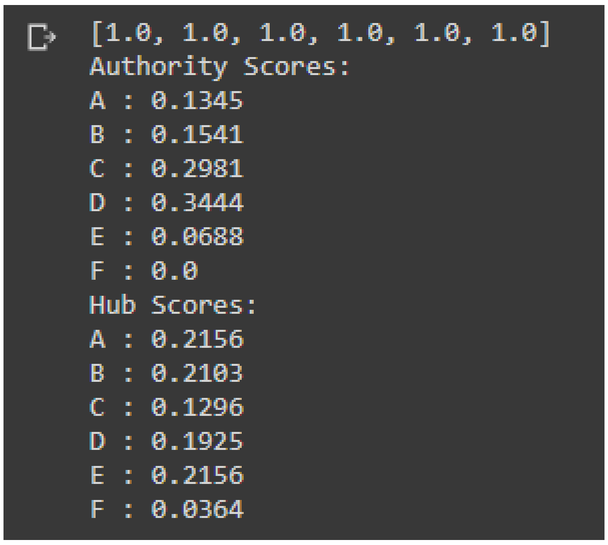
a. Site A(outlinks to B,C,D)
b. Site B(outlinks to A,C,D)
c. Site C(outlinks to A,D)
d. Site D(outlinks to C,D,E)
e. Site E(outlinks to B,C,D)
f. Site F(outlinks to A)

IMPLEMENTATION AND RESULTS:

```
# Implementing HITS Algorithm

import numpy as np

# Function to calculate the authority and hub score of the nodes
def authority_hub_score(outlinks):


  # size of the matrix
  size = outlinks.shape[0]

  # Initializing the lists
  hub_scores = [1.0 for i in range(size)]
  authority_scores = [1.0 for i in range(size)]

  # Printing initial Hub scores
  print(hub_scores)

  for _ in range(100):
    # Calculating the authority scores of the nodes
    for j in range(size):
      temp_auth = 0.0
      for i in range(size):
        if outlinks[i][j] == 1:
          temp_auth += hub_scores[i]
      authority_scores[j] = temp_auth

    # Normalizing the authority scores
    auth_sum = sum(authority_scores)
    # print(auth_sum)
    for i in range(len(authority_scores)):
      authority_scores[i] /= auth_sum

    # Calculating the hub scores of the nodes
    for i in range(size):
      temp_hub = 0.0
      for j in range(size):
        if outlinks[i][j] == 1:
          temp_hub += authority_scores[j]
```



```
      hub_scores[i] = temp_hub

    # Normalizing the hub scores
    hub_sum = sum(hub_scores)
    # print(hub_sum)
    for i in range(len(hub_scores)):
      hub_scores[i] /= hub_sum

  return authority_scores, hub_scores


outlinks = [[0,1,1,1,0,0],[1,0,1,1,0,0],[1,0,0,1,0,0],[0,0,1,1,1,0],[0,
1,1,1,0,0],[1,0,0,0,0,0]]
outlinks = np.reshape(outlinks, (6, 6))
authority_scores, hub_scores = authority_hub_score(outlinks)
print("Authority Scores:")
links=['A','B','C','D','E','F']
j=0
for i in (authority_scores):
  print(links[j],":",round(i, 4))
  j=j+1


j=0
print("Hub Scores:")
for i in (hub_scores):
  print(links[j],":",round(i, 4))
  j=j+1
```



```
[1.0, 1.0, 1.0, 1.0, 1.0, 1.0]
Authority Scores:
A : 0.1345
B : 0.1541
C : 0.2981
D : 0.3444
E : 0.0688
F : 0.0
Hub Scores:
A : 0.2156
B : 0.2103
C : 0.1296
D : 0.1925
E : 0.2156
F : 0.0364
```

CONCLUSION:HITS ALGORITHM HAS BEEN SUCCESFULLY IMPLEMENTED AND EXECUTED.