My Heristic is based on the three heuristic examples that were given in the class: next is the code that was given:

```
def open_move_score(game, player):
    return float(len(game.get_legal_moves(player)))

def improved_score(game, player):

    own_moves = len(game.get_legal_moves(player))
    opp_moves = len(game.get_legal_moves(game.get_opponent(player)))
    return float(own_moves - opp_moves)

def center_score(game, player):

    w, h = game.width / 2., game.height / 2.
    y, x = game.get_player_location(player)
    return float((h - y)**2 + (w - x)**2)
```

So, the heuristics that I came out are based on these three measures. However, for doing a heuritic that takes into account those three main measurements, I need to have all three on the same scale. I found out the min and max values for each of the three score.Therefore I apply a factor scale to each one in such a way that i got foe each one a number that goes from 0 to 1. This implementation is as follows:

```
    min_open_move_score = 0.0
    min_improved_score  = -7.0
    min_center_distance = 0.5
    max_open_move_score = 7.0
    max_improved_score = 7.0
    max_center_distance = 40.5

    _open_move_score = open_move_score(game, player)
    _improved_score = improved_score(game, player)
    _center_distance = center_score(game, player)

    if _open_move_score < 0:
        _open_move_score
    elif _open_move_score > 7:
        _open_move_score = 7
    if _improved_score < -7:
        _improved_score = -7
    elif _improved_score > 7:
        _improved_score = 7
    if _center_distance < 0.5:
        _center_distance = 0.5
    elif _center_distance > 40.5:
        _center_distance = 40.5

    _open_move_score = get_equal_scale(_open_move_score, min_open_move_score,
max_open_move_score)
    _improved_score = get_equal_scale(_improved_score, min_improved_score,
max_improved_score)
    _center_distance = get_equal_scale(_center_distance, min_center_distance,
max_center_distance)
```

where the function “get_equal_scale” is as follows:

```
def get_equal_scale(xi, xmin, xmax):

    delta = abs(xmax - xmin)
    x = (xi - xmin) / delta
    return x
```

Now I have the three basic scores (“_open_move_score” , “_improved_score” and “_center_distance”) in the same scale therefore I can mix them in a better heurstic.

My first try was to use a score function that just adds these tree basic scores as follows:

```
score = (_open_move_score + _improved_score + _center_distance )
```

my intuition about it was that this addition will gave the option that fits best taking into account all three basic scores. However when I implemented it (and compared it with the other 2 heuristic that I will explain below) this heuristic was the one that performs worst.

Therefore I though I should get the distance norm of this vector (which is composed of three measurements).

The definition of this new heuristic is as follows:

```
score = ((_open_move_score) ** 2 + (_improved_score) ** 2 + (_center_distance) ** 2) ** (0.5)
```

my next step was to improve the last heuristic while adjustinc a factor to each of the three basic scores. This was a iterative process and I found out that if I reduce the “_center_distance” by 2, I was getting a better result and some times it was even better that AB_Improved. Tehrefore my heuristic 3 is as follows:

```
score = ((_open_move_score) ** 2 + (_improved_score) ** 2 + (_center_distance * 0.5) ** 2) ** (0.5)
```

below there is a output from a tournament I performed:

```
                         *************************
                              Playing Matches
                         *************************

Match #   Opponent    AB_Improved   AB_Custom   AB_Custom_2  AB_Custom_3
                       Won | Lost   Won | Lost   Won | Lost   Won | Lost
   1       Random      10  |   0    10  |   0    10  |   0     9  |   1
   2       MM_Open     10  |   0     8  |   2     7  |   3     9  |   1
   3      MM_Center     9  |   1    10  |   0    10  |   0    10  |   0
   4     MM_Improved    8  |   2     8  |   2     8  |   2     6  |   4
   5       AB_Open      8  |   2     6  |   4     4  |   6     2  |   8
   6      AB_Center     6  |   4     6  |   4     7  |   3     6  |   4
   7     AB_Improved    6  |   4     6  |   4     6  |   4     2  |   8
--------------------------------------------------------------------------
          Win Rate:      81.4%        77.1%        74.3%        62.9%
```