#Level\_01

import numpy as np

from queue import Queue

def bfs(s):

color = np.empty(vertex, dtype = 'object')

color[:] = 'WHITE'

parent = np.empty(vertex, dtype = 'object')

parent[:] = np.NaN

d = np.zeros(vertex, dtype = 'int')

my\_queue = Queue(maxsize = vertex)

color[s] = 'GRAY'

parent[s] = np.NaN

d[s] = 0

my\_queue.put(s)

while not my\_queue.empty():

u = my\_queue.get()

#print(u)

for v in range(1,vertex):

if adj\_matrix[u] [v] == 1:

if color[v] == 'WHITE':

color[v] = 'GRAY'

d[v] = d[u]+1

parent[v] = u

my\_queue.put(v)

color[u] = 'BLACK'

return d[key]

#End of bfs

file = open('E:\SEMESTER\Summer- 21\CSE422\Lab\Lab-1\input\_level\_01.txt','r')

line\_1 = int(file.readline().strip())

line\_2 = int(file.readline().strip())

loop = line\_2

vertex = line\_1+1

adj\_matrix = np.zeros((vertex,vertex),dtype = 'int')

count = 0

while count<loop:

line = file.readline().strip()

a = line.split(' ')

u = int(a[0])

v = int(a[1])

adj\_matrix[u] [v] = 1

count+=1

line\_3 = int(file.readline().strip())

key = line\_3

result = bfs(0)

print(result)

#Level\_02

import numpy as np

from queue import Queue

def bfs(s):

color = np.empty(vertex, dtype = 'object')

color[:] = 'WHITE'

parent = np.empty(vertex, dtype = 'object')

parent[:] = np.NaN

d = np.zeros(vertex, dtype = 'int')

my\_queue = Queue(maxsize = vertex)

color[s] = 'GRAY'

parent[s] = np.NaN

d[s] = 0

my\_queue.put(s)

while not my\_queue.empty():

u = my\_queue.get()

#print(u)

for v in range(1, vertex):

if adj\_matrix[u] [v] == 1:

if color[v] == 'WHITE':

color[v] = 'GRAY'

d[v] = d[u]+1

parent[v] = u

my\_queue.put(v)

color[u] ='BLACK'

return d[key]

#End of bfs

file = open('E:\SEMESTER\Summer- 21\CSE422\Lab\Lab-1\input\_level\_02.txt','r')

line\_1 = int(file.readline().strip())

line\_2 = int(file.readline().strip())

loop = line\_2

vertex = line\_1+1

adj\_matrix = np.zeros((vertex,vertex), dtype = 'int')

count = 0

while count<loop:

line = file.readline().strip()

a = line.split(' ')

u = int(a[0])

v = int(a[1])

adj\_matrix[u] [v] = 1

count+=1

lina = int(file.readline().strip())

key = lina

nora = int(file.readline().strip())

lara = int(file.readline().strip())

nora = bfs(nora)

lara = bfs(lara)

if nora<lara:

print("Nora")

else:

print("Lara")

#Level\_03

import numpy as np

from queue import Queue

def bfs(s):

color = np.empty(vertex, dtype = 'object')

color[:] = 'WHITE'

parent = np.empty(vertex, dtype = 'object')

parent[:] = np.NaN

d = np.zeros(vertex, dtype = 'int')

my\_queue = Queue(maxsize = vertex)

color[s] = 'GRAY'

parent[s] = np.NaN

d[s] = 0

my\_queue.put(s)

while not my\_queue.empty():

u = my\_queue.get()

#print(u)

for v in range(1,vertex):

if adj\_matrix[u] [v] == 1:

if color[v] == 'WHITE':

color[v] = 'GRAY'

d[v] = d[u]+1

parent[v] = u

my\_queue.put(v)

color[u] = 'BLACK'

return d[key]

#End of bfs

file = open('E:\SEMESTER\Summer- 21\CSE422\Lab\Lab-1\input\_level\_03.txt','r')

line\_1 = int(file.readline().strip())

line\_2 = int(file.readline().strip())

loop = line\_2

vertex = line\_1+1

adj\_matrix = np.zeros((vertex,vertex), dtype = 'int')

count = 0

while count<loop:

line = file.readline().strip()

a = line.split(' ')

u = int(a[0])

v = int(a[1])

adj\_matrix[u] [v] = 1

count+=1

lina = int(file.readline().strip())

key = lina

n = int(file.readline().strip())

a = []

count = 0

while count<n:

i = int(file.readline().strip())

j = bfs(i)

a.append(j)

count+=1

result = np.min(a)

print(result)