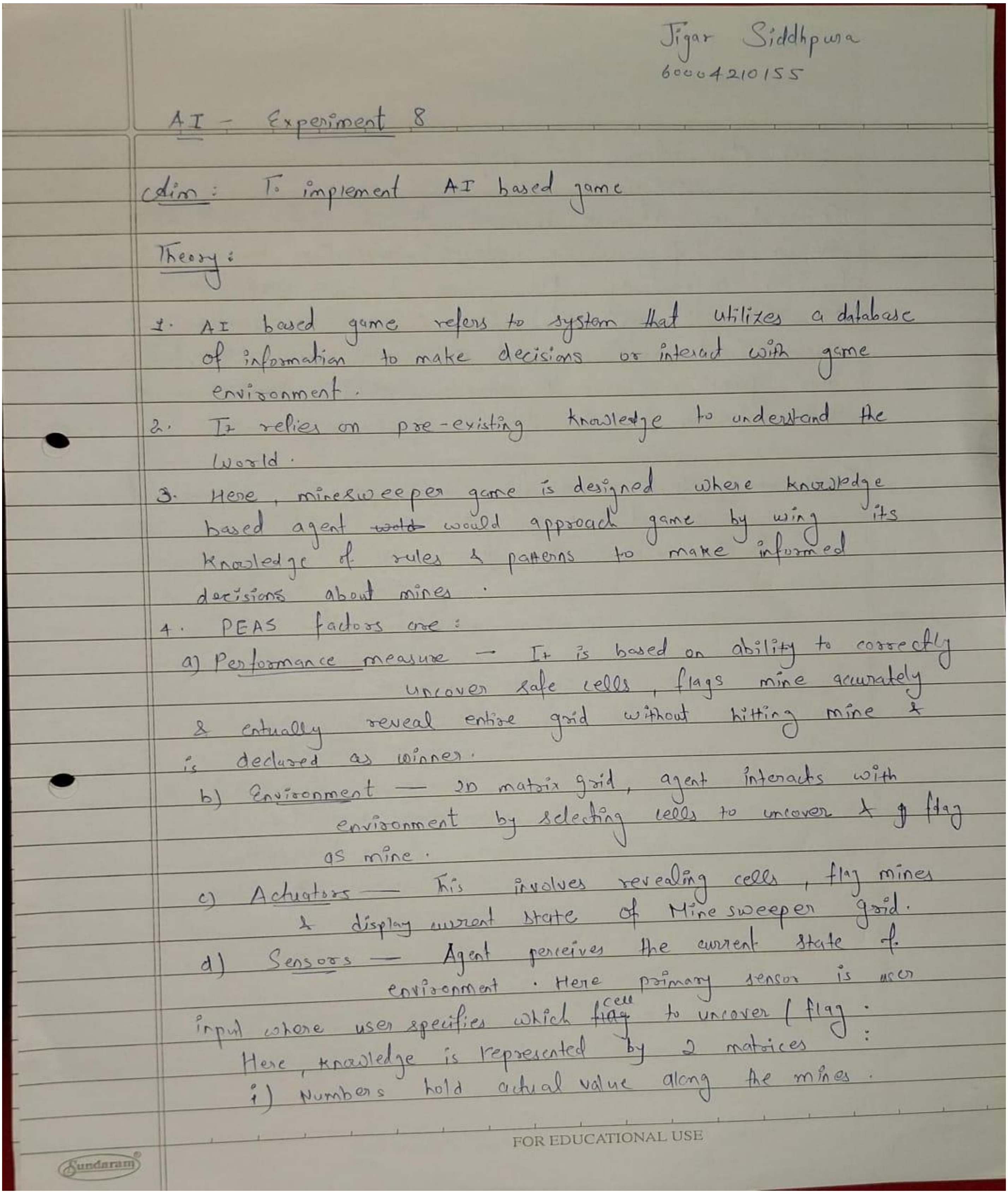
Name: Jigar Siddhpura **SAPID:** 60004200155

