Modern Education Society's
Wadia College of Engineering, Pune-01

NAME OF STUDENT:	**CLASS:**
SEMESTER/YEAR:	**ROLL NO:**
DATE OF PERFORMANCE:	**DATE OF SUBMISSION:**
EXAMINED BY:	**EXPERIMENT NO:**

TITLE: Write a program to Implement Page Rank Algorithm.

Problem Statement:
Write a program to Compute Similarity between two text documents.

Objectives:
To Implement Page Rank Algorithm.

Outcomes:
Student can understand how to Implement Page Rank Algorithm..

Tools Required:
Hardware:
Software: Open source operating system

Theory:
Introduction
Steps to Implement PageRank Algorithm:

1. Representing the Web as a Graph: Web pages can be represented as nodes in a graph, and hyperlinks between them are the edges. If Page A links to Page B, then an edge exists from A to B.
2. Initialize PageRank: Each page is initialized with an equal rank. For a graph with N nodes, the initial PageRank of each node is:
   PR(i)=1NPR(i) = \frac{1}{N}PR(i)=N1
   where PR(i) is the PageRank of page i.
3. Iterative Calculation of PageRank: The PageRank is calculated iteratively using the following formula:
   PR(i)=1−dN+d∑j∈M(i)PR(j)L(j)PR(i) = \frac{1 - d}{N} + d \sum_{j \in M(i)} \frac{PR(j)}{L(j)}PR(i)=N1−d+dj∈M(i)∑L(j)PR(j)
   - PR(i) is the PageRank of page i.
   - d is the damping factor (usually set to 0.85).
   - M(i) is the set of pages linking to page i.
   - L(j) is the number of outbound links from page j.
   - N is the total number of pages in the graph.
4. The first term, 1−dN\frac{1 - d}{N}N1−d, accounts for the probability that a random surfer visits any page randomly (teleportation). The second term, d∑j∈M(i)PR(j)L(j)d \sum_{j \in M(i)} \frac{PR(j)}{L(j)}d∑j∈M(i)L(j)PR(j), reflects the contribution from pages linking to page i.
5. Handling Dangling Nodes: A page with no outbound links is called a "dangling node." To handle this, during each iteration, the PageRank from dangling nodes can be distributed equally to all other nodes.
6. Convergence: The algorithm iteratively recalculates the PageRank of each page until the values converge (i.e., the change between iterations becomes smaller than a predefined threshold). Typically, convergence is reached after about 20-100 iterations.

Algorithm:

1. Graph Representation:
    - The web is modeled as a directed graph, where pages are nodes and hyperlinks are directed edges.
2. Damping Factor (d):
    - A damping factor (typically set to 0.85) is used to model the probability that a user randomly clicks on links rather than directly jumping to a new page.
3. Iterative Calculation:
    - PageRank values are recalculated iteratively until they converge.
4. Convergence Criteria:
    - The difference between the PageRank values from one iteration to the next must be below a certain threshold, ensuring stability.

Time Complexity:
The time complexity of the PageRank algorithm is O(V + E) per iteration, where:

- V is the number of vertices (pages) in the graph.
- E is the number of edges (links between pages).

Applications of PageRank:

- Search Engines: PageRank was originally used by Google to rank web pages based on their relevance.
- Social Networks: It can be used to measure the importance of individuals based on their connections.
- Recommendation Systems: It helps identify influential users or content in a network.

Steps:
Note: Install modules with the help of pip
pip install sklearn

Program:

```
# import some stuff
import numpy as np
from scipy.sparse import csc_matrix

from fractions import Fraction

# keep it clean and tidy
def float_format(vector, decimal):
   return np.round((vector).astype(np.float), decimals=decimal)

G = np.matrix([[1,1,0],
            [1,0,1],
            [0,1,0]])

n=len(G)
#print(n)
# transform G into markov matrix A
M = csc_matrix(G,dtype=np.float)
rsums = np.array(M.sum(1))[:,0]
ri, ci = M.nonzero()
M.data /= rsums[ri]
```

```
# WWW matrix
# we have 3 webpages and probability of landing to each one is 1/3
#(default Probability)
#n=len(M)
dp = Fraction(1,n)

E = np.zeros((3,3))
E[:] = dp

# taxation
beta = 0.85

# WWW matrix
A = beta * M + ((1-beta) * E)

# initial vector
r = np.matrix([dp, dp, dp])
r = np.transpose(r)

previous_r = r
for it in range(1,30):
    r = A * r
    #check if converged
    if (previous_r==r).all():
        break
    previous_r = r

print ("Final:\n", float_format(r,3))
print( "sum", np.sum(r))
```

Conclusion:

The PageRank algorithm revolutionized search engine ranking by introducing a method based on the structure of the web. Despite being developed decades ago, it remains a fundamental concept in information retrieval and network analysis. Modern variations of PageRank are used in various domains, from social networks to scientific research.

Questions:

Q.1)What is the main objective of the PageRank algorithm, and why is it significant in search engines?

Q.2)Explain how the web is represented as a graph for the purpose of implementing the PageRank algorithm?

Q.3) What role does the damping factor play in the PageRank algorithm? What happens if the damping factor is set to 1 or 0?