1. Encontrar una representación adecuada del espacio para los siguientes problemas:

   a) El juego de los 8 números (Taken):

   Dada una configuración de ocho números en un tablero de 3×3, llevarlo mediante el desplazamiento de números, a la siguiente situación final

1	2	3
8		4
7	6	5

   b) La torre de Hanói:

   Hay 64 discos de diámetro decreciente en un poste y hay que pasarlos a otro poste, utilizando un tercero para los pasos intermedios. Sólo puede moverse un disco a la vez, siempre deben estar en algún poste y no se puede colocar un disco sobre otro de menor tamaño.

   c) Cuadrado Latino:

   Consiste en llenar un cuadrado de 3×3 con un elemento del conjunto $\{1, 2, 3\}$ de forma tal que en cada fila y columna no haya elementos repetidos.

Soluciones

a)

- $S = \{M \in \mathcal{M}_{3\times 3}([\![0, 8]\!]) : |\{m_{ij}/i, j \in [\![1, 3]\!]\}| = 9\}$.
- Mover arriba: intercambia el 0 con el valor de arriba (siempre que el 0 no este en la primer fila).
- Mover abajo: intercambia el 0 con el valor de abajo (siempre que el 0 no este en la ultima fila).
- Mover izquierda: intercambia el 0 con el valor de la izquierda (siempre que el 0 no este en la primer columna).
- Mover derecha: intercambia el 0 con el valor de la derecha (siempre que el 0 no este en la última columna).

b)

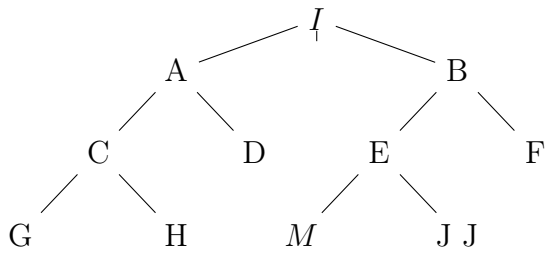
- $S = \left\{(A, B, C) : A, B, C \in [\![0, 64]\!]^{64} \wedge P(A) \wedge P(B) \wedge P(C) \wedge Q(A, B, C)\right\}$, donde:
  - $P(X) = \forall x_i \in X : x_i \leq x_{i+1}$
  - $Q(A, B, C) =$ «Los números en A, B y C no se repiten, a excepción del 0».
- Mover de A a B: intercambia el menor valor de A con el último 0 de B (siempre que el menor valor de A sea menor que el menor valor de B.
- Mover de A a C: Análogo.
- Mover de B a A: Análogo.
- Mover de B a C: Análogo.
- Mover de C a A: Análogo.
- Mover de C a B: Análogo.

c)

- $S = \{M \in \mathcal{M}_{3\times 3}([\![0, 3]\!]) : P(M) \wedge P(M^t)\}$, donde $P(M) = \forall i \in [\![1, 3]\!] (\forall j \in [\![1, 3]\!] : m_{ij} = 0 \vee m_{ij} \neq m_{ij+1})$.
- Llenar la celda m_{ij} con n, siempr y cuando la matriz resultante pertenezca al espacio de estados.

2. Escribir en pseudocódigo el algoritmo para la búsqueda a lo ancho y en profundidad, a partir del algoritmo de búsqueda general y realizando las especificaciones necesarias para cada caso.

   Dar la evolución de la Lista de Espera de Nodos, al aplicar las estrategias de búsqueda a lo ancho y en profundidad, al problema representado en el siguiente árbol, donde I es el estado inicial y M es el estado meta.



Solución

```
def dfs(problem):
    l = [(problem.startState, [], 0)]

    while l:
        node, actions, cost = l.pop()

        if problem.isGoalState(node):
            return node, actions, cost

        for succ, action, nextCost in problem.getSuccessors(node):
            l += [(succ, actions + [action], cost + nextCost)]
```

- `l ~ ["I"].`
- `l ~ ["A", "B"].`
- `l ~ ["C", "D", "B"].`
- `l ~ ["G", "H", "D", "B"].`
- `l ~ ["H", "D", "B"].`
- `l ~ ["D", "B"].`
- `l ~ ["B"].`
- `l ~ ["E", "F"].`
- `l ~ ["M", "J J", "F"].`

3. Representar el problema de los Misioneros y Caníbales descripto a continuación, mediante un espacio de estados y aplicar búsqueda primero a lo ancho para resolver el problema. Encontrar la solución de menos pasos.

   Tres misioneros y tres caníbales se encuentran en una orilla. Junto a una canoa en la que pueden cruzar 1 o 2 personas. Hay que encontrar la forma de pasarlos a todos a la otra orilla pero teniendo en cuenta que en ningún momento el número de misioneros sea menor que el de los caníbales.