DIV: C/C2 **Branch:** Computer Engineering

AI EXPERIMENT 8



Jigar Siddhpura
60004210155

AI - Experiment 8

Aim : To implement AI based game

Theory :

1. AI based game refers to system that utilizes a database of information to make decisions or interact with game environment.
2. It relies on pre-existing knowledge to understand the world.
3. Here, minesweeper game is designed where knowledge based agent ~~wold~~ would approach game by using its knowledge of rules & patterns to make informed decisions about mines.
4. PEAS factors are :

a) Performance measure — It is based on ability to correctly uncover safe cells, flags mine accurately & entually reveal entire grid without hitting mine & is declared as winner.

b) Environment — 2D matrix grid, agent interacts with environment by selecting cells to uncover & g flag as mine.

c) Actuators — This involves revealing cells, flag mines & display current state of Minesweeper grid.

d) Sensors — Agent perceives the current state of environment. Here primary sensor is user input where user specifies which ~~flag~~ cell to uncover / flag :

Here, knowledge is represented by 2 matrices :

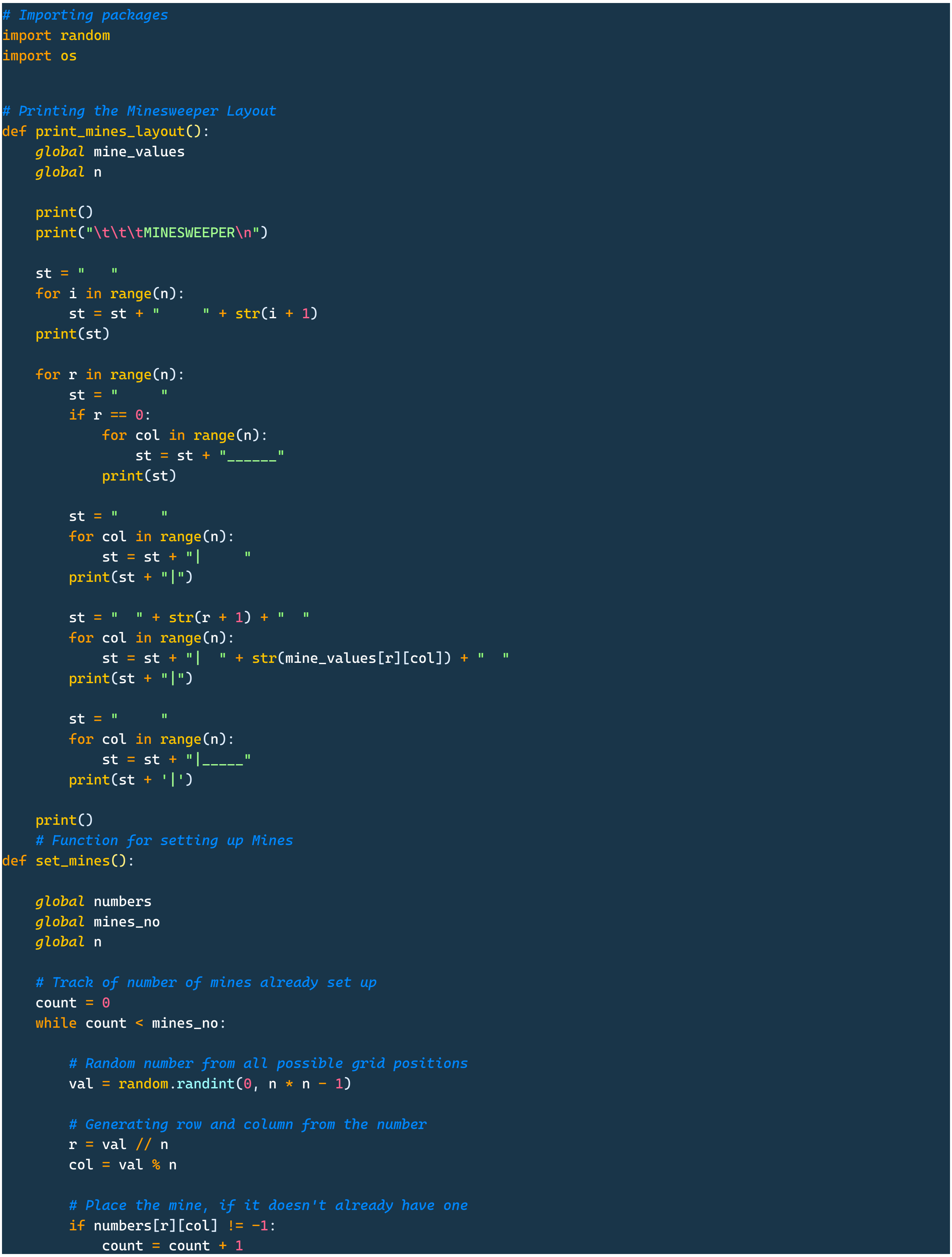
i) Numbers hold actual value along the mines.

