

**NOTA:** Consegnare la soluzione tramite un singolo file (word, pdf oppure txt)

**Esercizio 1 (6 punti)**

Si formalizzino le seguenti frasi in logica dei predicati:

1. I giocatori di scacchi sono persone intelligenti.
2. Qualunque giocatore di scacchi che vince si diverte.
3. Chiunque si diverte ed è intelligente, non è depresso.
4. Esiste un giocatore di scacchi che vince.

usando i seguenti predicati: **giocas(X)** (X è un giocatore di scacchi), **intelligente(X)** (X è una persona intelligente), **vince(X)** (X vince), **diverte(X)** (X si diverte), **depresso(X)** (X è depresso).

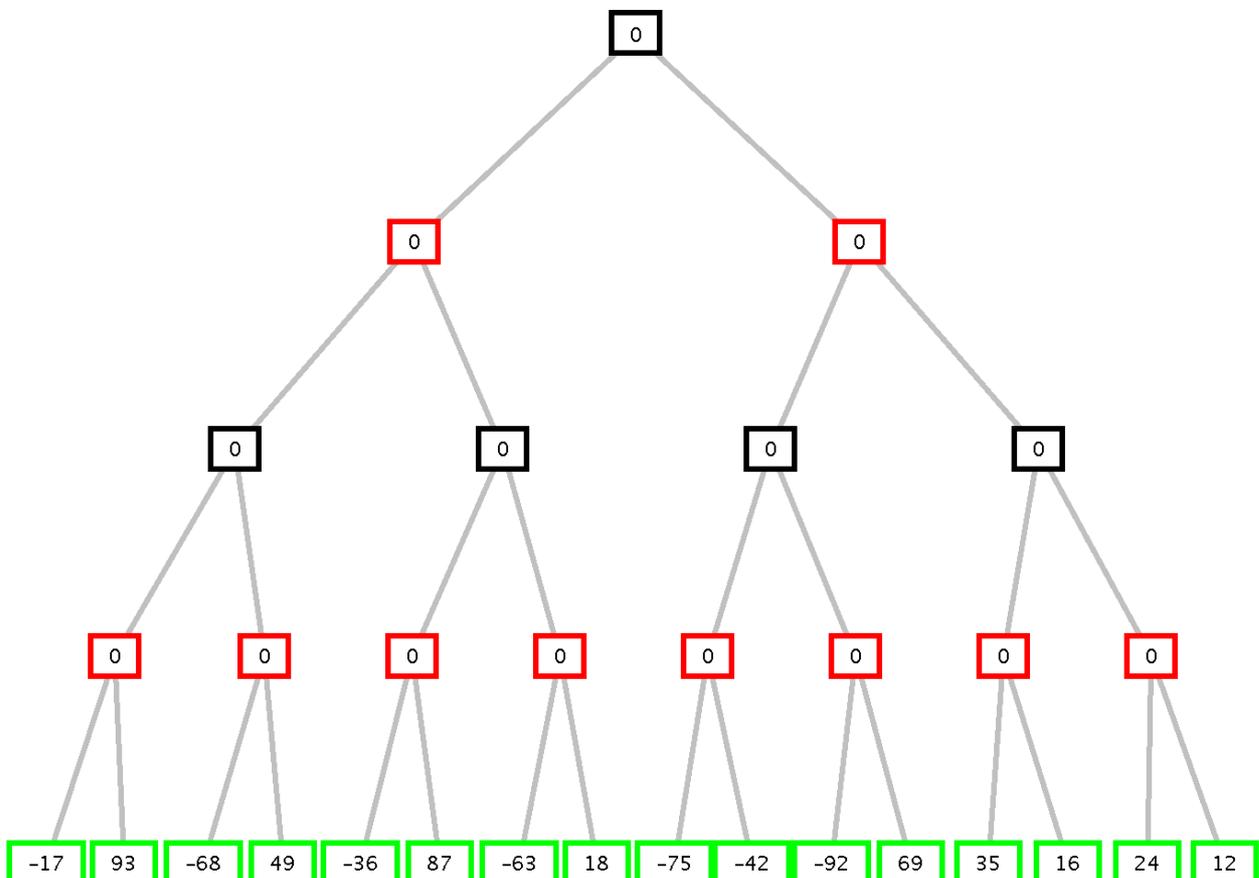
Le si trasformi in clausole e si usi poi il principio di risoluzione per dimostrare che c'è una persona intelligente che non è depressa.

**NOTA:** Si riportano i simboli degli operatori e quantificatori in logica:  $\forall \exists \wedge \vee \neg \rightarrow$

**Esercizio 2 (6 punti)**

Si consideri il seguente albero di gioco in cui il primo giocatore è MAX.

