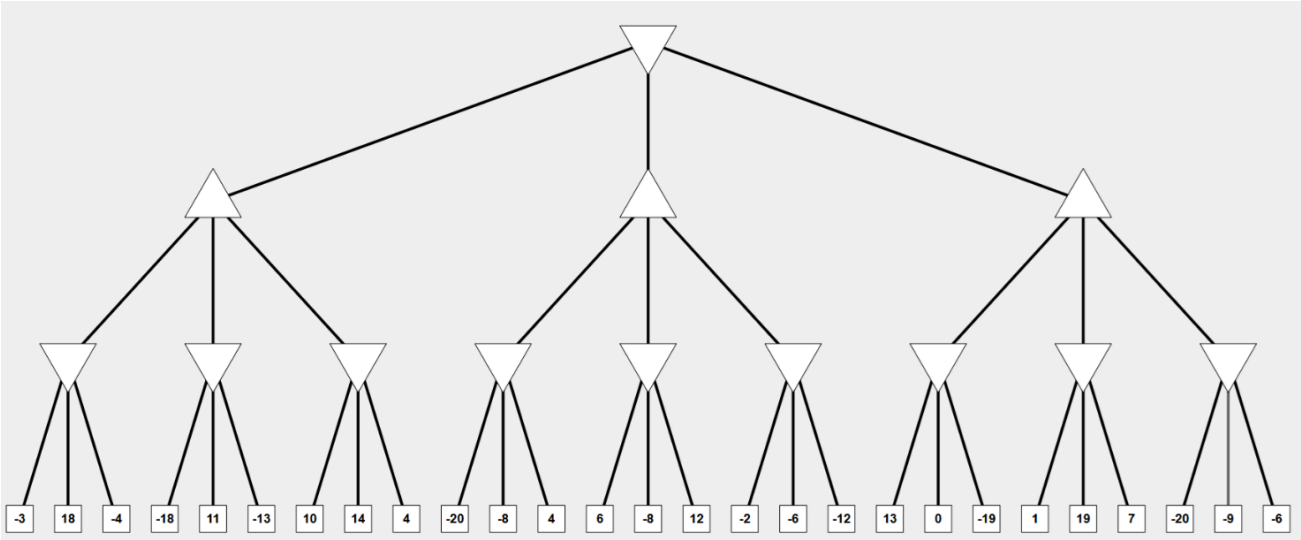
Esercizio 1 – Ricerca nello Spazio degli Stati

Dato il seguente albero di gioco, applicare l'algoritmo alpha-beta pruning. Riportare il valore di utilità propagato per MIN e MAX, e riportare anche i valori di alpha e beta propagati.



Esercizio 2 – Rappresentazione della Conoscenza

Utilizzando una **rete semantica partizionata**, rappresentare la seguente conoscenza: "*Peter afferma che tutti i giocatori hanno disputato la partita.*"

Utilizzando un **grafo concettuale**, rappresentare la seguente conoscenza: "*Il professore afferma che gli studenti superano il corso se il voto è maggiore di 18*".

Esercizio 3 - Ontologie

Data la seguente ontologia: https://www.w3.org/TR/owl-guide/wine.rdf

1) **Estendere l'ontologia** aggiungendo almeno un paio di object properties e data properties, inserendo delle restrizioni (almeno un paio di Value Constraints e Cardinality Constraints), e popolando l'ontologia con almeno 5 individui.

2) Effettuare le seguenti **query SPARQL**, fornendo una descrizione delle query e uno snapshot dei risultati (ottenuti tramite Protegè):
    a. Scelta una classe C, una data property P ed un Literal L, ricercare tutte le istanze di C in cui è applicata P e in cui P assume un valore L.
    b. Ricerca tutte le istanze della classe "Wine"
    c. Ricercare il numero dei vini che hanno come "VintageYear" il 1977
    d. Ricercare (tutti i vini provenienti da una "ItalianRegion" AND il cui colore è "Rose") OR (tutti i vini provenienti da "ChiantiRegion")
    e. Data una Object Property O, ricercare il range ed il dominio