Solución

- $S = \{(m, c, b, M, C) : m, c, M, C \in [\![0, 3]\!] \wedge b \in \mathbb{B} \in \wedge P\,(m, c, M, C)\}$ donde $P\,(m, c, M, C) = (m = 0 \vee m \geq c) \wedge (M = 0 \vee M \geq C) \wedge m + M = 3 \wedge c + C = 3$.

- $f_i\,(m, c, \top, M, C) = (m - i, c, \bot, M + i, C)$ para $i > 0$ siempre que el resultado pertenezca al espacio de estados.
- $f_i\,(m, c, \bot, M, C) = (m + i, c, \top, M - i, C)$ para $i > 0$ siempre que el resultado pertenezca al espacio de estados.
- $g_i\,(m, c, \top, M, C) = (m, c - i, \bot, M, C + i)$ para $i > 0$ siempre que el resultado pertenezca al espacio de estados.
- $g_i\,(m, c, \bot, M, C) = (m, c + i, \top, M, C - i)$ para $i > 0$ siempre que el resultado pertenezca al espacio de estados.

4. En el juego de los 8 números (Taken) con la siguiente configuración inicial:

2	8	3
1	6	4
7		5

   a) Resolverlo mediante la estrategia de búsqueda en profundidad estableciendo algún límite apropiado de profundidad y controlando los nodos de estados repetidos.

   b) Desarrollar las etapas de búsqueda considerando el método A* considerando:
   - h_1 = Cantidad de números fuera de lugar.
   - h_2 = Distancia en cuadras (de Manhattan).

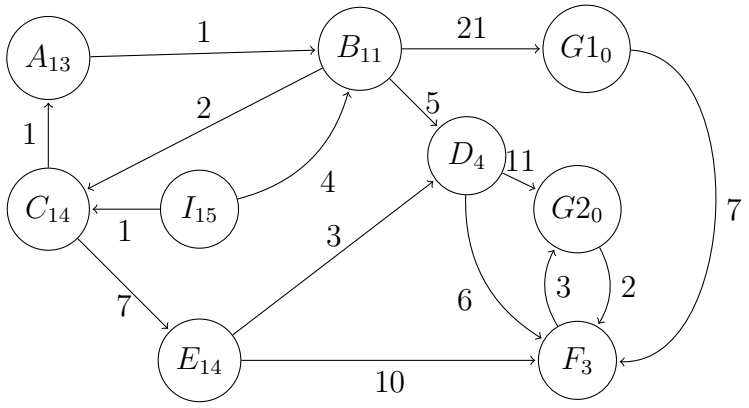
   Comparar los resultados obtenidos con ambas heurísticas (solución óptima, cantidad de nodos expandidos) en este caso de resolución y analizar si este comportamiento es un caso particular o si se puede generalizar a otros casos del juego.

5. Considerando el siguiente mapa de Rumania con las rutas existentes y las distancias en línea recta entre las distintas ciudades y Bucharest (Ejemplo extraído de Russell & Norvig).

   a) Aplique el algoritmo de búsqueda de costo uniforme para encontrar una solución al problema de ir desde Arad a Bucharest.

   b) Explorar el árbol de búsqueda con el método A* usando como heurística la distancia en línea recta, indicar en cada nodo el número de expansión y el valor de g y h.

6. Considérese el grafo dirigido de la figura, que representa un espacio de estados, siendo I el estado inicial y $G1$ y $G2$ los dos estados objetivos. El número que figura en cada estado corresponde al valor de una función heurística h' que estima el coste mínimo necesario para pasar de ese estado al objetivo más cercano. Cada arista está etiquetada con un número que representa el coste real de atravesar dicha arista.



   a) ¿Es h' una heurística admisible? Justifique.

   b) Indicar qué estado objetivo se alcanzará (si es que se alcanza alguno), qué estados se expandirán y en qué orden para cada uno de los siguientes algoritmos de búsqueda: primero en profundidad, A* y escalada simple (hill climbing). Cuando dos nodos tengan las mismas características por el criterio de selección que se esté usando, se seleccionará el primero por orden alfabético. Además, se evitarán las repeticiones de estados.

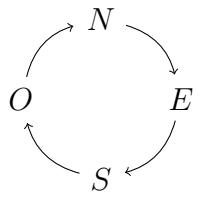
7. Un robot se mueve en una grilla de $n \times n$. En un determinado momento, el robot se encontrará en un celda (determinada por su posición x indicando la fila y la posición y indicando la columna), con una orientación la cual puede tener los siguientes valores: N (Norte), S (Sur), E (Este) y O (Oeste).

