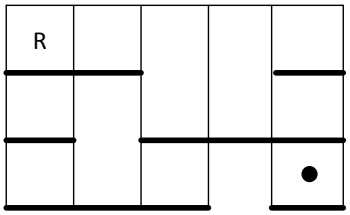
Sistemas Inteligentes – Examen Bloque 1, 13 noviembre 2020
Test A (1,75 puntos) puntuación: max (0, (aciertos – errores/3)*1,75/9)

Apellidos:	**Nombre**:
Grupo: A B C D E F G 4IA	

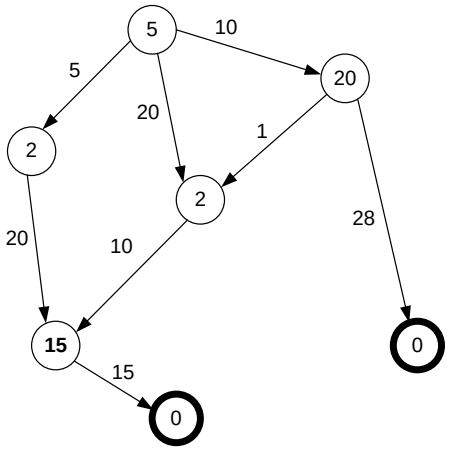
1) La figura muestra un grid en el que hay un robot en la posición (x,y)=(1,3) y quiere llegar a la casilla marcada con un punto en posición (x,y)=(5,1). Algunas casillas del grid no tienen suelo, es decir, hay huecos, y el robot nunca puede posicionarse en ellas. El robot puede realizar movimientos horizontales (derecha,izquierda), verticales (arriba,abajo) y diagonales (por ejemplo, desde la casilla (1,3) el robot solo puede moverse a la derecha y abajo). El coste de cada movimiento es 1. Indica la respuesta **CORRECTA**:



A. La solución óptima para este problema es 6.
B. La heurística 'Distancias de Manhattan' aplicada a este problema es una heurística admisible.
C. La expansión del nodo (x,y)=(5,2) genera 4 nodos hijo.
D. Ninguna de las respuestas anteriores es correcta.

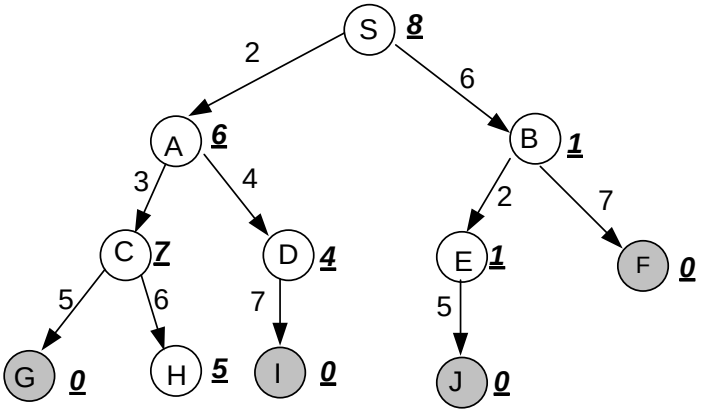
---

2) Dado el siguiente espacio de búsqueda, donde se indica el coste en las ramas, la estimación h(n) dentro de cada nodo, y el nodo inicial es el marcado como h(n)=5, indica la respuesta **CORRECTA**.



A. h(n) no es admisible.
B. Un Algoritmo Tree-Search, de tipo A, con control de repetidos en lista OPEN no obtiene la solución óptima.
C. Un Algoritmo Graph-Search, de tipo A, con control de repetidos en OPEN y sin re-expansión de nodos repetidos en CLOSED no obtiene la solución óptima
D. Una búsqueda en anchura obtiene la solución óptima.