ii) Mine-value what player sees.

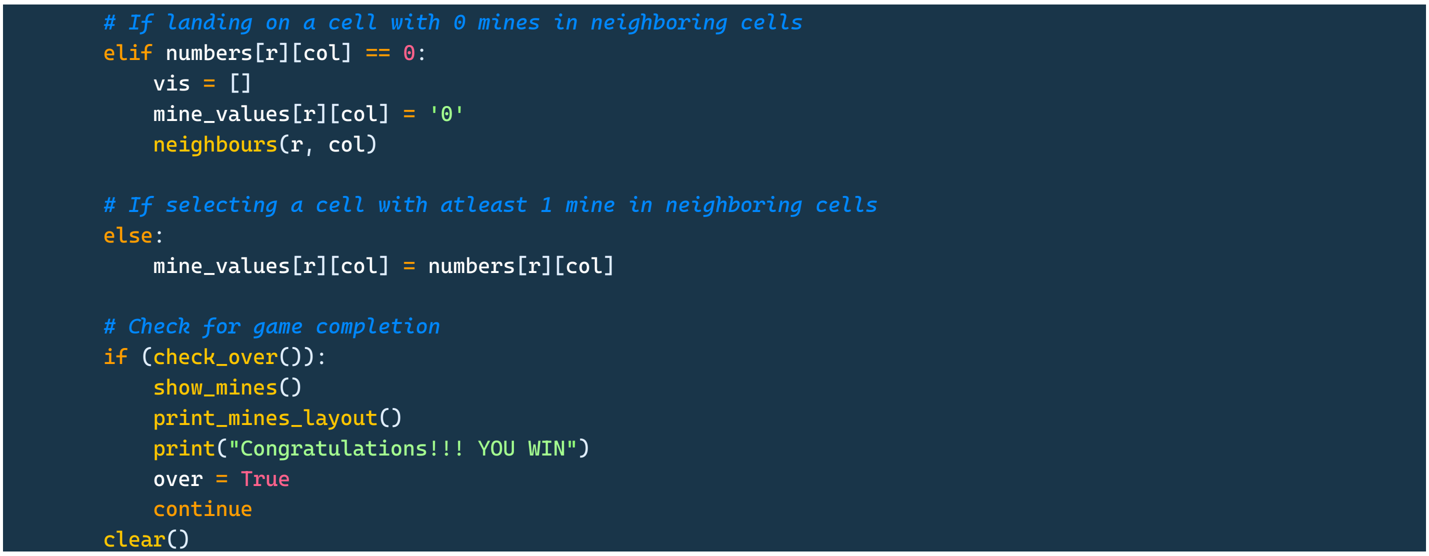
The agent uses inference to make decision about which cell to ~~or~~ uncover/flag. It uses set-mines() & set-values() function to randomly place mines & calc. the value of non-mine cell. neighbors() func is used to uncover all zero-valued neighbors when cell with zero is selected. This game is similar to wumpus world wherein agent uses inference to make decision.