   Los movimientos que puede realizar el robot en esta grilla son:

   - Avanza: el robot se dirige a la celda contigua (costo 2). Esta celda depende de la orientación del robot. La tabla siguiente resume la ubicación del robot después de ejecutar esta acción:

Orientación	Próxima ubicación
N	$x-1, y$
S	$x+1, y$
E	$x, y+1$
O	$x, y-1$

- Gira: el robot gira 90^{o} en el sentido de las agujas del reloj. El siguiente gráfico muestra la rotación del robot (costo 1)



El robot tendrá una posición inicial en la grilla y se le solicitará que llegue a alguna celda destino. Además se especificarán un grupo de celdas prohibidas. Por lo tanto, el robot tendrá que resolver como problema determinar el camino a seguir en la grilla para alcanzar el destino final, sin pasar por las celdas prohibidas.

Se pide:

a) Representar el problema de desplazamiento del robot mediante un espacio de estados.

b) Definir una función heurística y aplicar búsqueda A* para hallar el camino de menor costo entre (1,1) si el robot comienza con orientación N y (3,4) cualquier orientación, en una grilla de (5x5), teniendo como celdas prohibidas: (1,2), (1,4) y (2,2).

8. Dado el siguiente problema, consistente en partir de una configuración $[0X0X0X0X]$ pasar a otra $[0000XXXX]$, moviendo de a dos letras adyacentes a la vez (cualquier par) desplazándolas juntas y eliminando el hueco dejado.

   a) Representar el problema mediante espacio de estados.

   b) Proponer una heurística e implementar un método de búsqueda de modo de obtener la configuración deseada con el menor número de movimientos.

9. Supóngase que usted está diseñando un compilador para una máquina con un solo registro y las siguientes instrucciones:

UNO	Poner en el registro el valor 1
DOBLE	Duplicar el contenido del registro
SUMAR_UNO	Sumar uno al contenido del registro
RESTAR_UNO	Restar uno al contenido del registro

Estas instrucciones sólo pueden ejecutarse cuando el contenido del registro no es divisible por 3. Cuando esto ocurre, la única instrucción disponible es:

DIVIDIR	Dividir el valor del registro por 3

El problema es cómo poner el número 7 en el registro. Puede suponerse que el registro está inicializado en 1.

a) Formular este problema en términos de espacios de estado.

b) Supóngase que los costos de los operadores sobre un registro que contiene el número n son:

UNO	1
DOBLE	N
SUMAR_UNO	1
RESTAR_UNO	1
DIVIDIR	2N/3

Esto es, cada operador tiene un costo equivalente a la diferencia que produce respecto del valor inicial del registro, y la función heurística $h(n) = |7 - n|$. Dibuje el árbol de búsqueda utilizando A* y determine la solución encontrada.

10. Considere un problema de búsqueda en un universo cuyos estados sean: A, B, C, D, E. Las acciones posibles, en este universo, son dadas por la siguiente tabla:

Inicial	Final	Costo
A	B	3
A	C	2
B	C	5
B	D	3
C	D	5
C	B	1
C	E	11
D	E	5
E	B	8

$nodo$	$h_1\,(nodo)$	$nodo$	$h_2\,(nodo)$
A	8	A	7
B	6	B	8
C	6	C	5
D	4	D	5
E	0	E	0

a) ¿Es A* optimo con la heurística h_1? ¿Es A* óptima con h_2?

b) ¿Son las siguientes heurísticas admisibles?

1) $h_1\,(n)$
2) $h_2\,(n)$
3) min(h1(n) , h2(n))
4) h1 (n)+ h2(n)
5) (h1 (n)+ h2(n))/2
6) max(h1(n) , h2(n))
7) h1 (n) $\times$ h2(n)
8) (h1 (n) $\times$ h2(n)) 1/2

11. Consideremos un puzzle consistente en una fila con 4 fichas con la siguiente configuración inicial:

N	N	B	B	V

Hay dos fichas negras (N), dos blancas (B) y una casilla vacía (V). El puzzle permite los siguientes movimientos:

- Una ficha puede moverse a una casilla vacía adjunta con costo unidad.
- Una ficha puede saltar sobre otras dos como máximo hasta la celda vacía, con costo igual al número de fichas saltadas. El objetivo es llegar a tener todas las fichas blancas a la izquierda de todas las negras (sin importar la posición de la casilla vacía).

a) Especificar el problema como búsqueda en un espacio de estados.

b) Especificar una función heurística h para este problema, analizar si es admisible.

c) Encontrar una solución utilizando el algoritmo A* con esa función heurística. Mostrar el árbol de búsqueda desarrollado.