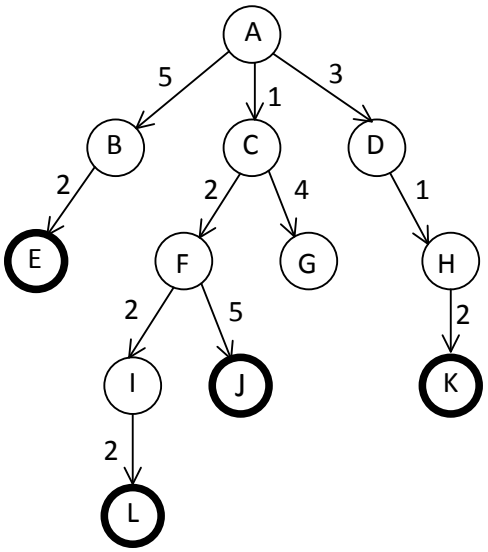
3) Dado el siguiente espacio de búsqueda, donde se indica el coste en las ramas y la estimación h(n) a la derecha de cada nodo, indica la respuesta **CORRECTA**.



A. Una expansión en profundidad generará tantos o más nodos que una expansión en coste uniforme
B. La h(n) es admisible
C. El algoritmo A obtendrá la senda óptima.
D. Ninguna de las anteriores es cierta

---

4) La figura muestra el espacio que se genera con la aplicación de un algoritmo de búsqueda. Los nodos marcados en negrita son nodos solución, y el algoritmo encuentra el **nodo K** como primera solución. Los valores de las aristas representan el coste de los operadores. ¿Qué algoritmo de búsqueda no informada se ha aplicado?



A. Búsqueda en anchura.
B. Búsqueda por coste uniforme.
C. Búsqueda en profundidad con nivel máximo de profundidad m=3.
D. Búsqueda por profundización iterativa.

5) Sea el árbol de búsqueda de la pregunta anterior, y una función heurística h(n) que sabemos que es admisible. Indica la respuesta **INCORRECTA**:

A. h(J)=h(E)
B. g(L) <= g(D)+h(D)
C. h(C) <= 6
D. h*(A)=6

---

6) Dada la BH={(lista e b a c d e a) (lista1 a c a d e e f g)} y la regla R1, indica cuál será la BH final.

```
(defrule R1
  ?f <- (lista  $?x ?a $?y)
        (lista1  $?p ?a $?q ?a  $?r)
=>
      (retract ?f)
      (assert (lista $?x $?y)))
```

A. {(lista e b a c d f g) (lista1 a c a d e e f g)}
B. {(lista b) (lista1 a c a d e e f g) }
C. {(lista b c d) (lista1 a c a d e e f g)}
D. {(lista e b a c d) (lista1)}

---

7) Indica cuál es el resultado final CORRECTO tras ejecutar el siguiente SBR con la BH inicial={(lista 34 77 34)}:

```
(defrule R1
   (declare (salience 15))
     ?f <- (lista $?x1 ?num $?x2)
 =>
    (retract ?f)
    (printout t "Mensaje 1" crlf))
```

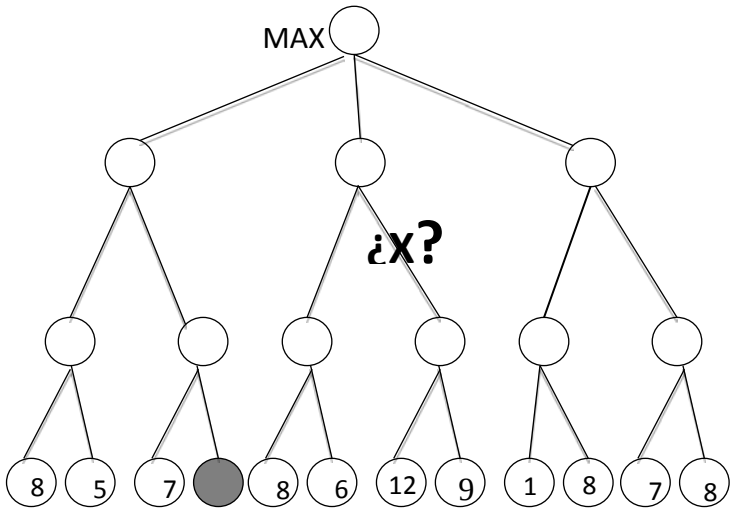
```
(defrule R2
     (declare (salience 20))
        ?f <- (lista ?num $?x ?num)
 =>
      (retract ?f)
      (printout t "Mensaje 2" crlf))
```

```
(defrule R3
  (declare (salience 30))
 =>
  (printout t "Mensaje 3"))
```

A. Mostrará tres veces el "Mensaje 3".
B. Mostrará una vez "Mensaje 2" y una vez "Mensaje 3".
C. Mostrará una vez "Mensaje 3" y una vez "Mensaje 2".
D. Mostrará una vez "Mensaje 3", una vez "Mensaje 2" y una vez "Mensaje 1".

