```
'''
PageRank Program
Based in part by: https://en.wikipedia.org/wiki/PageRank#Python
'''
import numpy as np

def pagerank(matrix, num_iterations=100, damping=0.85):
    '''
    Parameters:
        matrix: numpy array, an adjacency matrix where M of i,j represents a link (or edge)
        from j to i, s.t. for all j sum(i, M of i,j) = 1
        num_iterations: int, how many times to run pagerank, defaults to 100 rounds
        damping: float, defaults to 0.85
    Returns:
        a numpy array
        a vector of ranks s.t. v of i is the ith rank from [0,1]
        v sums to 1
    '''
    N = matrix.shape[1] # returns size of square matrix, equivalent to the number of vertices
    v = np.random.rand(N, 1) # grab a vector
    v = v / np.linalg.norm(v, 1) # L1 in the algorithm
    for iteration in range(num_iterations):
        v = damping * np.matmul(matrix, v) + (1-damping) / N
    return v


print('PageRank Program')
file_name = input('Enter a file name > ') + '.txt'
file = open(file_name, 'r')

edges = []
counter = 1
print(f'Attempting to read \'{file_name}\'')
if file.mode == 'r':
    line = file.readline() # read the first line
    elements_in_a_line = line.split() # split by the whitespace delimitter
    num_of_vertices = int(elements_in_a_line[0])  # store the num of vertices
    num_of_iterations = int(elements_in_a_line[1]) # store num of iterations
    print(f'\nThe number of vertices is {num_of_vertices}\nThe number of iterations is {num_of_iterations}')
    line = file.readline() # read the next line
    damping_factor = float(line) # store the damping factor
    print(f'The damping factor is {damping_factor}\n')
    line = file.readline() # read the next line
    while line: # and continue to do so until end of file
        elements_in_a_line = line.split() #split the line by whitespace
        edge_temp = [] # create an empty list of the values
        edge_temp.append(int(elements_in_a_line[0])) # add the 1st value
        edge_temp.append(int(elements_in_a_line[1])) # add the 2nd value
        edges.append(edge_temp) # make a new edge, add it to the list of edges
        line = file.readline() # read the next line

for edge in edges:
    print(f'vertex {edge[0]} links to the vertex {edge[1]}') # print all the vertex and their links read (i.e. all the edges)

adjacency_matrix = np.zeros((num_of_vertices, num_of_vertices)) # create a 'blank matrix' filled with all 0's size n * n
```

```
56 for edge in edges:
57     num_of_edges_connected = [x[0] for x in edges].count(edge[0]) # calculate the number of
   edges this vertex has
58     adjacency_matrix[edge[0]][edge[1]] = 1 / num_of_edges_connected # mark the connection
   in the corresponding row & col
59
60 print(f'\nGraph initialized with initial values before running pagerank:
   {adjacency_matrix}')
61 v = pagerank(adjacency_matrix, num_of_iterations, damping_factor)
62 print(f'\nConverged values after running pagerank for {num_of_iterations} iterations with
   damping = {damping_factor}:')
63 for i in range(len(v)):
64     print(f'Vertex {i} = {v[i][0]}')
65
66
67
68
```