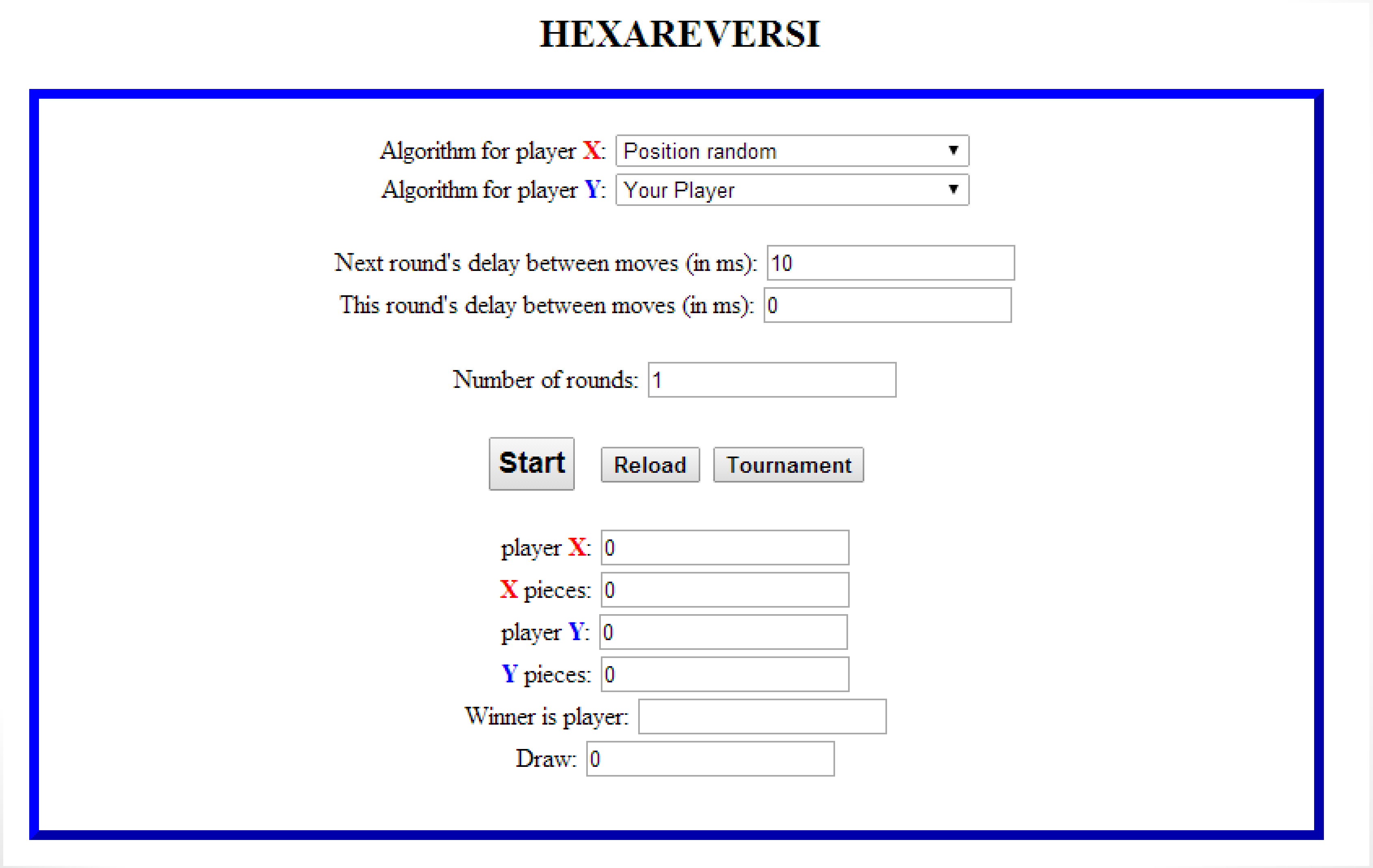
HEXAREVERSI

Project Description

By: Group t77
Lu Si Hong(A0094511L)
Ling Xiao Xiong Shaun (A0112036Y)

Implementation of UI



HEXAREVERSI

Algorithm for player X:	Position random
Algorithm for player Y:	Your Player
Next round's delay between moves (in ms):	10
This round's delay between moves (in ms):	0
Number of rounds:	1

