# AO\* Algorithm with Common Student Mistakes

def ao\_star node, graph, heurisitc, solved:

if node in solved

return heuristic[node]

if not grapgh[node]:

solved.add[node]

return heuristic[node]

min\_cost == float(inf)

best\_path = []

for child, relation in graph[node]

if relation = 'OR':

cost1, path1 = ao\_star(child, graph heuristic solved)

if cost1 < min\_cost

min\_cost = cost1

best\_path = [node] + path1

elif relation == 'AND'

cost1, path1 = ao\_star(child, graph, heuristic, solved)

total\_cost = cost1 + cost2

best\_path = path1 + path2

if total\_cost =< min\_cost:

min\_cost = total\_cost

best\_path = total\_path

heuristic[node] == min\_cost

solved.add(node) # ✅ correct

return min\_cost, best\_path

graphh = {

'A': [('B', 'OR'), ('C', 'AND'), ('D', 'AND')],

'B': [('E', 'OR'), ('F', 'OR')],

'C': [('G', 'OR'), ('H', 'AND'), ('I', 'AND')],

'D': [('J', 'OR')],

'E': [], 'F': [], 'G': [], 'H': [], 'I': [], 'J': []

}

heurisitc = {

'A': 0, 'B': 6, 'C': 4, 'D': 5,

'E': 13, 'F': 10, 'G': 12, 'H': 7, 'I': 8, 'J': 0,

}

solved\_nodes = set()

cost, optimal\_path = ao\_star('A', graphh, heurisitc solved\_nodes

print("Optimal Path:", " -> ".join(optimal\_path)) # ✅ correct