

OUTPUTS FOR MINESWEEPER GAME

*If the user starts the program, the board will be randomly generated.

- The board size $N \times N$ will be randomly chosen where $2 \leq N \leq 10$.
- Some cells will be randomly designated as mines. (In this version, the number of mines is equal to the board size N . For example, if N is 4 then there are 4 mines)
- The generated board will be saved to a file named "map.txt".

*If the user enters a move in the form of two integers (row and column), the program will process the move.

- If the move hits a mine, the game ends with the message "BOOM! You hit a mine. Game Over."
- If the cell has zero neighboring mines, a recursive flood fill will uncover adjacent cells.
- The current board will be displayed after each move using ASCII characters:
 - # for hidden cells
 - 0-8 for revealed cells showing nearby mine count
 - X for a revealed mine

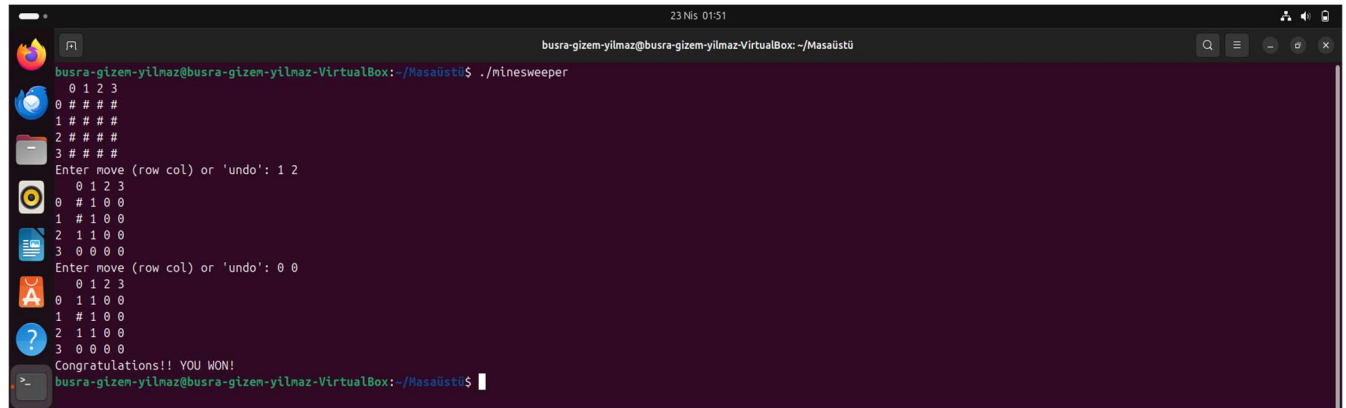
*If the user enters 'undo', the most recent move will be removed from the board.

- A custom stack (using two arrays for row and column values) will be used to keep track of moves.
- The undo command will remove the most recent move from this stack.

*If the game ends (win or loss), all valid moves will be saved to a file named "moves.txt".

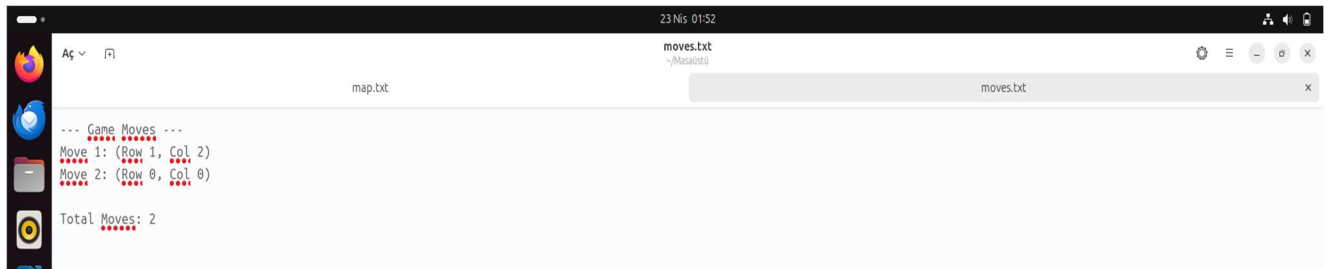
- The file will contain the history of all moves made during the game, formatted as specified.

