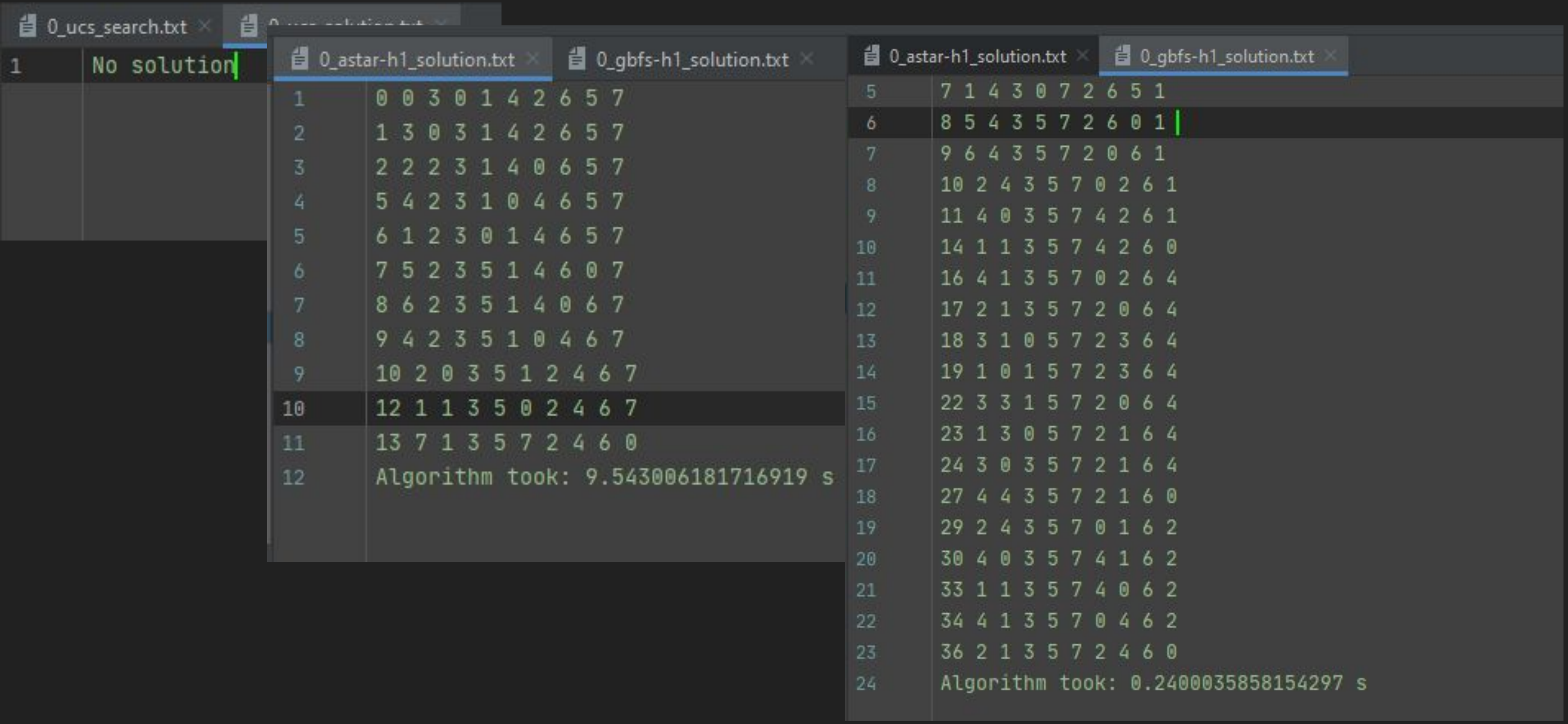


0_ucs_search.txt

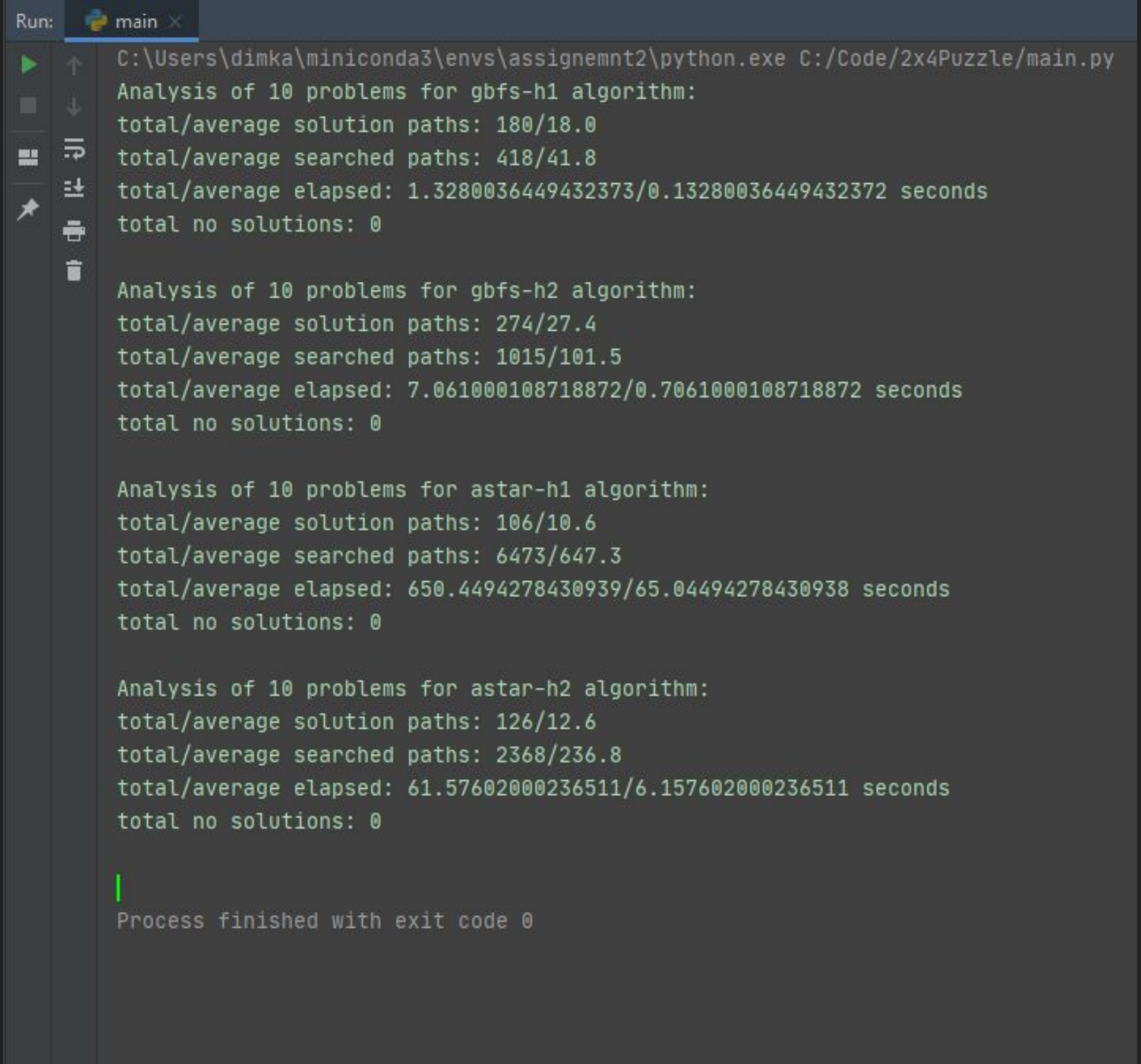
```
No solution
```

0_astar-h1_solution.txt    0_gbfs-h1_solution.txt

```
0 0 3 0 1 4 2 6 5 7
1 3 0 3 1 4 2 6 5 7
2 2 2 3 1 4 0 6 5 7
5 4 2 3 1 0 4 6 5 7
6 1 2 3 0 1 4 6 5 7
7 5 2 3 5 1 4 6 0 7
8 6 2 3 5 1 4 0 6 7
9 4 2 3 5 1 0 4 6 7
10 2 0 3 5 1 2 4 6 7
12 1 1 3 5 0 2 4 6 7
13 7 1 3 5 7 2 4 6 0
Algorithm took: 9.543006181716919 s
```

