

## Corrigé type du rattrapage d'IA

### Exercice01

1.
  - a. En rajoutant un nœud avec trois arcs.
  - b. En rajoutant un Not sur l'arc
  - c. Un nœud cible avec deux arcs entrants
  - d. Un nœud source avec deux arcs sortants
2. On identifie les relations 'arcs' à partir des verbes. Les nœuds sont les noms et les constantes de type vrai/faux.

### Exercice 02

```
deplacer(est,ouest).
deplacer(ouest,est).
transition([X,X,Chevre,Chou],wolf,[Y,Y,Chevre,Chou]) :- deplacer(X,Y).
transition([X,Loup,X,Chou],goat,[Y,Loup,Y,Chou]) :- deplacer(X,Y).
transition([X,Loup,Chevre,X],chou,[Y,Loup,Chevre,Y]) :- deplacer(X,Y).
transition([X,Loup,Chevre,Chou],rien,[Y,Loup,Chevre,Chou]) :- deplacer(X,Y).
sure([X,_X,_]). % le berger est la chevre se trouvent sur le meme coté du fleuve.
sure([X,X,_X]). % le berger, le loup et le chou se trouvent sur le meme coté du fleuve
resoudre([est,est,est,est],[ ]).
resoudre(Etat,[X|R]) :- transition(Etat, X, EtatSuivant), sure(EtatSuivant), resoudre(EtatSuivant, R).
%?- length(X,7), resoudre([ouest,ouest,ouest,ouest],X).
```

### Exercice03

```
schedule(Processes, Quantum, Schedule) :- ready_queue(Processes, [], ReadyQueue),
schedule_loop(ReadyQueue, Quantum, [], Schedule).
schedule_loop([], _, AccumulatedSchedule, AccumulatedSchedule).
schedule_loop(ReadyQueue, Quantum, AccumulatedSchedule, FinalSchedule) :-
dequeue(ReadyQueue, Process, RemainingProcesses),
execute_process(Process, Quantum, RemainingTime, UpdatedProcess),
append(AccumulatedSchedule, [(Process, Quantum)], TempSchedule),
( RemainingTime > 0 ->
enqueue(UpdatedProcess, RemainingProcesses, NewReadyQueue)
; NewReadyQueue = RemainingProcesses
),
schedule_loop(NewReadyQueue, Quantum, TempSchedule, FinalSchedule).
ready_queue([], Acc, Acc).
ready_queue([Process|Processes], Acc, ReadyQueue) :-
enqueue(Process, Acc, NewAcc),
ready_queue(Processes, NewAcc, ReadyQueue).
execute_process(process(Id, Burst), Quantum, RemainingTime, process(Id, RemainingTime)) :-
( Burst > Quantum ->
RemainingTime is Burst - Quantum
; RemainingTime = 0
).
enqueue(Process, Queue, [Process|Queue]).
add_to_end(Process, [], [Process]).
add_to_end(Process, [H|T], [H|Rest]) :-
add_to_end(Process, T, Rest).
```