**MiniMaxOpening & MiniMaxOpeningBlack**

(The number of White or Black pieces on the board are not matter. Each of the program take control on putting either the white piece(MiniMaxOpening) or the black piece(MiniMaxOpeningBlack))

Example Input: xxWxxxxxxxxxxxxxxxxxx Give depth : 2 (meaning W place the piece first)

Result:

Opening Phase (with White&Black， D=2)

Table

Description automatically generated

**MiniMaxGame (White&Black)** -- bigger is better for white. Estimator (+ -) doesn't matter.

Input(from MiniMax Opening): BWWBWBWBWWBBBWBxxWxxx

In which MiniMaxGame(White piece) and MiniMaxGameBlack(Black piece)

Graphical user interface, text, application, table

Description automatically generated

**ABGame(Compare to MiniMaxGame**) -- supposed the same result but improved run time.

Given input: xWWBWBxBWWBxBWBxWxxWB

A picture containing text

Description automatically generated

**MiniMaxOpening & ABOpening**

(Using AB pruning to advanced the MiniMax, Output should be the same. )

Graphical user interface, text, application

Description automatically generated

**Improved ()**

(The basic idea is to improve the static estimation function. Instead the improved function supposed to calculate all the potential mills, for example the white piece could have. Then the function could be use to evaluate the new improved estimation could be:

1000\*(whiteCount-blackCount+potentialMillCount) – movesCount

The given condition for example can be when the game is not end yet, in which white or black piece is greater than 2, or the depth level is not 0

)

(The improvement program runs into some problems for defining the edges and the mills connected, and there is only the psedo-code for the MIniMaxOpeningImproved.)