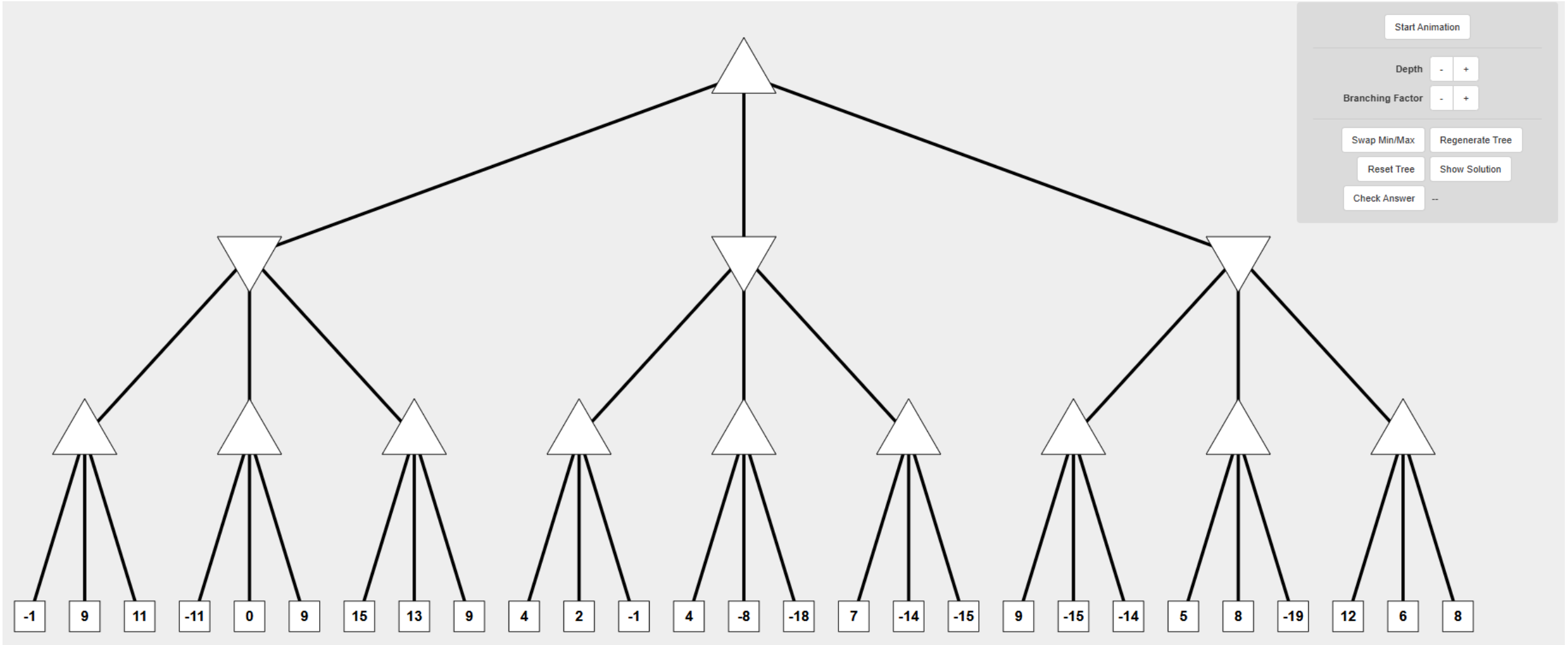
ESERCITAZIONE KEAI 2024/25

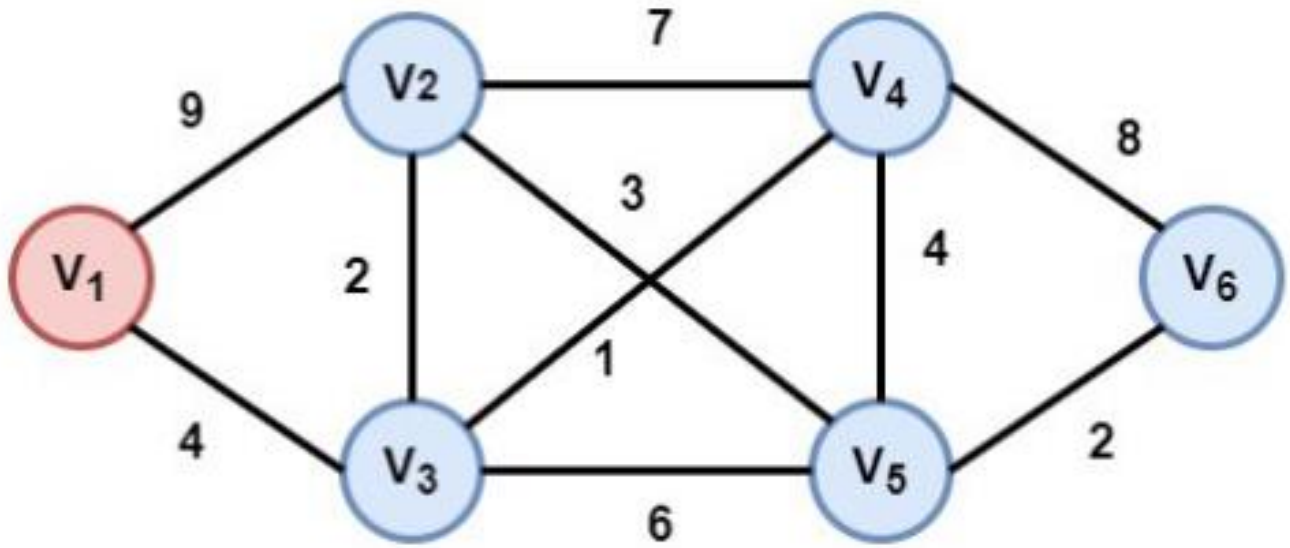
Esercizio Ricerca nello Spazio degli Stati: Alpha Beta Pruning

https://pascscha.ch/info2/abTreePractice/

MINI: registra in **beta;**
MAX: registra in **alpha;**
Pruning se beta <= alpha



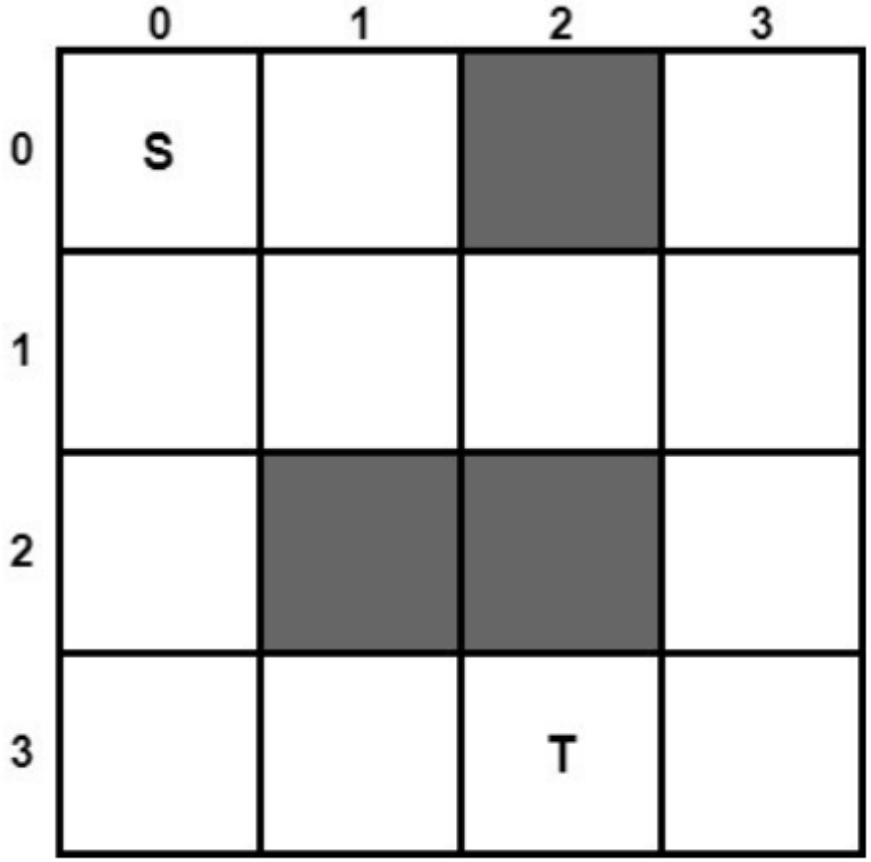
Esercizio Ricerca nello Spazio degli Stati: UCS



Pace	Opened	Closed
0	$\{(V_1,0)\}$	{-}

Esercizio Ricerca nello Spazio degli Stati: A*

- Supponiamo di avere un robot e di volere che il robot navighi dal punto S in posizione (0, 0) al punto T in posizione (3, 2).
- I quadrati grigi sono ostacoli che il robot non può superare.
- Come funzione euristica, utilizzeremo la distanza di Manhattan (nel calcolo non si tiene conto degli ostacoli).