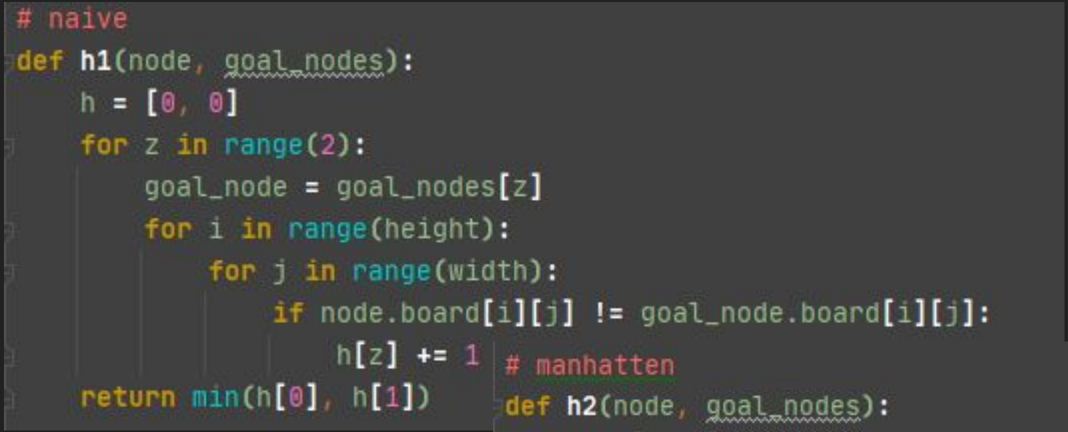
0_astar-h1_solution.txt    0_gbfs-h1_solution.txt

```
7 1 4 3 0 7 2 6 5 1
8 5 4 3 5 7 2 6 0 1
9 6 4 3 5 7 2 0 6 1
10 2 4 3 5 7 0 2 6 1
11 4 0 3 5 7 4 2 6 1
14 1 1 3 5 7 4 2 6 0
16 4 1 3 5 7 0 2 6 4
17 2 1 3 5 7 2 0 6 4
18 3 1 0 5 7 2 3 6 4
19 1 0 1 5 7 2 3 6 4
22 3 3 1 5 7 2 0 6 4
23 1 3 0 5 7 2 1 6 4
24 3 0 3 5 7 2 1 6 4
27 4 4 3 5 7 2 1 6 0
29 2 4 3 5 7 0 1 6 2
30 4 0 3 5 7 4 1 6 2
33 1 1 3 5 7 4 0 6 2
34 4 1 3 5 7 0 4 6 2
36 2 1 3 5 7 2 4 6 0
Algorithm took: 0.2400035858154297 s
```

