# Uniform Cost Search Sample Code

import queue as Q

def search(graph, start, end):

if start not in graph:

raise TypeError(str(start) + ' not found in graph !')

return

if end not in graph:

raise TypeError(str(end) + ' not found in graph !')

return

queue = Q.PriorityQueue()

queue.put((0, [start]))

while not queue.empty():

node = queue.get()

current = node[1][len(node[1]) - 1]

if end in node[1]:

print("Path found: " + str(node[1]) + ", Cost = " + str(node[0]))

break

cost = node[0]

for neighbor in graph[current]:

temp = node[1][:]

temp.append(neighbor)

queue.put((cost + graph[current][neighbor], temp))

def readGraph():

lines = int( input() )

graph = {}

for line in range(lines):

line = input()

tokens = line.split()

node = tokens[0]

graph[node] = {}

for i in range(1, len(tokens) - 1, 2):

# print(node, tokens[i], tokens[i + 1])

# graph.addEdge(node, tokens[i], int(tokens[i + 1]))

graph[node][tokens[i]] = int(tokens[i + 1])

return graph

def main():

graph = readGraph()

search(graph, 'Arad', 'Bucharest')

if \_\_name\_\_ == "\_\_main\_\_":

main()

"""

Sample Map Input:

14

Arad Zerind 75 Timisoara 118 Sibiu 140

Zerind Oradea 71 Arad 75

Timisoara Arad 118 Lugoj 111

Sibiu Arad 140 Oradea 151 Fagaras 99 RimnicuVilcea 80

Oradea Zerind 71 Sibiu 151

Lugoj Timisoara 111 Mehadia 70

RimnicuVilcea Sibiu 80 Pitesti 97 Craiova 146

Mehadia Lugoj 70 Dobreta 75

Craiova Dobreta 120 RimnicuVilcea 146 Pitesti 138

Pitesti RimnicuVilcea 97 Craiova 138 Bucharest 101

Fagaras Sibiu 99 Bucharest 211

Dobreta Mehadia 75 Craiova 120

Bucharest Fagaras 211 Pitesti 101 Giurgiu 90

Giurgiu Bucharest 90

"""