---

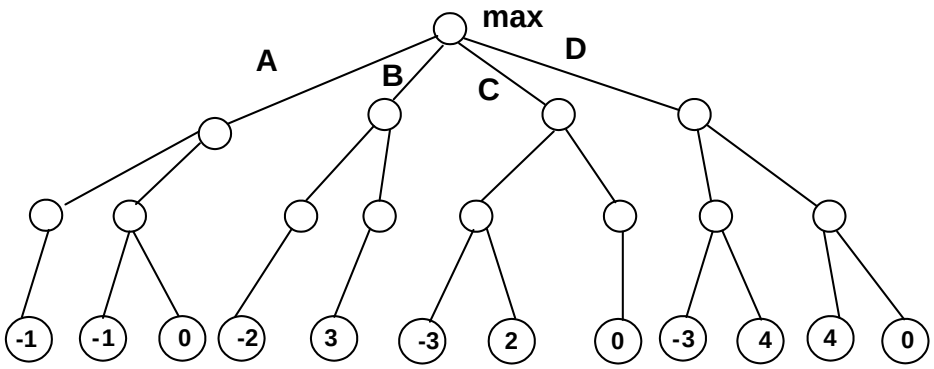
8) Dado el espacio de búsqueda de un juego representado en la figura siguiente, asumiendo que se aplica un procedimiento alfa-beta, indica el valor que debería tener el nodo terminal sombreado para que se produzca el corte ilustrado en la figura:



A. Con cualquier valor del nodo se produciría un corte
B. Menor que 8
C. Mayor o igual que 8
D. Nunca se podría producir el corte indicado (o ninguna de las anteriores)

---

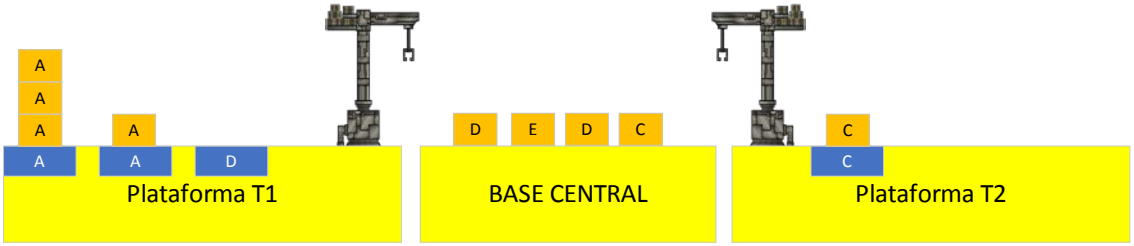
9) Dado el árbol de juego de la figura, ¿Cuántos nodos evitamos generar respecto a un algoritmo MINIMAX si realizamos una exploración alfa-beta?



A. 3
B. 4
C. 5
D. 6

Sistemas Inteligentes – Examen Bloque 1, 13 noviembre 2020
Problema: 2 puntos

En un puerto se dispone de un sistema para el manejo de contenedores. Existe una base central donde se han depositado contenedores de diferentes clases (A, B, C, ...). Todos los contenedores están accesibles, es decir están directamente en el suelo. Además de la clase, los contenedores pertenecerán a un tipo T1 o T2, cada contenedor según su clase pertenecerá a un tipo o a otro, pero nunca a ambos.



En el sistema existen también dos plataformas, cada una de ellas con su propia grúa. Cada plataforma y grúa corresponde a uno de los dos tipos de contenedor. De esta manera cada grúa sólo cogerá contenedores de su tipo correspondiente (1 cada vez) de la base central y los depositará sólo en su propia plataforma.

Cuando una grúa deposite un contenedor sobre su plataforma, siempre lo hará sobre una pila ya existente de la misma clase que dicho contenedor, siempre y cuando no rebase la altura máxima de 3 contenedores por pila.

