α-β Pseudocode:

```
alpha-beta(state, α, β, depth_limit):
    base case:
    if @ leaf node:
        return [state], static_eval of state
        depending on max|min, static_eval)
    elif depth_limit == 0:
        find heuristic based on max|min
        return [state], heuristic, staticeval)

    if max:
        for each child in state:
            α = max(α, alpha-beta(child, α, β, depth-1
            * if α ≥ β:
                break (blc α β will never let get bigger) - prune
            path = path + recurse(child)
            static_count ± recurse(child)
            score = max(recurse, score)
            return (best + path, maxsoc, static)
```

```
        return α
    if min:...
        for each child in state:
            β = min(β, alphabeta(child, α, β, depth-1))
            * if β ≤ α:
                break

        path = path + recurse(child)
        static_coort ± recurse(child)
        score = max(recurse, score)
        return (best_path, max_score, static)
```