Algorithm 1 myAIAlgorithm

```
Input List<positionTicTacToe> $board$, int $player$
Return positionTicTacToe $myNextMove$

procedure myAIAlgorithm($board, player$)

    initialization

    $winMove \leftarrow getWinMove(player)$                ▷ If we have a win move
    if $winMove$ exists then
        return $winMove$

    $forceMove \leftarrow getForceMove(player)$            ▷ If we have a force move
    if $forceMove$ exists then
        return $forceMove$

    $coreMove \leftarrow getFirstTwoSteps(player)$ ▷ Occupy the strongest points
    if $coreMove$ exists then
        return $coreMove$

    $maxValue = -\infty$
    $positionTicTacToe\ myNextMove$

    do                                                     ▷ Progressive deepening
        for <each available move $curMove$> do
            <make current move>
            $newVale = miniMax(depth, player, false, -\infty, +\infty)$
            if $newValue > maxValue$ then
                $maxValue = newValue$
                $myNextMove = curMove$
            <cancel current move>                          ▷ Backtracking
    while <time is still enough>
```

Algorithm 2 miniMax

```
    Input int $depth$, int $player$, boolean $maximizingPlayer$, int $alpha$, int $beta$
    Return int $value$
    procedure miniMax
        if $depth == 0$ then                                  ▷ search finish
            return $evaluation(player)$         ▷ evaluate the board configutaion
        if $maximizingPlayer$ then                            ▷ Maximizer
            $maxValue = newValue$
            $myNextMove = curMove$
        else                                                  ▷ Minimizer
            aaa
```