3) Realizzare le seguenti **Regole SWRL** (fornire anche uno snapshot del risultato della regola eseguita tramite Protegè):

   a. Scegli o crea tre classi differenti, qui indicate in generale come A, B e C, e una object property P (Puoi usare quelle già presenti o che hai creato in precedenza). Scrivi una regola per cui se esiste la relazione P tra due individui di A e B, allora l'individuo di A appartiene anche alla classe C.
   b. Scegli o crea tre classi differenti, qui indicate in generale come A, B e C, e due object properties P1 e P2 (Puoi usare quelle già presenti o che hai creato in precedenza).
   Scrivi una regola per cui se esiste una relazione P tra due individui di A e B, allora esiste una relazione P2 tra individui di A e C.
   c. Scegli o crea tre classi differenti, qui indicate in generale come A, B e una data property D1 (Puoi usare quelle già presenti o che hai creato in precedenza). Scrivi una regola per cui se un individuo di A possiede la proprietà D, ed essa si trova in un range (a tua scelta), allora esso è individuo anche di B.
   a. Se W è un vino il cui "VintageYear" è antecedente al 1970 e proveniente dalla "ItalianRegion" e prodotto dalla Winery "Foxen", allora W è un vino pregiato. Se per eseguire la regola occorrono proprietà non presenti nell'ontologia, si è liberi di inserirle.

Esercizio 4 – Logica - Creazione Funtore

Dati i seguenti fatti costituenti gli archi di un grafo, scrivere un programma in Prolog che ricerchi tutti i possibili percorsi tra due nodi. Costruire ricorsivamente una lista di funtori "Percorso".

```
edge(placeA, placeB,).
edge(placeA, placeC).
edge(placeA, placeD,).
edge(placeA, placeE).
edge(placeA, placeF).
edge(placeB, placeE).
edge(placeB, placeC).
edge(placeC, placeD).
edge(placeC, placeH).
edge(placeD, placeH).
edge(placeE, placeG).
edge(placeG, placeH).
edge(placeF, placeH).
```

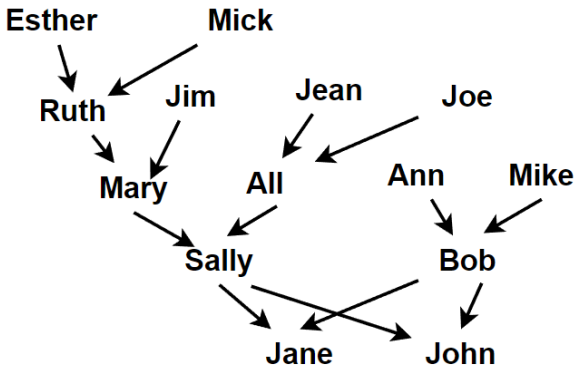
Esempio Output: ListaPercorsi = [percorso1(edge(placeA, placeB), edge(placeB, placeC), …), percorso2(…), …., percorsoN()]

Esercizio 4 – Logica - Creazione Funtore

Dati i seguenti fatti costituenti le relazioni familiari di un albero genealogico, scrivere un programma in Prolog che costruisca una struttura albero. La creazione deve essere ricorsiva.

```
parents(sally,jane).
parents(bob,jane).
parents(sally,john).
parents(bob,john).
parents(mary,sally).
parents(all,sally).
parents(ann,bob).
parents(mike,bob).
parents(jean,all).
parents(joe,all).
parents(ruth,mary).
parents(jim,mary).
parents(esther,ruth).
parents(mick,ruth).
```

Le relazioni sono rappresentante anche graficamente con la seguente figura:



Output atteso: lista fi funtori albero. Ogni funtore deve contenere tutta la discendenza di un personaggio.

Esercizio 4 – Logica - Ispezione Funtore

Dato il seguente funtore "albero binario": "*Albero = nodo1(nodo2(nodo4, nodo5), nodo3(nodo6, nodo7(nodo8, nodo9)))*", costruire un predicato ricorsivo che ispeziona l'albero stampando tutti i suoi nodi. Quando l'albero non ha più figli, stampare "Foglia vuota".