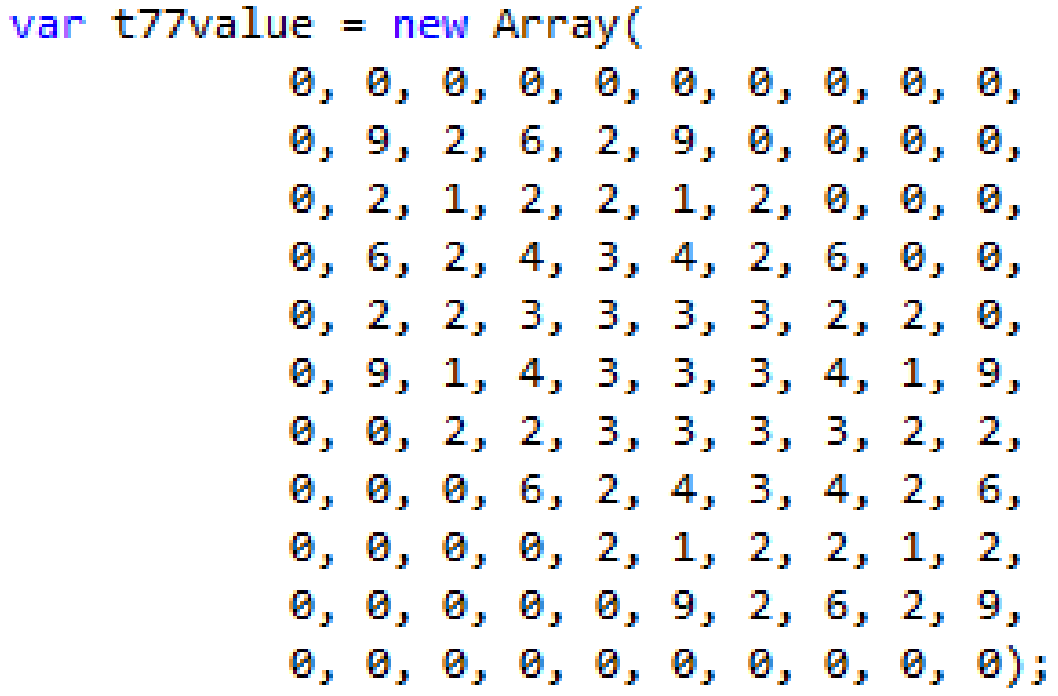
player X:	0
X pieces:	0
player Y:	0
Y pieces:	0
Winner is player:	
Draw:	0

Playing Algorithm

- Made use of 2 Heuristics
  - Mobility
  - Stability

Playing Algorithm- Stability

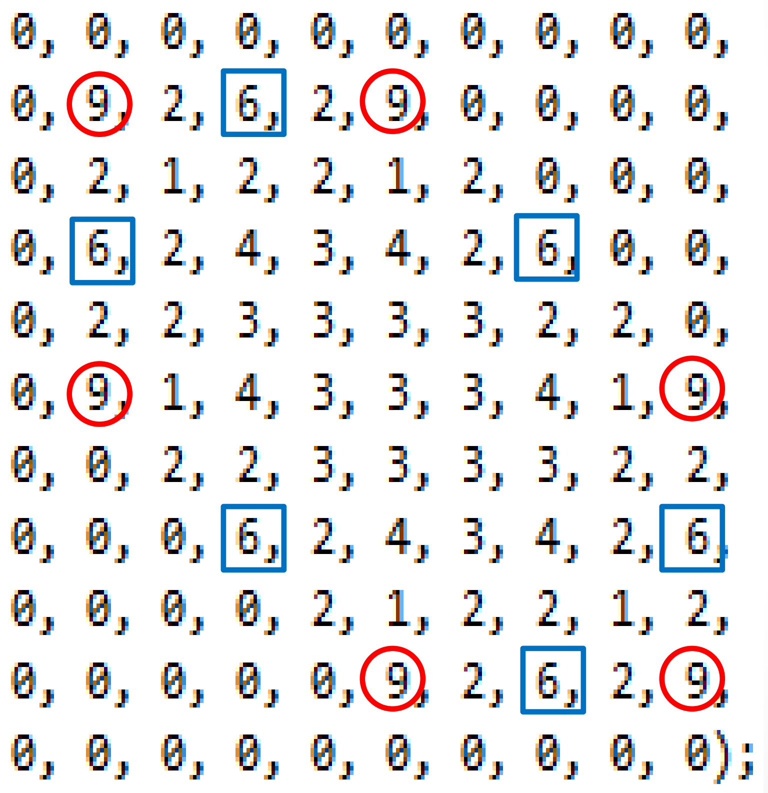
- Values assigned to each possible position of the piece on the board
- Code Fragment:



```
var t77value = new Array(
           0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
           0, 9, 2, 6, 2, 9, 0, 0, 0, 0,
           0, 2, 1, 2, 2, 1, 2, 0, 0, 0,
           0, 6, 2, 4, 3, 4, 2, 6, 0, 0,
           0, 2, 2, 3, 3, 3, 3, 2, 2, 0,
           0, 9, 1, 4, 3, 3, 3, 4, 1, 9,
           0, 0, 2, 2, 3, 3, 3, 3, 2, 2,
           0, 0, 0, 6, 2, 4, 3, 4, 2, 6,
           0, 0, 0, 0, 2, 1, 2, 2, 1, 2,
           0, 0, 0, 0, 0, 9, 2, 6, 2, 9,
           0, 0, 0, 0, 0, 0, 0, 0, 0, 0);
```