- a) Si indichi come l' algoritmo min-max risolve il problema indicando il valore con cui viene etichettato il nodo iniziale e la mossa selezionata dal primo giocatore (arco a1, o a2).
- b) Si mostrino poi i tagli che l' algoritmo alfa-beta consente indicando gli archi che verranno tagliati. Si indichino i nomi degli archi iniziando con la lettera "a" e facendola seguire con un numero crescente da sinistra a destra e dall'alto al basso. Ad esempio, i due archi che si dipartono dalla radice saranno nominati a1 (quello più a sinistra) e a2. L' arco che connette il nodo foglia più a sinistra (con valore -17) sarà denominato a15, mentre l' ultimo arco che connette il nodo foglia più a destra (valore 12) a30.



### Esercizio 3 (5 punti)

Si consideri il seguente CSP che lega le variabili A, B, C, D:

A::[4, 5, 6, 7, 8, 9, 10]      A>=B\*2  
B::[0, 1, 2, 3, 4, 5, 6]      C<=B-1  
C::[0, 1, 2, 3, 4, 5, 6]  
D::[0, 1, 2, 3, 4, 5, 6]      A=D-1

Si applichi, durante la ricerca fino alla prima soluzione, il Forward Checking dopo ogni passo di labeling, considerando, nella scelta della prossima variabile da istanziare l'euristica **Minimum Remaining Value** (poi, a parità di cardinalità di dominio, scegliere in base all'ordine alfabetico dei nomi delle variabili). Nel labeling, per il valore da assegnare alla variabile, si considerino i valori di dominio in ordine crescente, partendo dal più piccolo.

### Esercizio 4 (5 punti)

Si realizzi un predicato Prolog `lista_coppie(L1, L2, L3)` che date due liste **L1** e **L2** di interi di uguale lunghezza deve restituire una nuova lista **L3** contenente delle liste di due elementi `[el1,el2]` che appartengono a **L1** o **L2**, compaiono nelle stessa posizione nelle due liste e in cui il primo elemento `el1` è minore o uguale del secondo `el2`. Si tenga conto anche del possibile caso in cui **L1** e **L2** siano liste vuote: in tal caso **L3** sarà anche essa vuota.

Esempi:

```
?- lista_coppie([3,7,0,5],[4,6,0,3],L3).
```

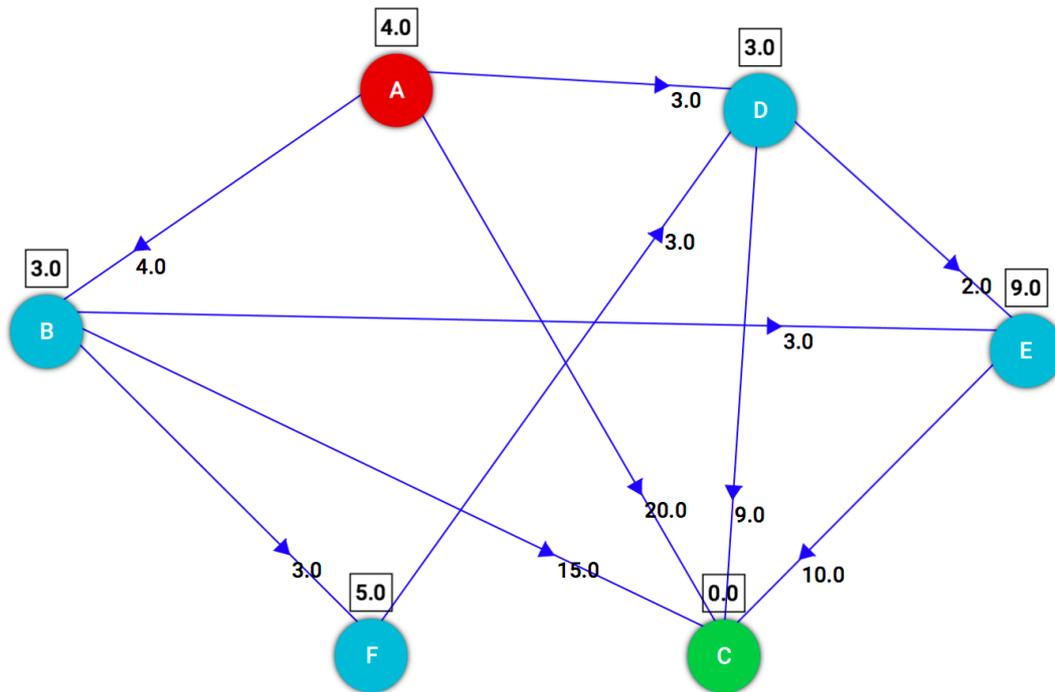
```
L3 = [[3, 4], [6, 7], [0, 0], [3, 5]]
```

```
?- lista_coppie([],[],L3).
```

```
L3 = []
```

