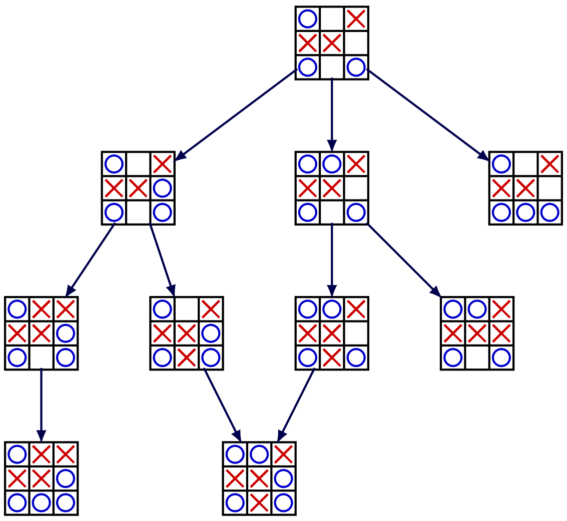
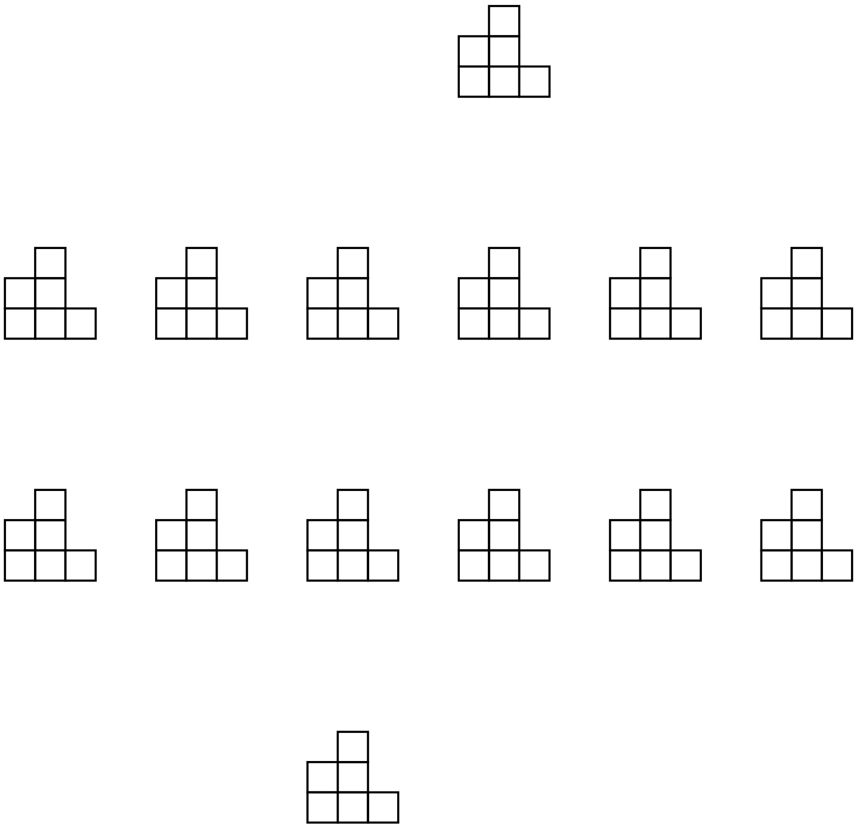
Jeux

Exercice 1.3 :



Exercice 1.4



Exercice 2.1 :

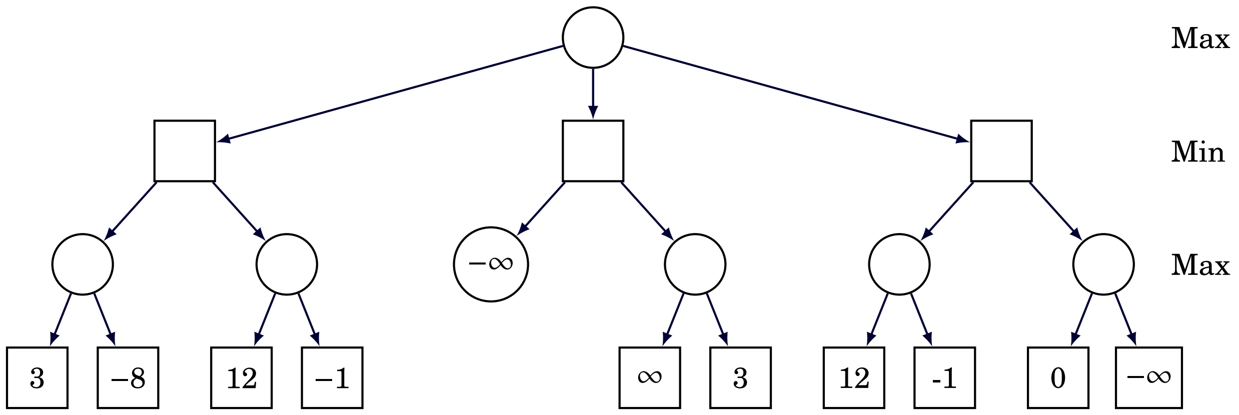
```
let rec min_max jr etat    .    =
    match coups jr etat with
    | [] -> let g = gagnant etat in
            if g = 1 then -1 else g+1
    | l -> let f, deb = if jr = 0 then max, ref (-1) else min, ref 1 in
            let h successeur =
                .
                .
                .
                deb := f !deb v
            in
            List.iter h l;
            !deb
```

Exercice 2.2 :

```
let heuristique etat =
    match etat.(1).(1) with
    | 'O' -> 50
    | 'X' -> -50
    | _ -> 0

let rec min_max jr etat nbcoups =
    match .                    with
    | .       -> let g = gagnant etat in
                if g = 1 then -100 else 100*(g+1)
    | .
    | .       -> let f, deb = if jr = 0 then max, ref (-100) else min, ref 100 in
                let h successeur =
                    let v = min_max (1-jr) successeur (nbcoups-1) in
                    deb := f !deb v
                in
                List.iter h l;
                !deb
```

Exercice 2.3 :



Élagage alpha-beta :

```
let rec min_max jr etat nbcoups alpha beta =
    match                         with
    |         -> let g = gagnant etat in
            if g = 1 then -100 else 100*(g+1)
    |
    |         -> let deb = if jr = 0 then ref (-100) else ref 100 in
           let h successeur =
                if jr = 0 then begin
                    if not (!deb >= beta) then begin
                        let v = min_max (1-jr) successeur (nbcoups-1) (max !deb alpha) beta in
                        deb := max v !deb
                    end end;
                if jr = 1 then begin
                    if not (!deb <= alpha) then begin
                        let v = min_max (1-jr) successeur (nbcoups-1) alpha (min !deb beta) in
                        deb := min v !deb
                    end end
            in
            List.iter h l;
            !deb

let rec alphabeta jr etat nbcoups = min_max jr etat nbcoups (-100) 100
```