Conclusion: Hence, code for minesweeper was implemented & we understood how AI based agent game works & how it makes use of knowledge based agents.

Code :

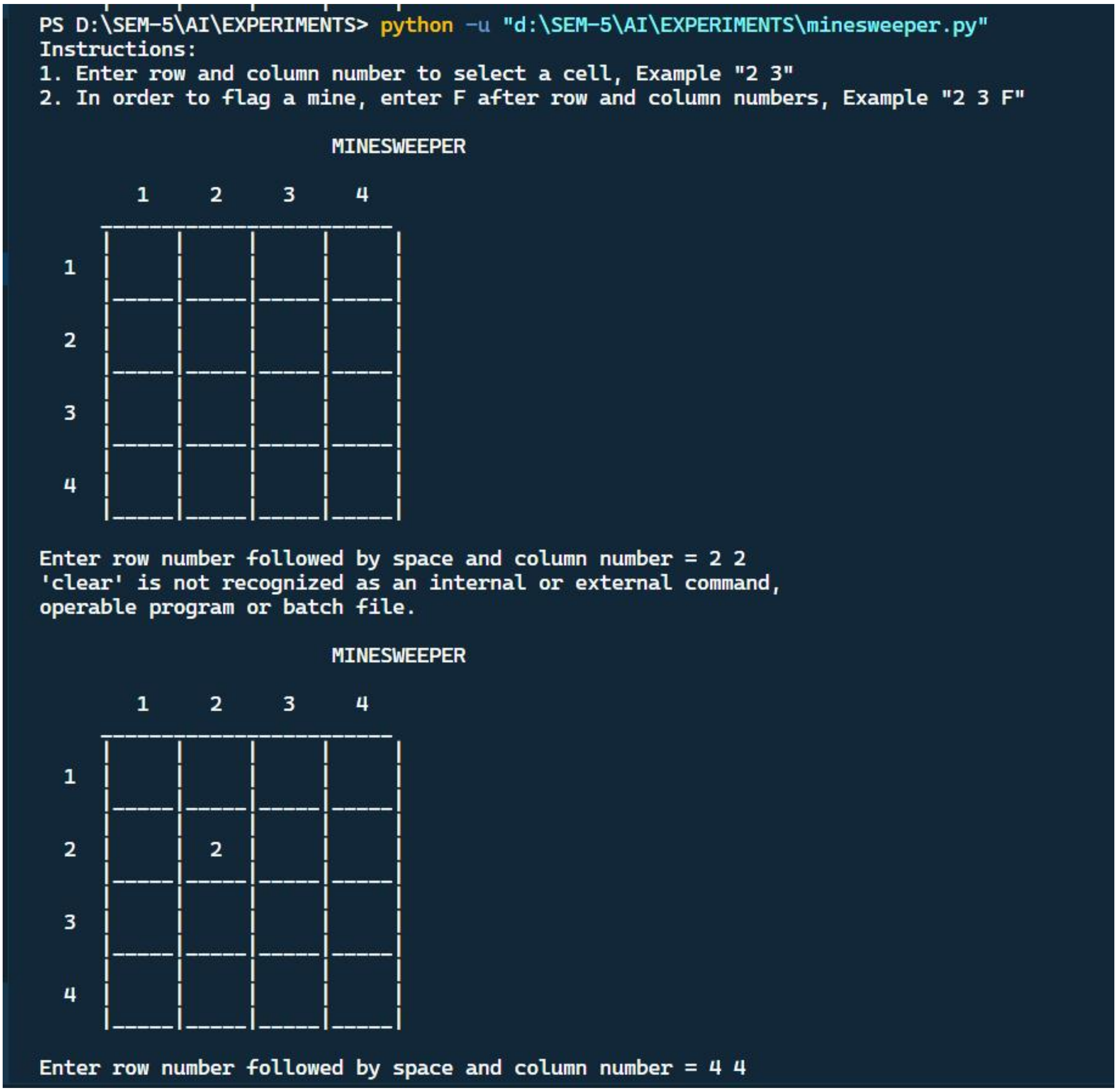


```python
# Importing packages
import random
import os


# Printing the Minesweeper Layout
def print_mines_layout():
    global mine_values
    global n

    print()
    print("\t\t\tMINESWEEPER\n")

    st = "   "
    for i in range(n):
        st = st + "     " + str(i + 1)
    print(st)

    for r in range(n):
        st = "     "
        if r == 0:
            for col in range(n):
                st = st + "______"
            print(st)

        st = "     "
        for col in range(n):
            st = st + "|     "
        print(st + "|")

        st = "  " + str(r + 1) + "  "
        for col in range(n):
            st = st + "|  " + str(mine_values[r][col]) + "  "
        print(st + "|")

        st = "     "
        for col in range(n):
            st = st + "|_____"
        print(st + '|')

    print()
    # Function for setting up Mines
def set_mines():

    global numbers
    global mines_no
    global n

    # Track of number of mines already set up
    count = 0
    while count < mines_no:

        # Random number from all possible grid positions
        val = random.randint(0, n * n - 1)

        # Generating row and column from the number
        r = val // n
        col = val % n

        # Place the mine, if it doesn't already have one
        if numbers[r][col] != -1:
            count = count + 1
            numbers[r][col] = -1

# Function for setting up the other grid values
def set_values():

    global numbers
    global n

    # Loop for counting each cell value
    for r in range(n):
        for col in range(n):

            # Skip, if it contains a mine
            if numbers[r][col] == -1:
                Continue

            # Check up
            if r > 0 and numbers[r - 1][col] == -1:
                numbers[r][col] = numbers[r][col] + 1
            # Check down
            if r < n - 1 and numbers[r + 1][col] == -1:
                numbers[r][col] = numbers[r][col] + 1
            # Check left
            if col > 0 and numbers[r][col - 1] == -1:
                numbers[r][col] = numbers[r][col] + 1
            # Check right
            if col < n - 1 and numbers[r][col + 1] == -1:
                numbers[r][col] = numbers[r][col] + 1
            # Check top-left
            if r > 0 and col > 0 and numbers[r - 1][col - 1] == -1:
                numbers[r][col] = numbers[r][col] + 1
            # Check top-right
            if r > 0 and col < n - 1 and numbers[r - 1][col + 1] == -1:
                numbers[r][col] = numbers[r][col] + 1
            # Check below-left
            if r < n - 1 and col > 0 and numbers[r + 1][col - 1] == -1:
                numbers[r][col] = numbers[r][col] + 1
            # Check below-right
