Name : Keerthana Thonupunuri

Milestone 3 Propose a Heuristic to improve performance of your code and/or reduce the running time of code

The greater number of empty spots on the board, the greater the chance of increasing our score as we have more space and can further the game. I started with this intuition, and backed by the research paper[https://theresamigler.files.wordpress.com/2020/03/2048.pdf], I created a heuristic function that would allow me to select the next best swipe direction if multiple swipe directions resulted in same score. This way, I increased my chances of obtaining a high score.

When swiping in 2 or more directions resulted in same score, we apply filters in following order to choose the next best move

1. Select the directions that result in highest number of empty spots (Filter 1)
2. If there are more than one after the above filter, select the directions that contain maximum number of non-decreasing or non-increasing rows or non-decreasing or non-increasing columns. (Filter 2)
3. Select one direction randomly from the directions left after applying filter 2.

How to run the program

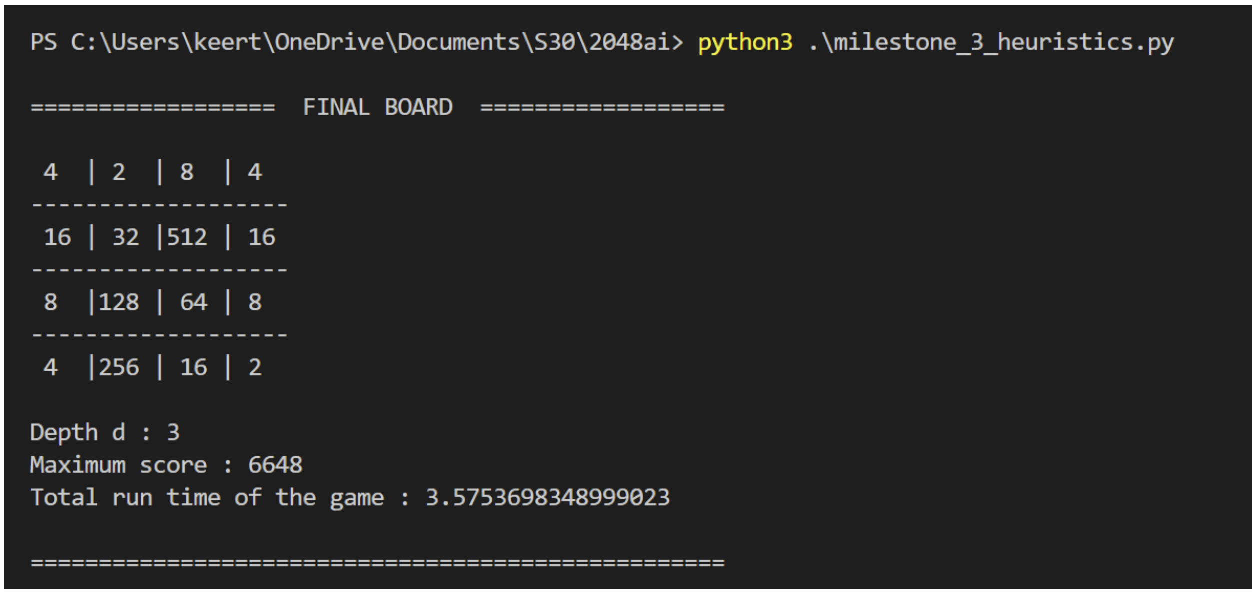
Execute following commands to run the programs on a windows terminal

```
“python3 .\milestone_3_heuristics.py”
```