La estructura del patrón dinámico será la siguiente (no modificable):

(puerto b^m plataforma T1 [pila c^s n^s]^m grúa T1 p^s plataforma T2 [pila c^s n^s]^m grúa T2 p^s)

donde: $b^m \in$ {A, B, C, D, …} cada elemento denota un contenedor de la clase correspondiente que está en la base central.
$c^s \in$ {A, B, C, D, ...} tipo de contenedor que se almacena en esa pila
$n^s \in$ [0 3] número de contenedores de la pila.
$p^s \in$ {vacio, A, B, C, …}. Clase del contenedor que ha cogido la grúa, si no ha cogido nada pondrá vacío.

NOTA: Para que una grúa pueda depositar el contenedor que mantiene en una pila, dicha pila (de la clase del contenedor) ya debe existir, aunque no tenga contenedores, en cuyo caso aparecerá como pila de dicha clase (A, B, etc.) con 0 contenedores. No es lo mismo una pila vacía (0 contenedores) a que no exista pila (en cuyo caso no aparece en el patrón). Inicialmente, en el sistema no hay ninguna pila. Las pilas vacías serán creadas mediante reglas.

Se pide usando CLIPS y exploración en grafos:

a) (0,3 puntos) Escribir la **base de hechos inicial** sabiendo que inicialmente en la base central se dispone de los siguientes contenedores: A, E, B, B, C, D, B, B, A, E, D. Se sabe que el tipo T1 corresponde a las clases A y D, y la T2 a la B, C y E. Ambas plataformas no contienen ninguna pila (ni siquiera vacías) y las grúas están vacías. Para los hechos estáticos usar el patrón que se desee.

b) (0,5 puntos) Escribir la **regla pila_inicial** (una única regla para crear pilas vacías en las plataformas). Si en la base central existe un contenedor de una clase determinada, en su plataforma no existe ninguna pila para dicha clase y su grúa está vacía, esta regla deberá crear una pila vacía de dicha clase en su plataforma correspondiente. La creación de esta pila vacía se representa en el dibujo como un rectángulo azul (gris oscuro en impresión sin color).

c) (0,7 puntos) Escribir la **regla coge_contenedor** (una única regla). La regla utilizará la grúa (vacía) de una plataforma (T1 o T2) y cogerá un contenedor de la base central que pertenezca a su tipo, con la condición de que ya exista una pila de su clase en su plataforma y no esté ya completa (no tenga 3 contenedores).

d) (0,5 puntos) Escribir la **regla parada**. Esta regla se disparará cuando el sistema haya dejado todos los contenedores en sus respectivas plataformas e imprimirá por pantalla el número total de pilas creadas.

```
(deffacts datos
      (puerto A E B B C D B B A E D plataforma T1 grua T1 vacio plataforma T2 grua
T2 vacio)
      (contenedor T1 A D)
      (contenedor T2 B C E)
)

;; al principio cree una pila vacía de un contenedor existente si no hay nada en su
plataforma ni en su grua
(defrule pila_inicial
      (puerto $?p1 ?p $?p2 plataforma ?t $?l grua ?t vacio $?r)
      (test (not (member$ ?p $?l)))
      (contenedor ?t $? ?p $?)
      =>
      (assert (puerto $?p1 ?p $?p2 plataforma ?t $?l pila ?p 0 grua ?t vacio $?r))
)

;;coge un contenedor solo si su grua está vacía y hay una pila no completa (menos de
3)

(defrule coge_contenedor
      (puerto $?p1 ?p $?p2 plataforma ?t $?e1 pila ?p ?c $?e2 grua ?t vacio $?r)
      (contenedor ?t $? ?p $?)
      (test (< ?c 3))
      =>
      (assert (puerto $?p1 $?p2 plataforma ?t $?e1 pila ?p ?c $?e2 grua ?t ?p $?r))
)

;;la grua deja un contenedor y no llena la pila o no quedan más de esa clase


(defrule parada
      (declare (salience 100))
      (puerto plataforma T1 $?p1 grua ? vacio  plataforma T2 $?p2 grua ? vacio )
      =>
      (printout t "el número de pilas es " (+ (div (length$ $?p1) 3)(div (length$
$?p2) 3)) crlf)
      (halt)
)
```