            if r < n - 1 and col < n - 1 and numbers[r + 1][col + 1] == -1:
                numbers[r][col] = numbers[r][col] + 1

# Recursive function to display all zero-valued neighbours
def neighbours(r, col):

    global mine_values
    global numbers
    global vis

    # If the cell already not visited
    if [r, col] not in vis:

        # Mark the cell visited
        vis.append([r, col])

        # If the cell is zero-valued
        if numbers[r][col] == 0:

            # Display it to the user
            mine_values[r][col] = numbers[r][col]

            # Recursive calls for the neighbouring cells
            if r > 0:
                neighbours(r - 1, col)
            if r < n - 1:
                neighbours(r + 1, col)
            if col > 0:
                neighbours(r, col - 1)
            if col < n - 1:
                neighbours(r, col + 1)
            if r > 0 and col > 0:
                neighbours(r - 1, col - 1)
            if r > 0 and col < n - 1:
                neighbours(r - 1, col + 1)
            if r < n - 1 and col > 0:
                neighbours(r + 1, col - 1)
            if r < n - 1 and col < n - 1:
                neighbours(r + 1, col + 1)

        # If the cell is not zero-value
        if numbers[r][col] != 0:
            mine_values[r][col] = numbers[r][col]

# Function for clearing the terminal
def clear():
    os.system("clear")

# Function to display the instructions
def instructions():
    print("Instructions:")
    print("1. Enter row and column number to select a cell, Example \"2 3\"")
    print(
        "2. In order to flag a mine, enter F after row and column numbers, Example \"2 3 F\""
    )

# Function to check for completion of the game
def check_over():
    global mine_values
    global n
    global mines_no

    # Count of all numbered values
    count = 0

    # Loop for checking each cell in the grid
    for r in range(n):
        for col in range(n):

            # If cell not empty or flagged
            if mine_values[r][col] != ' ' and mine_values[r][col] != 'F':
                count = count + 1

    # Count comparison
    if count == n * n - mines_no:
        return True
    else:
        return False

# Display all the mine locations
def show_mines():
    global mine_values
    global numbers
    global n

    for r in range(n):
        for col in range(n):
            if numbers[r][col] == -1:
                mine_values[r][col] = 'M'

if __name__ == "__main__":
    # Size of grid
    n = 8
    # Number of mines
    mines_no = 8

    # The actual values of the grid
    numbers = [[0 for y in range(n)] for x in range(n)]
    # The apparent values of the grid
    mine_values = [[' ' for y in range(n)] for x in range(n)]
    # The positions that have been flagged
    flags = []

    # Set the mines
    set_mines()

    # Set the values
    set_values()

    # Display the instructions
    instructions()