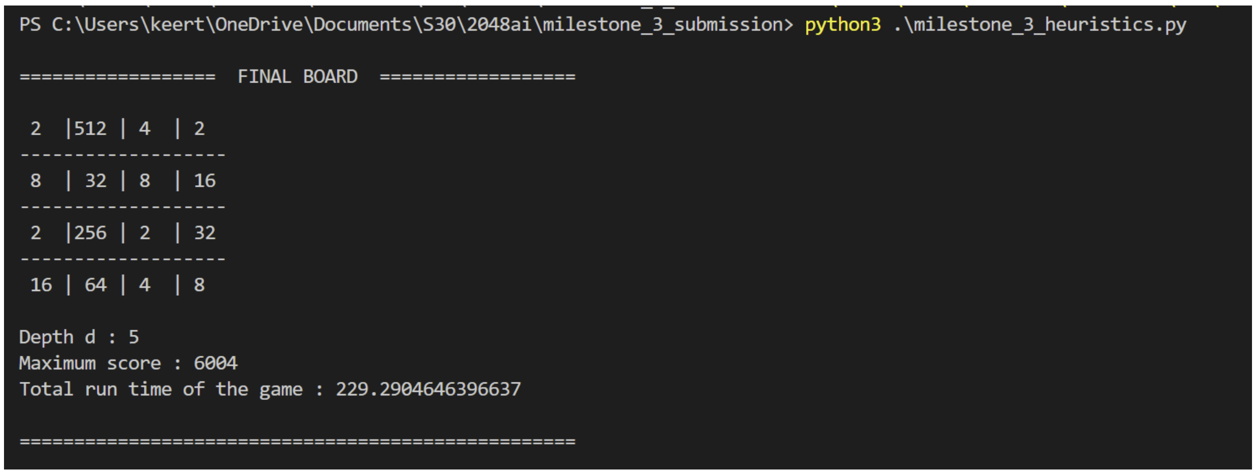
```
“python3 .\milestone_3_alpha_beta_heuristics.py”
```

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:

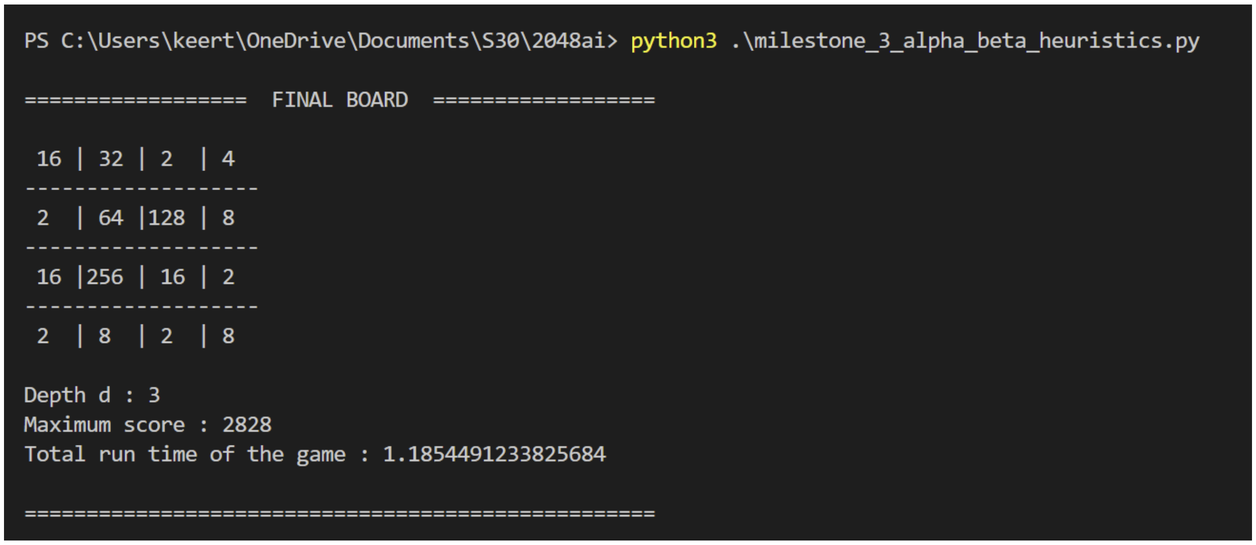


```
PS C:\Users\keert\OneDrive\Documents\S30\2048ai> python3 .\milestone_3_heuristics.py

==================  FINAL BOARD  ==================

 4  | 2  | 8  | 4
-------------------
 16 | 32 |512 | 16
-------------------
 8  |128 | 64 | 8
-------------------
 4  |256 | 16 | 2

Depth d : 3
Maximum score : 6648
Total run time of the game : 3.5753698348999023

===================================================
```

