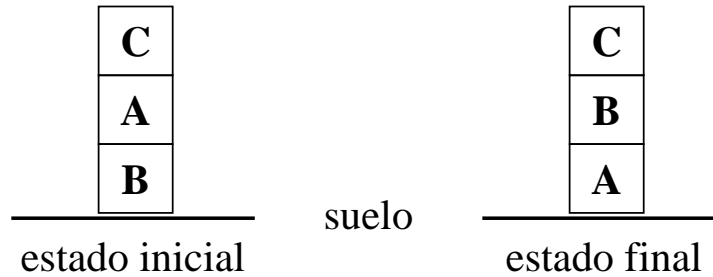


Ejercicio 1 [2.5 puntos]

Consideraremos el siguiente problema de espacio de estados:

Disponemos de tres bloques etiquetados con las letras **A**, **B** y **C** situados inicialmente como se indica en la figura y queremos moverlos hasta llegar al estado final mostrado:



Un movimiento consiste en tomar un bloque libre (es decir, que no tiene nada encima) y situarlo encima del “suelo” o encima de otro bloque libre.

Se pide:

1. Definir una heurística que permita encontrar una solución a este problema por el procedimiento de búsqueda en escalada.
2. Representar el árbol de búsqueda en escalada con dicha heurística, enumerando los nodos según se van analizando.

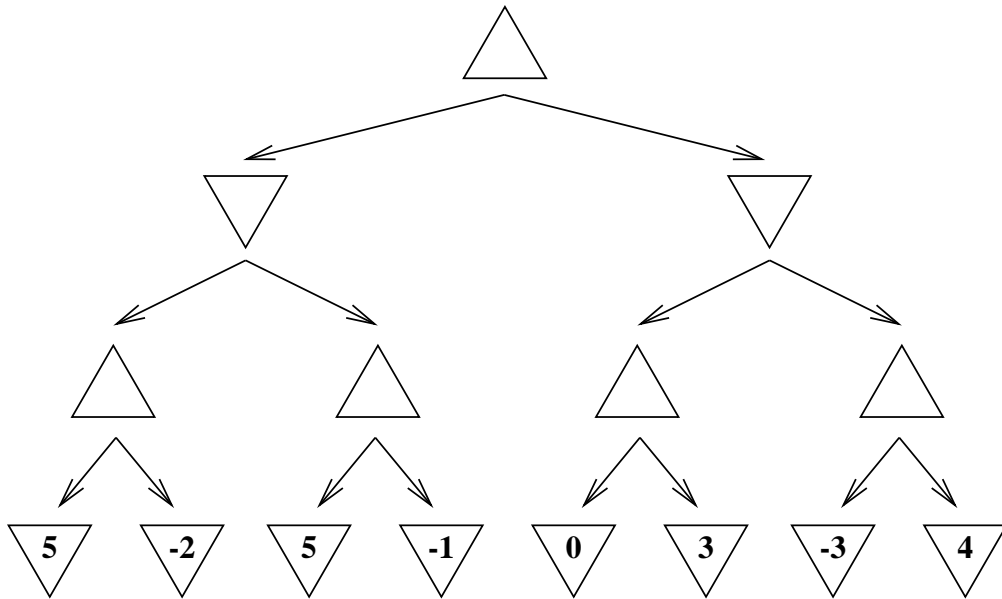
Ejercicio 2 [2.5 puntos]

En los tres primeros apartados de este ejercicio consideraremos un juego con N operadores, que es analizado utilizando la estrategia alfa-beta hasta una profundidad de 3 (el nodo raíz está a profundidad 0), partiendo de un nodo MAX. En estas circunstancias contestar a las siguientes preguntas:

- ¿Cuál es la cantidad mínima de nodos analizados, incluyendo las hojas?
- ¿En qué condiciones (valores de las cotas iniciales, valores de la función de evaluación estática en las hojas, ...) se corta la mayor cantidad del árbol?
- ¿En qué condiciones (valores de las cotas iniciales, valores de la función de evaluación estática en las hojas, ...) se puede cortar el primer hijo de un nodo MAX?

Apellidos:
 Nombre:

- Aplicar la estrategia alfa-beta con cotas iniciales $-\infty$ (α) y $+\infty$ (β) al siguiente árbol:



Ejercicio 3 [2.5 puntos]

Se considera el siguiente programa:

```
p(0,X,X).
p(s(X),Y,s(Z)) :- p(X,Y,Z).
```

```
q(0,X).
q(s(X),s(Y)) :- q(X,Y).
```

Construir el árbol de resolución SLD correspondiente a dicho programa y a la pregunta

?- p(X,Y,s(s(0))), q(X,Y).

indicando las respuestas obtenidas.

Ejercicio 4 [2.5 puntos]

Se considera el procedimiento general de búsqueda en Prolog definido por

```
busqueda(M,S) :-
    estado_inicial(E),
    busqueda(M,[[E]],S).
```

```
busqueda(_M,Abiertos,S) :-
    Abiertos = [[E|C]|_],
```

```
estado_final(E),
reverse([E|C],S).
busqueda(M,Abiertos,S) :-
    selecciona(M,Abiertos,N,R),
    sucesores(N,Sucesores),
    expande(M,R,Sucesores,NAbiertos),
    busqueda(M,NAbiertos,S).
```

Se pide:

1. Definir la relación `sucesores(+N,?L)` que se verifique si L es la lista de los sucesores del nodo N.
 2. Definir la relación `expande(+M,+L1,+Sucesores,?L2)` en el caso de la búsqueda en profundidad; es decir que se verifique si M es `profundidad` y L2 es la lista obtenida expandiendo (según el método M) la lista de nodos L1 con la lista de nodos `Sucesores`.
 3. Definir la relación `selecciona(+M,+LN1,?N,?LN2)` en el caso de la búsqueda `optimal`; es decir, que se verifique si M es `optimal`, N es el nodo de la lista LN1 seleccionado en el caso de la búsqueda `optimal` y LN2 es la lista de los restantes nodos.
-