    # Variable for maintaining Game Loop
    over = False

    # The GAME LOOP
    while not over:
        print_mines_layout()

        # Input from the user
        inp = input(
            "Enter row number followed by space and column number = ").split()

        # Standard input
        if len(inp) == 2:

            # Try block to handle errant input
            try:
                val = list(map(int, inp))
            except ValueError:
                clear()
                print("Wrong input!")
                instructions()
                Continue

        # Flag input
        elif len(inp) == 3:
            if inp[2] != 'F' and inp[2] != 'f':
                clear()
                print("Wrong Input!")
                instructions()
                Continue

            # Try block to handle errant input
            try:
                val = list(map(int, inp[:2]))
            except ValueError:
                clear()
                print("Wrong input!")
                instructions()
                Continue

            # Sanity checks
            if val[0] > n or val[0] < 1 or val[1] > n or val[1] < 1:
                clear()
                print("Wrong input!")
                instructions()
                Continue

            # Get row and column numbers
            r = val[0] - 1
            col = val[1] - 1

            # If cell already been flagged
            if [r, col] in flags:
                clear()
                print("Flag already set")
                Continue

            # If cell already been displayed
            if mine_values[r][col] != ' ':
                clear()
                print("Value already known")
                Continue

            # Check the number for flags
            if len(flags) < mines_no:
                clear()
                print("Flag set")

                # Adding flag to the list
                flags.append([r, col])

                # Set the flag for display
                mine_values[r][col] = 'F'
                continue
            else:
                clear()
                print("Flags finished")
                Continue

        else:
            clear()
            print("Wrong input!")
            instructions()
            Continue

        # Sanity checks
        if val[0] > n or val[0] < 1 or val[1] > n or val[1] < 1:
            clear()
            print("Wrong Input!")
            instructions()
            Continue

        # Get row and column number
        r = val[0] - 1
        col = val[1] - 1

        # Unflag the cell if already flagged
        if [r, col] in flags:
            flags.remove([r, col])

        # If landing on a mine --- GAME OVER
        if numbers[r][col] == -1:
            mine_values[r][col] = 'M'
            show_mines()
            print_mines_layout()
            print("Landed on a mine. GAME OVER!!!!!")
            over = True
            continue
```

