# Cost to find the AND and OR path

def Cost(H, condition, weight = 1):

cost = {}

if 'AND' in condition:

AND\_nodes = condition['AND']

Path\_A = ' AND '.join(AND\_nodes)

PathA = sum(H[node]+weight for node in AND\_nodes)

cost[Path\_A] = PathA

if 'OR' in condition:

OR\_nodes = condition['OR']

Path\_B =' OR '.join(OR\_nodes)

PathB = min(H[node]+weight for node in OR\_nodes)

cost[Path\_B] = PathB

return cost

# Update the cost

def update\_cost(H, Conditions, weight=1):

Main\_nodes = list(Conditions.keys())

Main\_nodes.reverse()

least\_cost= {}

for key in Main\_nodes:

condition = Conditions[key]

print(key,':', Conditions[key],'>>>', Cost(H, condition, weight))

c = Cost(H, condition, weight)

H[key] = min(c.values())

least\_cost[key] = Cost(H, condition, weight)

return least\_cost

# Print the shortest path

def shortest\_path(Start,Updated\_cost, H):

Path = Start

if Start in Updated\_cost.keys():

Min\_cost = min(Updated\_cost[Start].values())

key = list(Updated\_cost[Start].keys())

values = list(Updated\_cost[Start].values())

Index = values.index(Min\_cost)

# FIND MINIMIMUM PATH KEY

Next = key[Index].split()

# ADD TO PATH FOR OR PATH

if len(Next) == 1:

Start =Next[0]

Path += '<--' +shortest\_path(Start, Updated\_cost, H)

# ADD TO PATH FOR AND PATH

else:

Path +='<--('+key[Index]+') '

Start = Next[0]

Path += '[' +shortest\_path(Start, Updated\_cost, H) + ' + '

Start = Next[-1]

Path += shortest\_path(Start, Updated\_cost, H) + ']'

return Path

H = {'A': -1, 'B': 5, 'C': 2, 'D': 4, 'E': 7, 'F': 9, 'G': 3, 'H': 0, 'I':0, 'J':0}

Conditions = {

'A': {'OR': ['B'], 'AND': ['C', 'D']},

'B': {'OR': ['E', 'F']},

'C': {'OR': ['G'], 'AND': ['H', 'I']},

'D': {'OR': ['J']}

}

# weight

weight = 1

# Updated cost

print('Updated Cost :')

Updated\_cost = update\_cost(H, Conditions, weight=1)

print('\*'\*75)

print('Shortest Path :\n',shortest\_path('A', Updated\_cost,H))