EXAMPLE 1

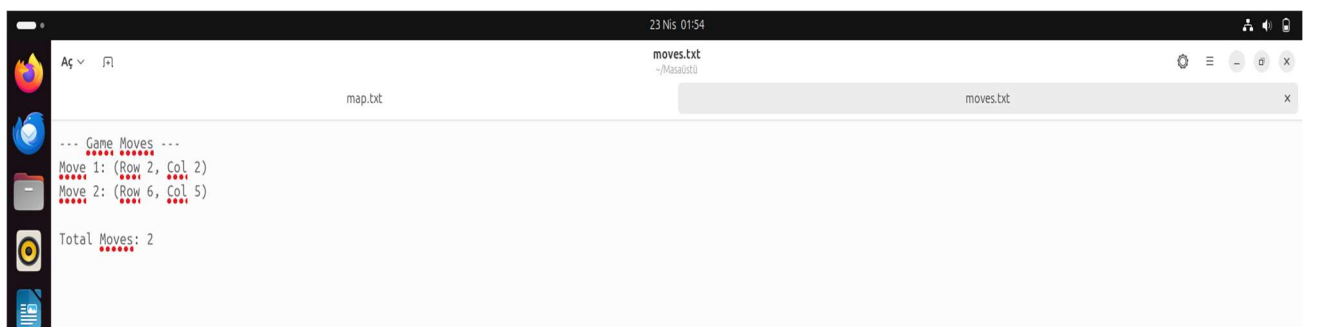
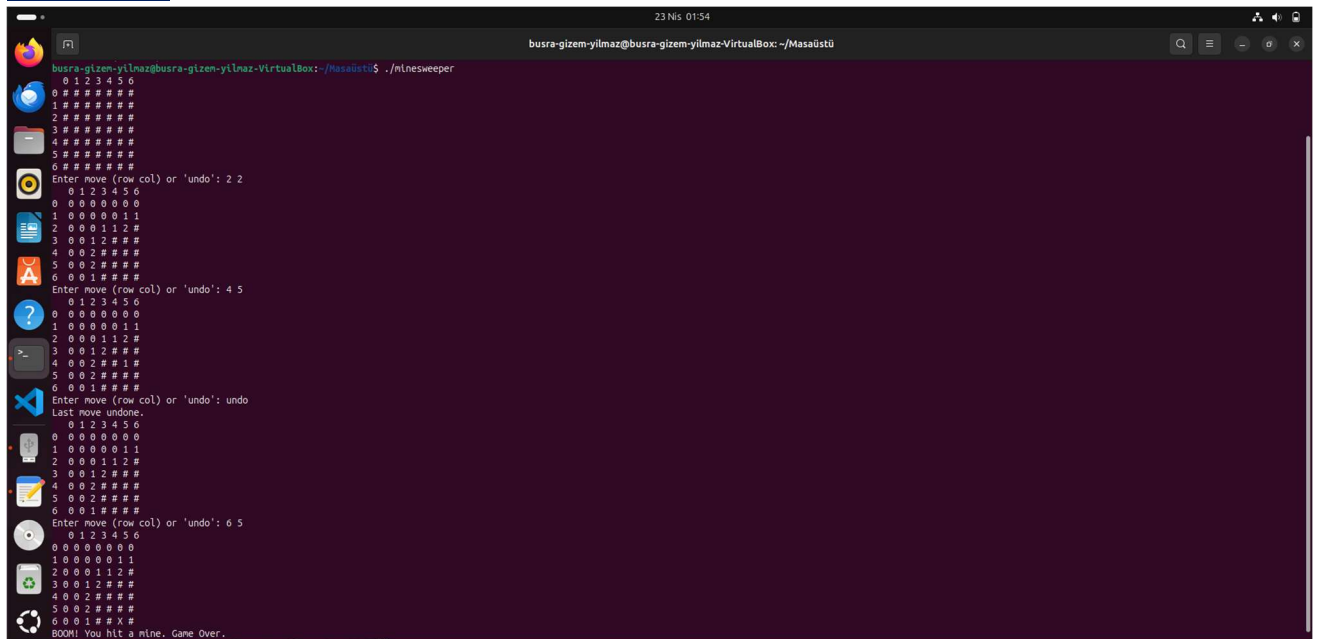


```
busra-gizen-yilmaz@busra-gizen-yilmaz-VirtualBox: ~/Masaüstü
busra-gizen-yilmaz@busra-gizen-yilmaz-VirtualBox: ~/Masaüstü$ ./minesweeper
0 1 2 3
0 # # # #
1 # # # #
2 # # # #
3 # # # #
Enter move (row col) or 'undo': 1 2
0 1 2 3
0 # 1 0 0
1 # 1 0 0
2 1 1 0 0
3 0 0 0 0
Enter move (row col) or 'undo': 0 0
0 1 2 3
0 1 1 0 0
1 # 1 0 0
2 1 1 0 0
3 0 0 0 0
Congratulations!! YOU WON!
busra-gizen-yilmaz@busra-gizen-yilmaz-VirtualBox: ~/Masaüstü$
```

