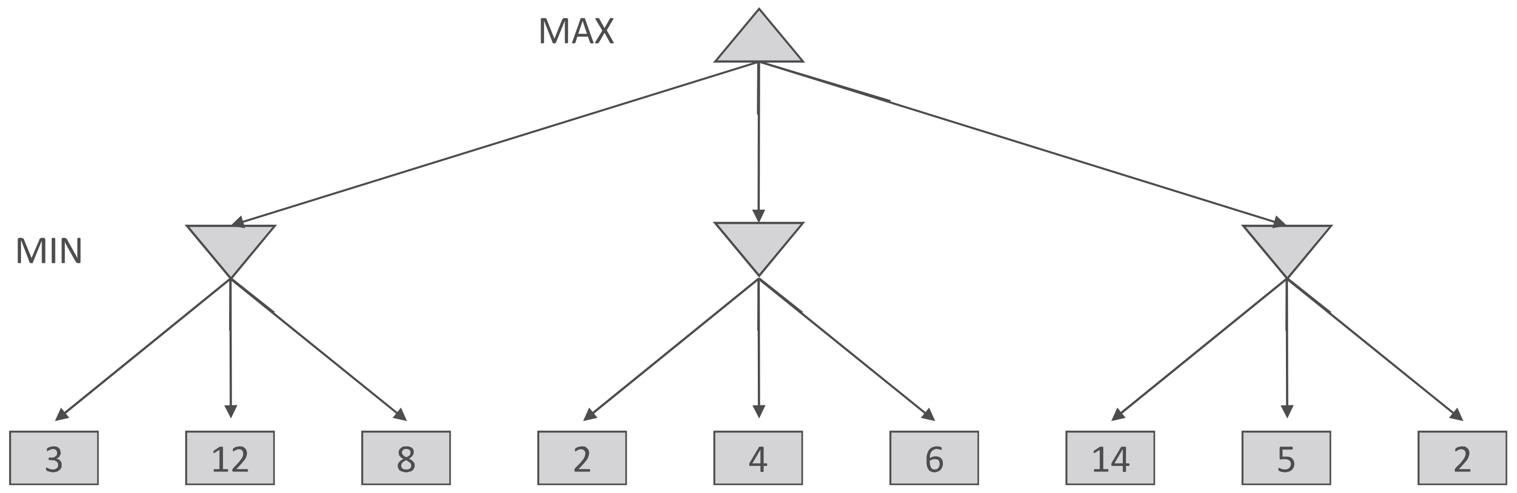
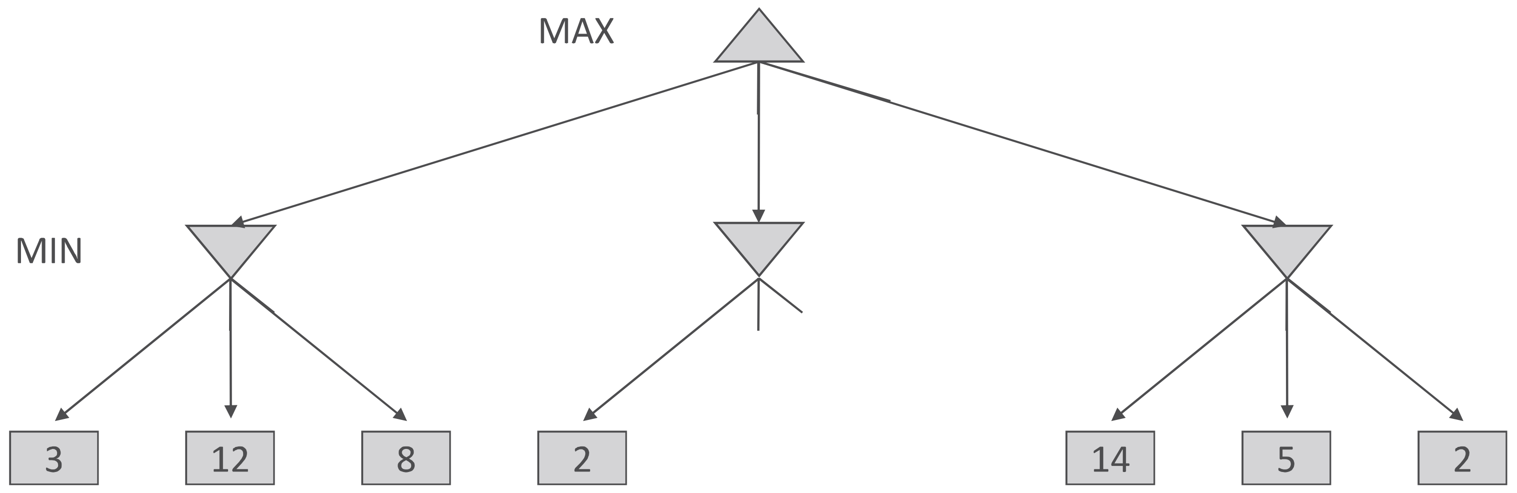
Minimax Example



Minimax Pruning



Alpha-Beta Pruning

- **Alpha** α: value of the best choice so far for MAX (lower bound of Max utility)
- **Beta** β: value of the best choice so far for MIN (upper bound of Min utility)
- Expanding at MAX node **n**: update α
  - If a child of **n** has value greater than β, stop expanding the MAX node **n**
  - Reason: MIN parent of **n** would not choose the action which leads to **n**

- At MIN node **n**: update β
  - If a child of **n** has value less than α, stop expanding the MIN node **n**
  - Reason: MAX parent of **n** would not choose the action which leads to **n**

Alpha-Beta Implementation

```
def value(state, $\alpha$, $\beta$):
    if the state is a terminal state: return the state's utility
    if the next agent is MAX: return max-value(state, $\alpha$, $\beta$)
    if the next agent is MIN: return min-value(state, $\alpha$, $\beta$)
```

```
def max-value(state, $\alpha$, $\beta$):
    initialize $v = -\infty$
    for each successor of state:
        $v = \max(v, \text{value}(\text{successor}, \alpha, \beta))$
        if $v \geq \beta$ return v
        $\alpha = \max(\alpha, v)$
    return v
```

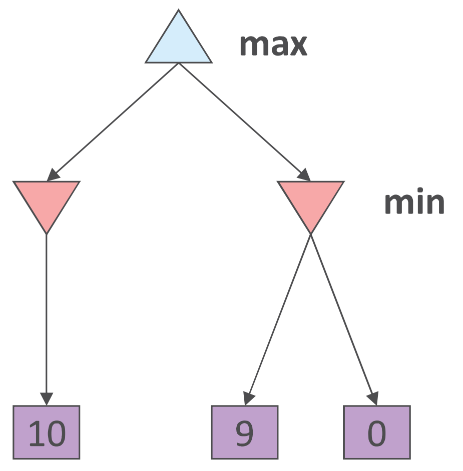
```
def min-value(state , $\alpha$, $\beta$):
    initialize $v = +\infty$
    for each successor of state:
        $v = \min(v, \text{value}(\text{successor}, \alpha, \beta))$
        if $v \leq \alpha$ return v
        $\beta = \min(\beta, v)$
    return v
```

Alpha-Beta Pruning Properties

- This pruning has no effect on minimax value computed for the root!

- Values of intermediate nodes might be wrong
  - Important: children of the root may have the wrong value
  - So the most naïve version won’t let you do action selection

- Good child ordering improves effectiveness of pruning


max
min
10
9
0