Task1:

def bfs(graph, start, goal):

queue = [[start]]

visited = []

while queue:

path = queue.pop(0)

node = path[len(path)-1]

if node not in visited:

x = graph[node]

for y in x:

n\_path = list(path)

n\_path.append(y)

queue.append(n\_path)

if y == goal:

move = len(n\_path)-1

return move

pos = int(input('number of different fixed positions: '))

num = int(input('number of connection: '))

lines = []

for n in range(num):

lines.append(input().strip().split(' '))

lina = input('Position of Lina: ')

graph = {}

for z in lines:

x, y = z[0], z[1]

if x in graph.keys():

graph[x].append(y)

else:

graph[x] = [y]

if y in graph.keys():

graph[y].append(x)

else:

graph[y] = [x]

min\_num = bfs(graph, "0", lina)

print('Minimum number of moves Nora needs is: ' + str(min\_num) )

Task2:

def bfs(graph, start, goal):

queue = [[start]]

visited = []

while queue:

path = queue.pop(0)

node = path[len(path)-1]

if node not in visited:

x = graph[node]

for y in x:

n\_path = list(path)

n\_path.append(y)

queue.append(n\_path)

if y == goal:

move = len(n\_path)-1

return move

pos = int(input('number of different fixed positions: '))

num = int(input('number of connection: '))

lines = []

for n in range(num):

lines.append(input().strip().split(' '))

lina = input('Position of Lina: ')

nora = input('Position of Nora: ')

lara = input('Position of Lara: ')

graph = {}

for z in lines:

x, y = z[0], z[1]

if x in graph.keys():

graph[x].append(y)

else:

graph[x] = [y]

if y in graph.keys():

graph[y].append(x)

else:

graph[y] = [x]

nora\_min = bfs(graph, nora, lina)

lara\_min = bfs(graph, lara, lina)

if nora\_min > lara\_min:

print("Lara")

else:

print("Nora")

Task3:

def bfs(graph, start, goal):

queue = [[start]]

visited = []

while queue:

path = queue.pop(0)

node = path[len(path)-1]

if node not in visited:

x = graph[node]

for y in x:

n\_path = list(path)

n\_path.append(y)

queue.append(n\_path)

if y == goal:

move = len(n\_path)-1

return move

pos = int(input('number of different fixed positions: '))

num = int(input('number of connection: '))

lines = []

for n in range(num):

lines.append(input().strip().split(' '))

lina = input('Position of Lina: ')

graph = {}

for z in lines:

x, y = z[0], z[1]

if x in graph.keys():

graph[x].append(y)

else:

graph[x] = [y]

if y in graph.keys():

graph[y].append(x)

else:

graph[y] = [x]

participent = int(input('Number of participants: '))

participent\_list = []

for i in range(participent):

i\_pos = input('position of '+ str(i+1) +' participants:')

participent\_list.append(bfs(graph, i\_pos, lina))

print('Minimum number of moves the winner needed to go :',min(participent\_list))