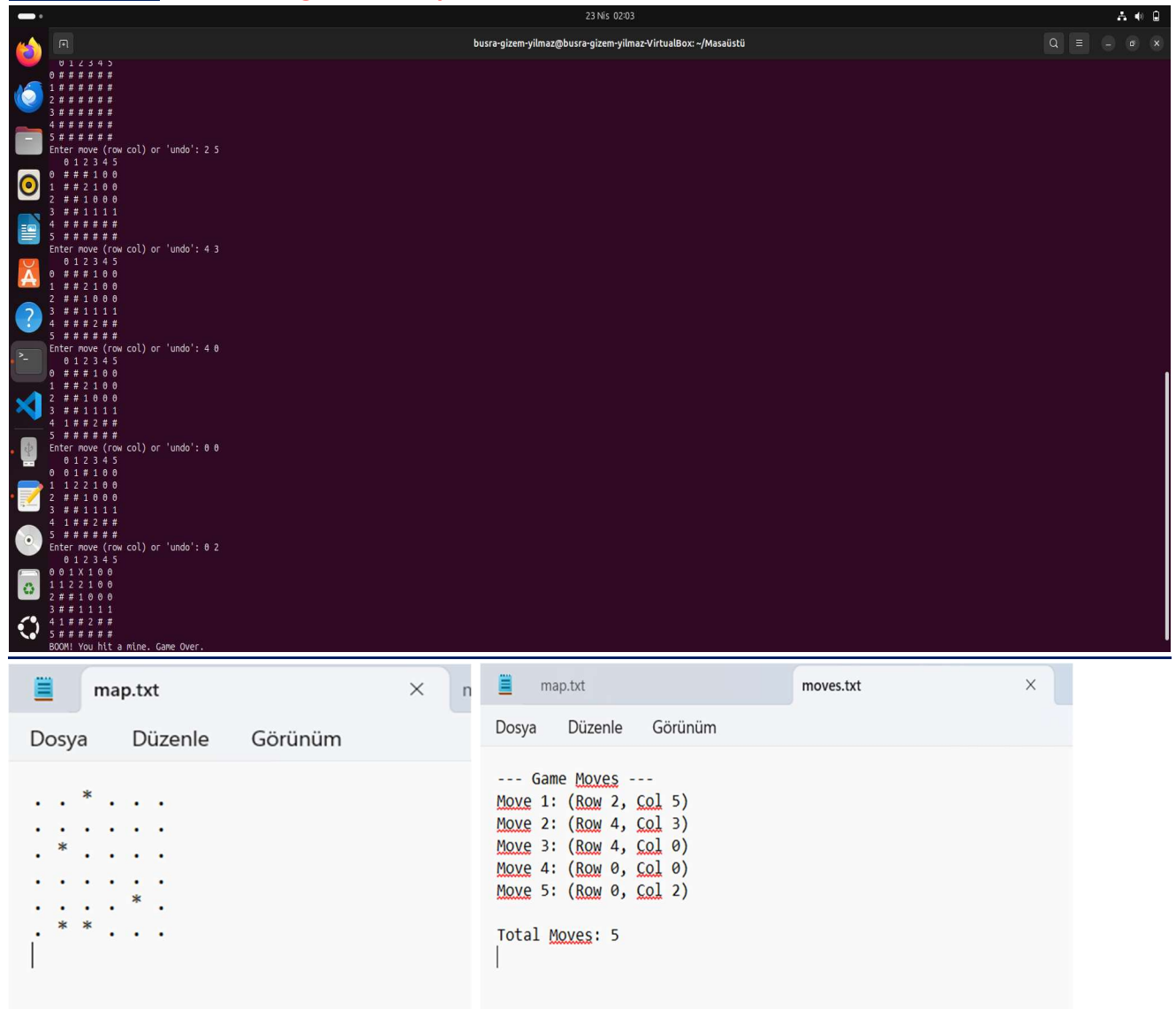




EXAMPLE 2



EXAMPLE 3 (I sent this game's map.txt and moves.txt)



The screenshot shows a terminal window with a Minesweeper game session. The game board is a 6x6 grid. The initial state is as follows:

```
0 1 2 3 4 5
0 #####
1 #####
2 #####
3 #####
4 #####
5 #####
```