Playing Algorithm- Stability

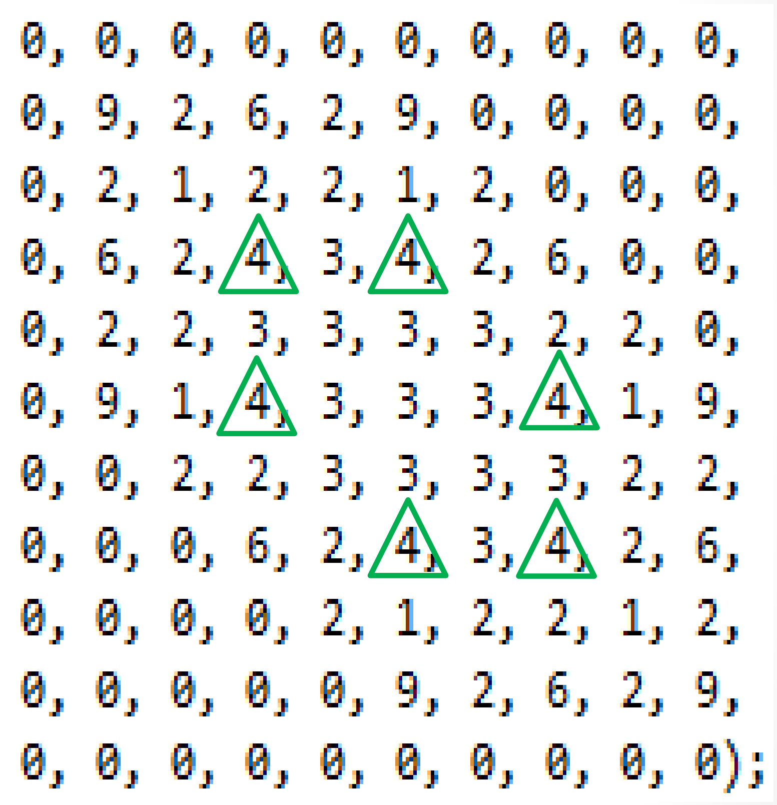
- Corners are of greatest importance.
  - No opposite-facing sides, capturable only by placing a piece on it.
  - No risk.
  - Value of 9
- Stepping stone to achieve corner.
  - Sides against the edge only have one opposite face.
  - Low risk.
  - Value of 6



```
0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 9, 2, 6, 2, 9, 0, 0, 0, 0,
0, 2, 1, 2, 2, 1, 2, 0, 0, 0,
0, 6, 2, 4, 3, 4, 2, 6, 0, 0,
0, 2, 2, 3, 3, 3, 3, 2, 2, 0,
0, 9, 1, 4, 3, 3, 3, 4, 1, 9,
0, 0, 2, 2, 3, 3, 3, 3, 2, 2,
0, 0, 0, 6, 2, 4, 3, 4, 2, 6,
0, 0, 0, 0, 2, 1, 2, 2, 1, 2,
0, 0, 0, 0, 0, 9, 2, 6, 2, 9,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0);
```

Playing Algorithm- Stability

- Stepping stone to achieve corner.
  - Non-edge position have more than one opposite face.
  - Easier to lose control if opponent dominants a stretch along the axis.
  - Medium risk.
  - Value of 4
- Positions between the corners and the middle of the edges are risky.
  - Counters the opponent's acquiring of a middle of an edge
  - No value in achieving the corner it is near to
  - Accelerates the opponents capturing of other vital positions if corner is taken
  - Value of 2