Esercizio Rappresentazione della Conoscenza

Utilizzando una rete semantica partizionata, rappresentare la seguente conoscenza:

Marco afferma che il tennis è interessante

Esercizio Rappresentazione della Conoscenza

Utilizzando un grafo concettuale, rappresentare la seguente conoscenza:

Matteo pensa che i cantanti amano fare i concerti

Esercizio Prolog 1: Liste e Ricorsione

Data una lista di interi, iterare la lista, stampando su un file solo i valori della lista maggiori di 5

- **open(path, mode, fd)**: per aprire un file in uno specifico **path**. Se il file non esiste viene creato secondo una delle modalità descritte dal secondo parametro "**mode**": se viene specificata l'opzione "**write**" il vecchio contenuto del file viene rimpiazzato con il nuovo ogni volta che viene effettuata una open, mentre se viene specificata l'opzione "**append**" il nuovo contenuto viene aggiunto in coda al vecchio. Usare **fd** per riferirsi al file.
- **write(fd, testo)**: per scrivere sul file.
- **close(fd)**: è buona norma chiudere il file una volta terminato l'utilizzo.

Esercizio Prolog 2: Liste e Ricorsione

Creare una regola ricorsiva che prende in ingresso una lista di stringhe, e torna in output la stessa lista dove ad ogni valore viene concatenata la stringa "ciao"

- **atom_concat(Stringa1, StringaDaAgg, NewElement)**: concatena alla Stringa “Stringa1” una seconda stringa “StringaDaAgg”. La variabile NewElement contiene la stringa finale risultato della concatenazione

Esercizio Prolog 3: Creazione Ricorsiva di un Funtore

Creare un predicato build functor/3 che costruisce ricorsivamente un albero binario di profondità specificata.

build functor(Depth, Value, Tree)

L'albero avrà la forma: node(Left, Right)

Esercizio Prolog 4: Ispezione Ricorsiva di un Funtore

Creare una regola ricorsiva che ispeziona il seguente funtore:

computer(cpu(intel_i9), componenti(ram(sedici_gb), gpu(nvidia_rtx_3080)))

La regola deve stampare il nome di tutte le componenti. Quando l'arità di un componente è 0, stampare "Componente non specificato.

Esercizio Prolog 5: Attraversamento di Grafi

Siano dati i seguenti fatti:

```
edge(a,b).
edge(c,d).
edge(a,c).
edge(d,e).
edge(b,d).
edge(f,g).
```

Creare un regola che ricerca tutti i percorsi tra due nodi e li inserisce in una lista.

Esercizio Prolog 5: Attraversamento di Grafi

Siano dati i seguenti fatti:

```
edge(a,b, 2).
edge(c,d, 6).
edge(a,c, 6).
edge(d,e, 5).
edge(b,d, 2).
edge(f,g, 5).
```

Creare un regola che ricerca il path con costo minimo.

Esercizio Prolog 6: Alberi

Creare una regola che verifica se due alberi sono isomorfi

Traccia PIZZA:

Data la seguente ontologia: https://protege.stanford.edu/ontologies/pizza/pizza.owl

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 "Pizza" per cui è stata scelta la base B ed almeno un ingrediente di Tipo I OR tutte le pizze di tipo "SpicyPizza".
    c. Ricercare il numero delle pizze presenti nell'ontologia.
    d. Ricercare tutte le pizze che contengono la stringa "margherita" nel nome. Porre un limite a 2 risultati.
    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.
    d. Se P è una Pizza, se possiede almeno un ingrediente di tipo "Mozzarella" ed un ingrediente di tipo "Pomodoro" e la base è di tipo "DeepPanBase", allora la pizza è una margherita.