The user enters moves: (row col) or 'undo': 2 5, 4 3, 4 0, 0 0, 0 2. The game board updates accordingly, revealing numbers and mines. The final state is as follows:

```
0 1 2 3 4 5
0 0 1 X 1 0 0
1 1 2 2 1 0 0
2 # 1 0 0 0
3 # 1 1 1 1
4 1 # 2 # #
5 #####
```

The terminal also shows the following messages:

```
Enter move (row col) or 'undo': 2 5
Enter move (row col) or 'undo': 4 3
Enter move (row col) or 'undo': 4 0
Enter move (row col) or 'undo': 0 0
Enter move (row col) or 'undo': 0 2
BOOM! You hit a mine. Game Over.
```

Below the terminal, two text files are shown:

map.txt

```

. . * . . .
. . . . .
. * . . .
. . . . .
. . . * .
. * * . .
```

moves.txt

```
--- Game Moves ---
Move 1: (Row 2, Col 5)
Move 2: (Row 4, Col 3)
Move 3: (Row 4, Col 0)
Move 4: (Row 0, Col 0)
Move 5: (Row 0, Col 2)

Total Moves: 5
```

SOME ERRORS ABOUT MINESWEEPER GAME

- *If the user enters a value that is not two integers or the word 'undo', the program will display an "Invalid value. Try again." error message and prompt for input again.
- *If the user enters coordinates that are outside the bounds of the board (greater than or equal to the board size), the program will display an "Invalid move. Try again." error message and prompt for input again.
- *If the user enters 'undo' as the first move (when the stack is empty), the program will display a error message and prompt for input again. The undo command can only be used after at least one valid move has been made.
- *If the user enters the same move more than once (cell that has already been revealed), the program will display an error message and prompt for input again.

```
busra-gizen-yilmaz@busra-gizen-yilmaz-VirtualBox:~/Masaüstü$ ./minesweeper
  0 1 2 3 4
0 #####
1 #####
2 #####
3 #####
4 #####
Enter move (row col) or 'undo': 1 7
Invalid move! Try again.
Enter move (row col) or 'undo': 9 8
Invalid move! Try again.
Enter move (row col) or 'undo': 1 a
Invalid value! Try again.
Enter move (row col) or 'undo': . 3
Invalid value! Try again.
Enter move (row col) or 'undo': undoo
Invalid value! Try again.
Enter move (row col) or 'undo': abcd
Invalid value! Try again.
Enter move (row col) or 'undo': 0 4
  0 1 2 3 4
0 #####1
1 #####
2 #####
3 #####
4 #####
Enter move (row col) or 'undo': █
```

```
busra-gizen-yilmaz@busra-gizen-yilmaz-VirtualBox:~/Masaüstü$ ./minesweeper
  0 1 2 3 4 5
0 #####
1 #####
2 #####
3 #####
4 #####
5 #####
Enter move (row col) or 'undo': undo
You cannot undo because the stack is empty.
Enter move (row col) or 'undo': 2 5
  0 1 2 3 4 5
0 #####
1 #####
2 #####1
3 #####
4 #####
5 #####
Enter move (row col) or 'undo': undo
Last move undone.
  0 1 2 3 4 5
0 #####
1 #####
2 #####
3 #####
4 #####
5 #####
Enter move (row col) or 'undo':
```

```
23 Nis 02:11
busra-gizen-yilmaz@busra-gizen-yilmaz-VirtualBox: ~/Masaüstü
busra-gizen-yilmaz@busra-gizen-yilmaz-VirtualBox:~/Masaüstü$ ./minesweeper
  0 1 2 3 4 5 6 7 8 9
0 #####
1 #####
2 #####
3 #####
4 #####
5 #####
6 #####
7 #####
8 #####
9 #####
Enter move (row col) or 'undo': 1 3
  0 1 2 3 4 5 6 7 8 9
0 #####
1 ##1#####
2 #####
3 #####
4 #####
5 #####
6 #####
7 #####
8 #####
9 #####
Enter move (row col) or 'undo': 1 3
You already tried this cell. Try a different move.
Enter move (row col) or 'undo': 1 3
You already tried this cell. Try a different move.
Enter move (row col) or 'undo': 2 4
  0 1 2 3 4 5 6 7 8 9
0 ##1001###
1 1211001121
2 0000000000
3 0000000000
4 0111000000
5 01#1011100
6 13#201#100
7 ##1011100
8 ##2110011
```