# Milestone 3: Implement Alpha Beta Pruning

This is an extension for milestone\_3.py where we implement alpha beta pruning.

#################################################################################

#

#                       Milestone 3 With Alpha Beta Pruning

#

#################################################################################

## How to run the program

Execute following command to run the program on a windows terminal

“python3 .\milestone\_3\_alpha\_beta.py”

The default depth is configured to be 3. However, to change the depth, please change “depth” variable value in milestone\_3 function in milestone\_3.py

## Output of program execution:

The output of the execution of the program at different depths are shown below

Text

Description automatically generated

Fig 1

Text

Description automatically generated

Fig 2

Text

Description automatically generated

Fig 3

Text

Description automatically generated

Fig 4

The score greatly increased as we increased the depth d. In fact, when depth is 9 we could get 2048 on board on the first run (Fig 4) although it took significant amount of time ~ 4.9 hours.

The runtime of the program greatly reduced when we introduced alpha-beta pruning. In the table below, we compare run times with and without alpha beta pruning:

|  |  |  |
| --- | --- | --- |
| **Depth** | **Without Pruning (seconds)** | **With Pruning (seconds)** |
| 3 | 3.97 | 1.60 |
| 5 | 275.96 | 63.04 |
| 7 | NA | 1255.45 |
| 9 | NA | 17659.48 |

Thus from the above table, alpha beta pruning method greatly reduced the run time of the program.

The outputs of the program without pruning algorithm can be found in Appendix A.

## Appendix A

Execution outputs of algorithm without alpha beta pruning method:

Text

Description automatically generated

Text

Description automatically generated