### Esercizio 5 (6 punti)

Si consideri il seguente grafo, dove A è il nodo iniziale e C il nodo goal, e il numero associato agli archi è il costo dell'operatore per andare dal nodo di partenza al nodo di arrivo dell'arco. Vicino ad ogni nodo, in un quadrato, è indicata inoltre la stima euristica della sua distanza dal nodo goal C:



Si applichi la ricerca **A\*** su alberi (non tenendo quindi traccia dei nodi già visitati) indicando:

- i nodi espansi nell'ordine di espansione;
- i nodi sulla strada della soluzione e il costo della soluzione;
- se è garantita o meno l'ottimalità.

### Esercizio 6 (3 punti)

Si spieghi brevemente il predicato predefinito Prolog `cut`, le sue caratteristiche e si riporti un esempio che ne mostri l'utilizzo.

**Esercizio 1**

1.  $\forall X \text{ giocas}(X) \rightarrow \text{intelligente}(X)$
2.  $\forall X \text{ giocas}(X) \wedge \text{vince}(X) \rightarrow \text{diverte}(X)$
3.  $\forall X \text{ diverte}(X) \wedge \text{intelligente}(X) \rightarrow \neg \text{depresso}(X)$
4.  $\exists X \text{ giocas}(X) \wedge \text{vince}(X)$

Goal:  $\exists X \text{ intelligente}(X) \wedge \neg \text{depresso}(X)$

Clausole:

1.  $\neg \text{giocas}(X) \vee \text{intelligente}(X)$
2.  $\neg \text{giocas}(X) \vee \neg \text{vince}(X) \vee \text{diverte}(X)$
3.  $\neg \text{diverte}(X) \vee \neg \text{intelligente}(X) \vee \neg \text{depresso}(X)$ .
4.  $\text{giocas}(c1)$ . (c1 costante di Skolem)
5.  $\text{vince}(c1)$ .
6. G(Neg):  $\neg \text{intelligente}(X) \vee \text{depresso}(X)$ .

Risoluzione:

- 7: 6 + 1:  $\neg \text{giocas}(X) \vee \text{depresso}(X)$
- 8: 7 + 4:  $\text{depresso}(c1)$ .
- 9: 8 + 3:  $\neg \text{diverte}(c1) \vee \neg \text{intelligente}(c1)$
- 10: 9 + 2:  $\neg \text{giocas}(c1) \vee \neg \text{vince}(c1) \vee \neg \text{intelligente}(c1)$
- 11: 10 + 1:  $\neg \text{giocas}(c1) \vee \neg \text{vince}(c1)$
- 12: 11 + 4:  $\neg \text{vince}(c1)$
- 13: 12 + 5: Clausola vuota Contraddizione!!

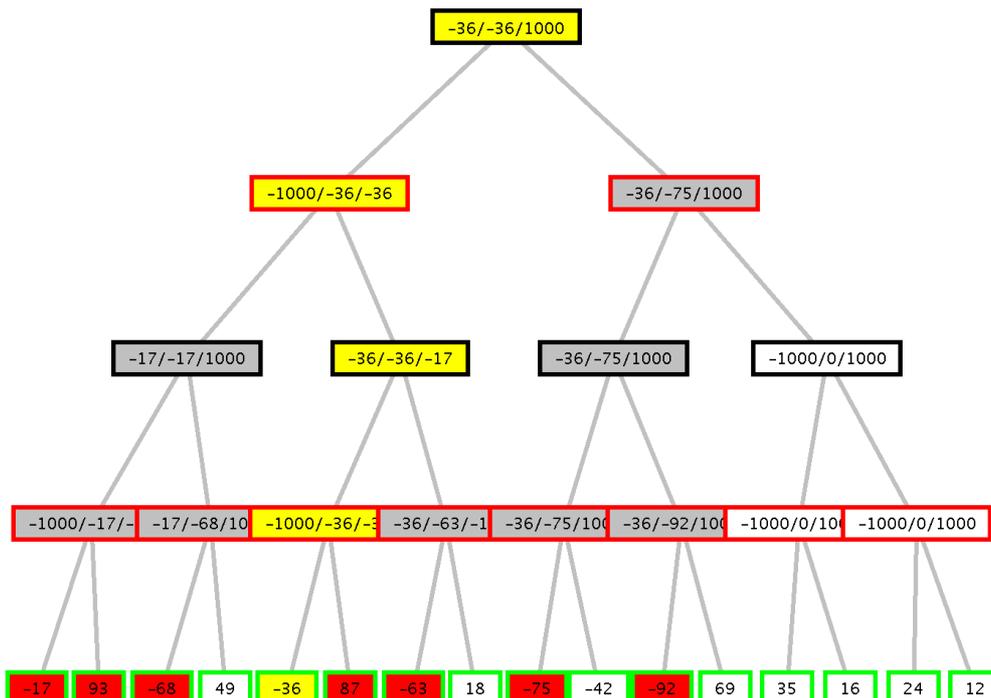
Oppure (qualche passaggio in meno):

