

**Instituto Politécnico Nacional**

**ESCOM**

**Actividad**

**Inteligencia Artificial**

**6CV2**

**Cabrera García Daniel**

**from collections import deque**

**def cruzar\_puente(personas):**

**# Crear la cola de estados**

**cola = deque([(personas, [], 17, True)])**

**while cola:**

**# Obtener el siguiente estado de la cola**

**orilla\_inicial, orilla\_opuesta, tiempo\_restante, antorcha\_inicial = cola.popleft()**

**# Verificar si es un estado final**

**if not orilla\_inicial:**

**return (orilla\_opuesta, tiempo\_restante)**

**# Generar estados sucesores**

**for cruzar\_inicial in cruzar\_desde(orilla\_inicial, antorcha\_inicial):**

**nuevo\_estado = cruzar(cruzar\_inicial, orilla\_inicial, orilla\_opuesta, tiempo\_restante, antorcha\_inicial)**

**cola.append(nuevo\_estado)**

**for cruzar\_opuesta in cruzar\_desde(orilla\_opuesta, not antorcha\_inicial):**

**nuevo\_estado = cruzar(cruzar\_opuesta, orilla\_opuesta, orilla\_inicial, tiempo\_restante, not antorcha\_inicial)**

**cola.append(nuevo\_estado)**

**return None**

**def cruzar\_desde(orilla, antorcha):**

**cruces = []**

**for i in range(len(orilla)):**

**persona1 = orilla[i]**

**cruces.append([persona1])**

**for j in range(i + 1, len(orilla)):**

**persona2 = orilla[j]**

**if antorcha:**

**cruces.append([persona1, persona2])**

**return cruces**

**def cruzar(cruzar, orilla\_inicial, orilla\_opuesta, tiempo\_restante, antorcha\_inicial):**

**nueva\_orilla\_inicial = [p for p in orilla\_inicial if p not in cruzar]**

**nueva\_orilla\_opuesta = orilla\_opuesta + cruzar**

**tiempo\_cruce = max([tiempo(p) for p in cruzar])**

**nuevo\_tiempo\_restante = tiempo\_restante - tiempo\_cruce**

**nueva\_antorcha\_inicial = not antorcha\_inicial**

**return (nueva\_orilla\_inicial, nueva\_orilla\_opuesta, nuevo\_tiempo\_restante, nueva\_antorcha\_inicial)**

**def tiempo(persona):**

**return {'A': 1, 'B': 2, 'C': 5, 'D': 10}[persona]**

**# Ejemplo de uso**

**personas = ['A', 'B', 'C', 'D']**

**solucion = cruzar\_puente(personas)**

**if solucion:**

**orilla\_opuesta, tiempo\_restante = solucion**

**print(f"Solución: {orilla\_opuesta}, tiempo restante: {tiempo\_restante} minutos")**

**else:**

**print("No se encontró una solución")**