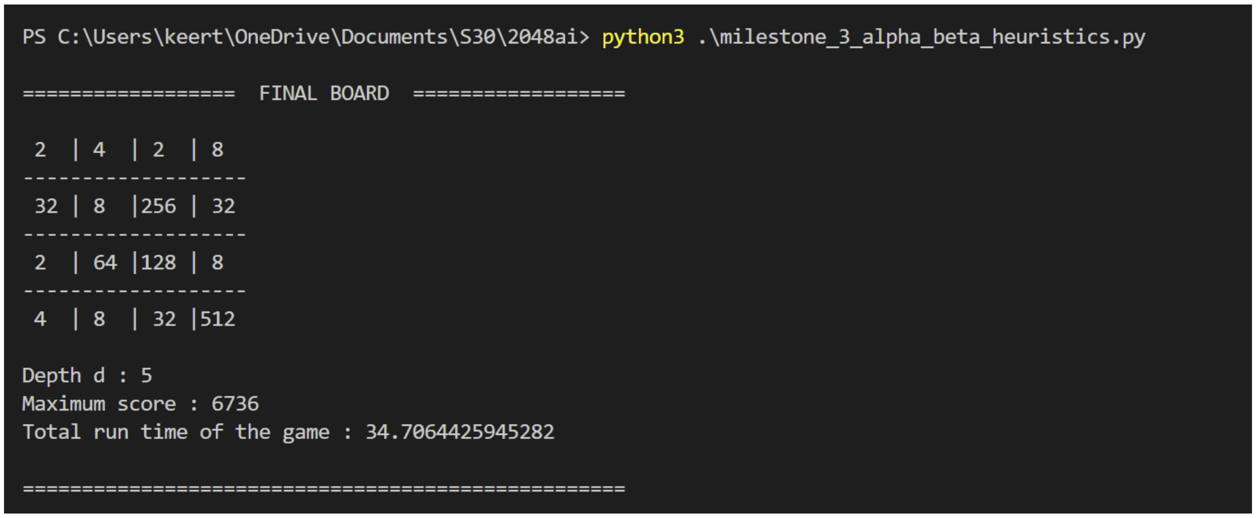


```
PS C:\Users\keert\OneDrive\Documents\S30\2048ai\milestone_3_submission> python3 .\milestone_3_heuristics.py

==================  FINAL BOARD  ==================

 2  |512 | 4  | 2
-------------------
 8  | 32 | 8  | 16
-------------------
 2  |256 | 2  | 32
-------------------
 16 | 64 | 4  | 8

Depth d : 5
Maximum score : 6004
Total run time of the game : 229.2904646396637

===================================================
```

I have executed heuristics with alpha beta pruning as well:



```
PS C:\Users\keert\OneDrive\Documents\S30\2048ai> python3 .\milestone_3_alpha_beta_heuristics.py

==================  FINAL BOARD  ==================

 16 | 32 | 2  | 4
-------------------
 2  | 64 |128 | 8
-------------------
 16 |256 | 16 | 2
-------------------
 2  | 8  | 2  | 8

Depth d : 3
Maximum score : 2828
Total run time of the game : 1.1854491233825684

===================================================
```



```
PS C:\Users\keert\OneDrive\Documents\S30\2048ai> python3 .\milestone_3_alpha_beta_heuristics.py

==================  FINAL BOARD  ==================

 2  | 4  | 2  | 8
-------------------
 32 | 8  |256 | 32
-------------------
 2  | 64 |128 | 8
-------------------
 4  | 8  | 32 |512

Depth d : 5
Maximum score : 6736
Total run time of the game : 34.7064425945282

===================================================
```

Although, we may not be able to prove the efficiency of the heuristic algorithm, if we could run the algorithm for around 100 times, we could clearly see the difference in distribution of high scores between algorithm without heuristic and with heuristic.