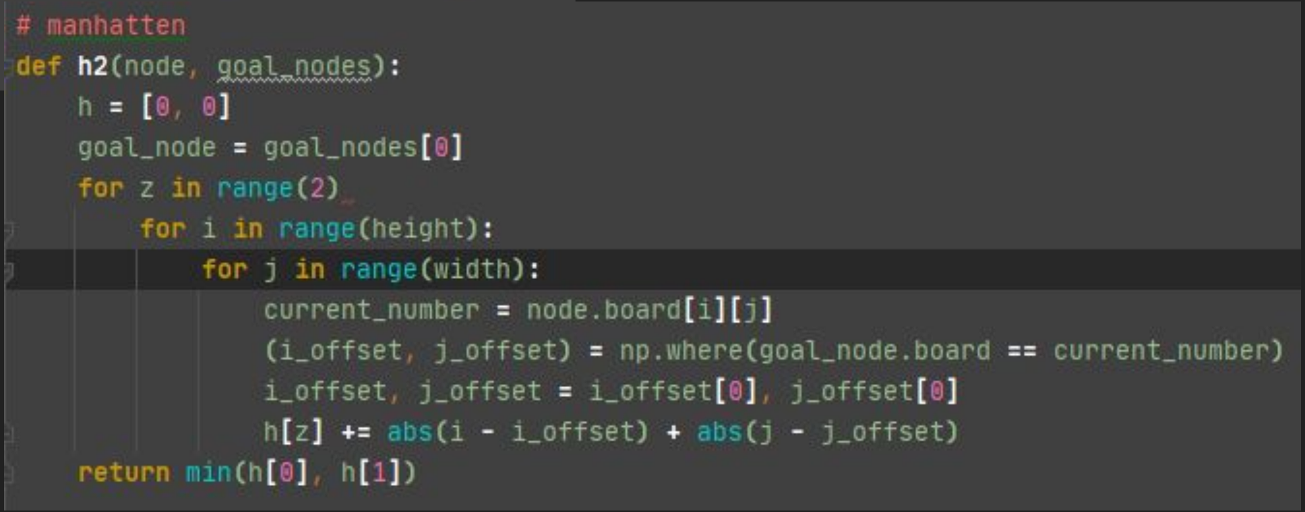


```
C:\Users\dimka\miniconda3\envs\assignemnt2\python.exe C:/Code/2x4Puzzle/main.py
Analysis of 10 problems for gbfs-h1 algorithm:
total/average solution paths: 180/18.0
total/average searched paths: 418/41.8
total/average elapsed: 1.3280036449432373/0.13280036449432372 seconds
total no solutions: 0

Analysis of 10 problems for gbfs-h2 algorithm:
total/average solution paths: 274/27.4
total/average searched paths: 1015/101.5
total/average elapsed: 7.061000108718872/0.7061000108718872 seconds
total no solutions: 0

Analysis of 10 problems for astar-h1 algorithm:
total/average solution paths: 106/10.6
total/average searched paths: 6473/647.3
total/average elapsed: 650.4494278430939/65.04494278430938 seconds
total no solutions: 0

Analysis of 10 problems for astar-h2 algorithm:
total/average solution paths: 126/12.6
total/average searched paths: 2368/236.8
total/average elapsed: 61.57602000236511/6.157602000236511 seconds
total no solutions: 0


Process finished with exit code 0
```

## Heuristics:

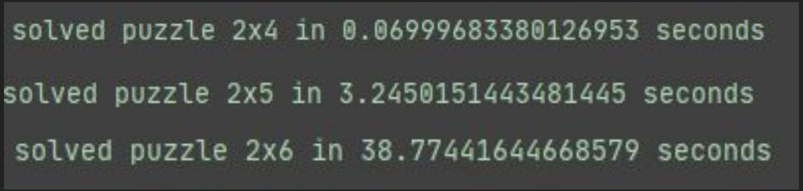


```
# naive
def h1(node, goal_nodes):
    h = [0, 0]
    for z in range(2):
        goal_node = goal_nodes[z]
        for i in range(height):
            for j in range(width):
                if node.board[i][j] != goal_node.board[i][j]:
                    h[z] += 1
    return min(h[0], h[1])
```

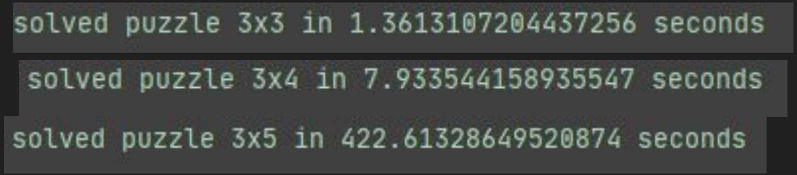


```
# manhatten
def h2(node, goal_nodes):
    h = [0, 0]
    goal_node = goal_nodes[0]
    for z in range(2):
        for i in range(height):
            for j in range(width):
                current_number = node.board[i][j]
                (i_offset, j_offset) = np.where(goal_node.board == current_number)
                i_offset, j_offset = i_offset[0], j_offset[0]
                h[z] += abs(i - i_offset) + abs(j - j_offset)
    return min(h[0], h[1])
```

## GBFS h1 was the fastest, scale up trial



```
solved puzzle 2x4 in 0.06999683380126953 seconds
solved puzzle 2x5 in 3.2450151443481445 seconds
solved puzzle 2x6 in 38.77441644668579 seconds
```



```
solved puzzle 3x3 in 1.3613107204437256 seconds
solved puzzle 3x4 in 7.933544158935547 seconds
solved puzzle 3x5 in 422.61328649520874 seconds
```