- 7: 6+3:  $\neg \text{diverte}(X) \vee \neg \text{intelligente}(X)$
- 8: 7+2:  $\neg \text{giocas}(X) \vee \neg \text{vince}(X) \vee \neg \text{intelligente}(X)$
- 9: 8+1:  $\neg \text{giocas}(X) \vee \neg \text{vince}(X)$
- 10: 9+4:  $\neg \text{vince}(c1)$
- 11: 10+5: clausola vuota

**Esercizio 2**

Min-max: strada in giallo, ma senza i tagli – valore nodo radice -36, ramo a1.

In rosso i nodi espansi, in giallo la strada trovata, i nodi in bianco non sono esplorati per effetto dei tagli alfa-beta.



Archi tagliati a6, a18, a22, a24, a26. In totale cinque tagli.  
 Scelta per il ramo a1, valore propagato -36.

### Esercizio 3

Con euristica MRV:

	<b>A</b>	<b>B</b>	<b>C</b>	<b>D</b>
Labeling	<b>A=4</b>	[0...6]	[0...6]	[0...6]
FC		[0...2]	[0...6]	<b>[5]</b>
Labeling	A=4	[0...2]	[0...6]	<b>D=5</b>
FC		[0...2]	[0...6]	
Labeling	A=4	<b>B=0</b>	[0...6]	D=5
FC			<b>[]</b>	
<b>Backtracking</b>				
Labeling	A=4	<b>B=1</b>	[0...6]	D=5
FC			<b>[0]</b>	
Labeling	A=4	B=1	<b>C=0</b>	D=5
<b>Soluzione:</b>	<b>A=4</b>	<b>B=1</b>	<b>C=0</b>	<b>D=5</b>

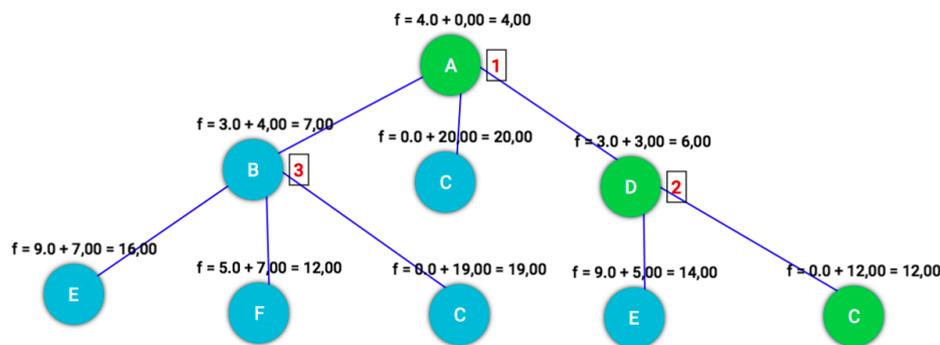
### Esercizio 4

```

lista_coppie([], [], []):-!.
lista_coppie([E11|Y1],[E12|Y2],[[E11,E12]|Y3]):-
    E11 =< E12, !, lista_coppie(Y1,Y2,Y3).
lista_coppie([E11|Y1],[E12|Y2],[[E12,E11]|Y3]):- lista_coppie(Y1,Y2,Y3).
    
```

### Esercizio 5

Con A\*, i nodi espansi sono ADBC, la soluzione trovata ADC è ottimale perché l'euristica è ammissibile e il costo della soluzione trovata è 12:



Show operations

- 1) A [f = 4.0 + 0.00 = 4.00]
- 2) D [f = 3.0 + 3.00 = 6.00]
- 3) B [f = 3.0 + 4.00 = 7.00]
- /) E [f = 9.0 + 7.00 = 16.00]
- /) F [f = 5.0 + 7.00 = 12.00]
- /) C [f = 0.0 + 19.00 = 19.00]
- /) C [f = 0.0 + 20.00 = 20.00]
- /) E [f = 9.0 + 5.00 = 14.00]
- /) C [f = 0.0 + 12.00 = 12.00]

Path cost: 12.0  
 Nodes expanded: 3  
 Queue size: 5  
 Max queue size: 6

### Esercizio 6

Si vedano le slide del corso.