My turn:

```
minimax(s,game,depth):
    if game.terminal(s) or depth=0
    then game.f(s)
    else
        val = game.minval
        foreach a ∈ game.actions(s)
            val' = maximin(game.child(a,s),
                           game,depth-1)
            if val' > val then
                val = val'
                besta = a
    return val
```

Opponent's turn:

```
maximin(s,game,depth):
    if game.terminal(s) or depth=0
    then game.f(s)
    else
        val = game.maxval
        foreach a ∈ game.actions(s)
            val' = minimax(game.child(a,s),
                           game,depth-1)
            if val' < val then
                val = val'
                besta = a
    return val
```

Initial call:

```
minimax(initialstate,game,maxdepth)
```

My turn:

```
minimaxAB(s,game,depth,a,b):
    if game.terminal(s) or depth=0
    then game.f(s)
    else
        val = game.minval
        foreach op ∈ game.actions(s)
            val' = maximinAB(game.child(op,s),
                        game,depth-1,max(a,val),b)
            if val' ≥ b then return val
            if val' > val then
                val = val'
                bestaction = op
        return val
```

Opponent's turn:

```
maximinAB(s,game,depth,a,b):
    if game.terminal(s) or depth=0
    then game.f(s)
    else
        val = game.maxval
        foreach op ∈ game.actions(s)
            val' = minimaxAB(game.child(op,s),
                        game,depth-1,a,min(b,val))
            if val' ≤ a then return val
        if val' < val then
            val = val'
            besta = op
    return val
```

Initial call:

```
minimax(initialstate,game,maxdepth,
                game.minval,game.maxval)
```