Games

```
function ALPHA_BETA(s) returns an action
   v ← MAX_VALUE(s,-∞,+∞)
   return a ∈ actions(s) with value v
```

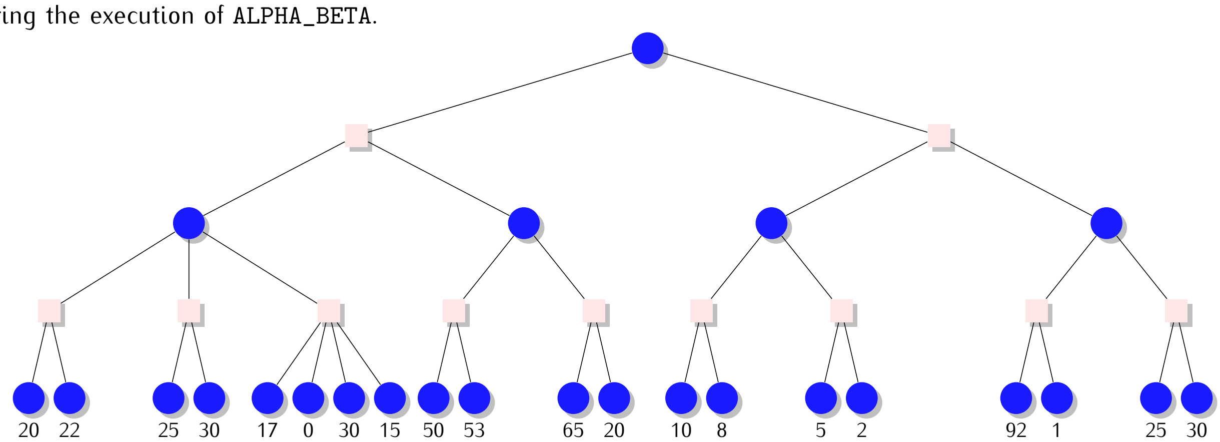
- result(s,a) is the state achieved when we take action a in state s
- α gives a bound on the worst possible value for max
- β gives a bound on the worst possible value for min

```
function MIN_VALUE(s,α,β) returns a utility value
     if terminal?(s) then return utility(s)
     v ← +∞
     for each a ∈ actions(s) do
          v ← min{v,MAX_VALUE(result(a,s),α,β)}
          if v ≤ α then return v
          β ← min{β,v}
     return v
```

```
function MAX_VALUE(s,α,β) returns a utility value
     if terminal?(s) then return utility(s)
     v ← −∞
     for each a ∈ actions(s) do
          v ← max{v, MIN_VALUE(result(a,s),α,β)}
          if v ≥ β then return v
          α ← max{α,v}
     return v
```

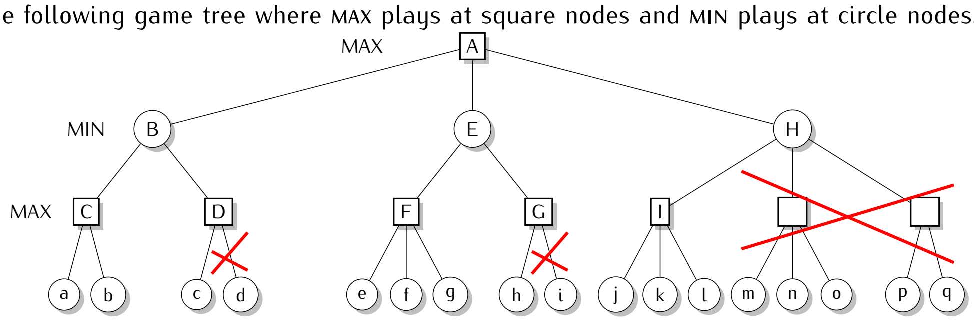
Exercise 1

Apply the algorithms MINIMAX and ALPHA_BETA to the game tree below, where the circles are nodes where MAX plays and the squares are the nodes where MIN plays. Show the values for α, β and v for each node during the execution of ALPHA_BETA.



Exercise 2

Consider the following game tree where MAX plays at square nodes and MIN plays at circle nodes.



- Give values to the leaves from a up to q such that the algorithm α-β cuts *exactly* the indicated branches.
- Apply the algorithm on the tree with your values.