EXPERIMENT 5

Python program for uniformed cost search

import heapq

def uniform\_cost\_search(graph, start, goal):

queue = [(0, start)] # (cost, node)

visited = set()

while queue:

cost, current = heapq.heappop(queue)

if current in visited:

continue

visited.add(current)

if current == goal:

return cost

for neighbor, edge\_cost in graph[current]:

if neighbor not in visited:

heapq.heappush(queue, (cost + edge\_cost, neighbor))

return None

# Example usage:

graph = {

'A': [('B', 2), ('C', 3)],

'B': [('D', 4)],

'C': [('D', 5)],

'D': [('E', 6)],

'E': []

}

print(uniform\_cost\_search(graph, 'A', 'E'))

OUT PUT

