|  |  |
| --- | --- |
| Name: | Prerna Sunil Jadhav |
| Sap  Id: | 60004220127 |
| Class: | T. Y. B.Tech (Computer Engineering) |
| Course: | Data Mining and Warehouse Laboratory |
| Course Code: | DJ19CEL501 |
| Experiment  No.: | 08 |

**AIM:** Implementation of Page Rank Algorithm

CODE:

import numpy as np

def page\_rank\_algorithm(graph,damping\_factor): outgoing = dict()

incoming\_nodes = dict() coefficients = dict()

# Outgoing Nodes

for i in range(len(graph)): outgoing[i]=0

for i,node in enumerate(graph): for edge in node:

if edge:

outgoing[i] += 1

# Incoming Nodes

for i in range(len(graph)): temp=[]

for node in graph: if node[i]:

temp.append(node) incoming\_nodes[i] = temp

# Coefficient Matrix

for i,node in enumerate(graph): temp = []

for j,other\_node in enumerate(graph):

if other\_node in incoming\_nodes[i]:

temp.append(damping\_factor\*(1.0/outgoing[j]))

elif i == j:

temp.append(-1) else:

temp.append(0) coefficients[i] = temp

coefficients\_list = []

for key,value in coefficients.items(): coefficients\_list.append(value)

constant\_matrix = []

for i in range(len(graph)):

constant\_matrix.append(damping\_factor-1)

pageranks =

np.linalg.solve(np.array(coefficients\_list),np.array(constant\_matr ix))

print()

for i,rank in enumerate(pageranks):

print('Page Rank of {} is {:.4f}'.format(chr(65+i), rank))

def main():

n = int(input('Enter the number of nodes : ')) d=float(input('Enter the damping factor : '))

graph = []

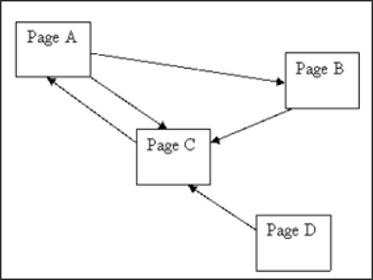
print('Enter Adjacency Matrix with terms separated by a space: ')

for i in range(n):

temp\_list = input().split(' ')

graph.append(list(map(int,temp\_list))) page\_rank\_algorithm(graph,d)

main()



GRAPH:

OUTPUT:

Enter the number of nodes : 4

Enter the damping factor : 0.85

Enter Adjacency Matrix with terms separated by a space : 0 1 1 0

0 0 1 0

1 0 0 0

0 0 1 0

Page Rank of A is 1.4901 Page Rank of B is 0.7833 Page Rank of C is 1.5766 Page Rank of D is 0.1500