Experiment 8

IDLE Shell 3.9.13

File Edit Shell Debug Options Window Help

```
    pageranks = np.linalg.solve(np.array(coefficients_list),np.array(constant_matrix))

    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()
SyntaxError: multiple statements found while compiling a single statement
>>>
================================================================================
==================
Enter the number of nodes : 4
Enter the damping factor : 0.5
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 1.2308
Page Rank of B is 0.8077
Page Rank of C is 1.4615
Page Rank of D is 0.5000
>>>
================================================================================
==================
Enter the number of nodes : 3
Enter the damping factor : 0.5
Enter Adjacency Matrix with terms separated by a space :
0 1 1
0 0 1
1 0 0

Page Rank of A is 1.0769
Page Rank of B is 0.7692
Page Rank of C is 1.1538
>>>
```

PRA.py - C:/Users/djsce.student/Desktop/PRA.py (3.9.13)

File Edit Format Run Options Window Help

```
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_matrix))

    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 : '))
```

IDLE Shell 3.9.13

File Edit Shell Debug Options Window Help

```
    pageranks = np.linalg.solve(np.array(coefficients_list),np.array(constant_matrix))

    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()
SyntaxError: multiple statements found while compiling a single statement
>>>
==========
==========
Enter the number of nodes : 4
Enter the damping factor : 0.5
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 1.2308
Page Rank of B is 0.8077
Page Rank of C is 1.4615
Page Rank of D is 0.5000
>>>
==========
==========
Enter the number of nodes : 3
Enter the damping factor : 0.5
Enter Adjacency Matrix with terms separated by a space :
0 1 1
0 0 1
1 0 0

Page Rank of A is 1.0769
Page Rank of B is 0.7692
Page Rank of C is 1.1538
>>>
```

PRA.py - C:/Users/djsce.student/Desktop/PRA.py (3.9.13)

File Edit Format Run Options Window Help

```
    # 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_matrix))

    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()
```