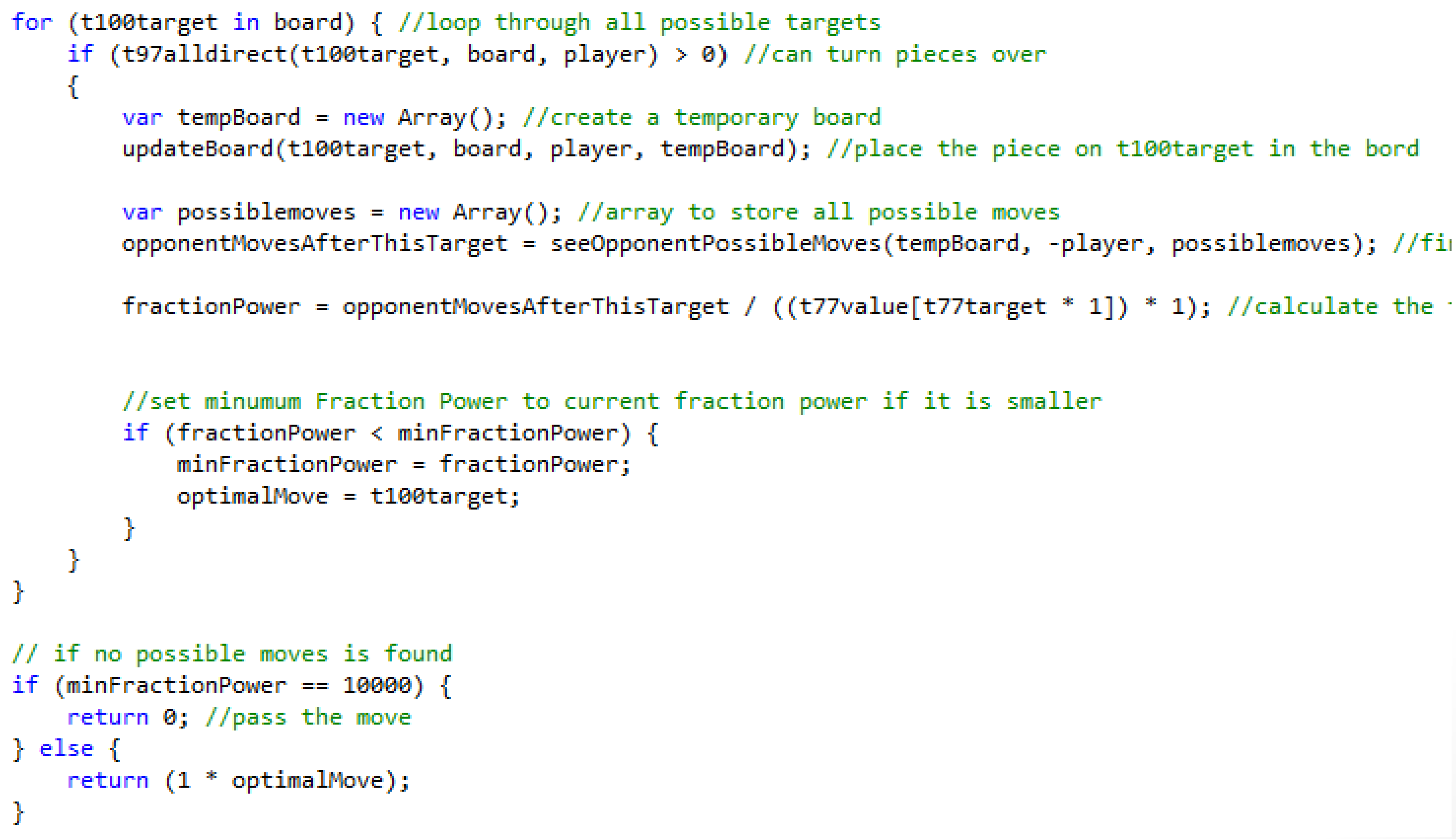
```
0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 9, 2, 6, 2, 9, 0, 0, 0, 0,
0, 2, 1, 2, 2, 1, 2, 0, 0, 0,
0, 6, 2, 4, 3, 4, 2, 6, 0, 0,
0, 2, 2, 3, 3, 3, 3, 2, 2, 0,
0, 9, 1, 4, 3, 3, 3, 4, 1, 9,
0, 0, 2, 2, 3, 3, 3, 3, 2, 2,
0, 0, 0, 6, 2, 4, 3, 4, 2, 6,
0, 0, 0, 0, 2, 1, 2, 2, 1, 2,
0, 0, 0, 0, 0, 9, 2, 6, 2, 9,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0);
```

Playing Algorithm - Mobility

- Minimize opponent moves available
  - Opponent either have to:
    - Pass (Bad move for opponent)
    - Choose sub-optimal move from limited move set

Playing Algorithm - Mobility

- Code Sample:



```
for (t100target in board) { //loop through all possible targets
    if (t97alldirect(t100target, board, player) > 0) //can turn pieces over
    {
        var tempBoard = new Array(); //create a temporary board
        updateBoard(t100target, board, player, tempBoard); //place the piece on t100target in the bord

        var possiblemoves = new Array(); //array to store all possible moves
        opponentMovesAfterThisTarget = seeOpponentPossibleMoves(tempBoard, -player, possiblemoves); //fi

        fractionPower = opponentMovesAfterThisTarget / ((t77value[t77target * 1]) * 1); //calculate the


        //set minumum Fraction Power to current fraction power if it is smaller
        if (fractionPower < minFractionPower) {
            minFractionPower = fractionPower;
            optimalMove = t100target;
        }
    }
}

// if no possible moves is found
if (minFractionPower == 10000) {
    return 0; //pass the move
} else {
    return (1 * optimalMove);
}
```