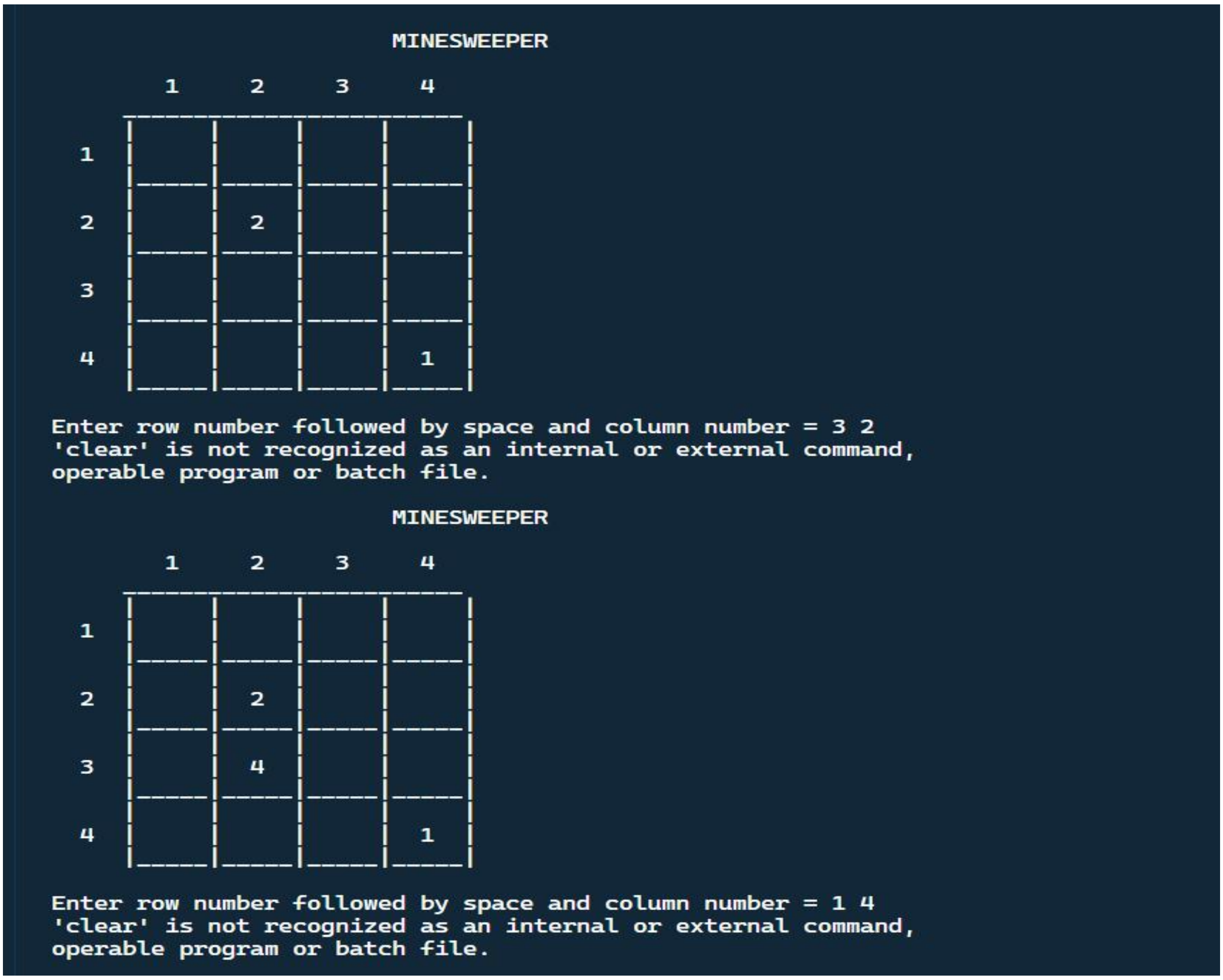


```
        # If landing on a cell with 0 mines in neighboring cells
        elif numbers[r][col] == 0:
            vis = []
            mine_values[r][col] = '0'
            neighbours(r, col)

        # If selecting a cell with atleast 1 mine in neighboring cells
        else:
            mine_values[r][col] = numbers[r][col]

        # Check for game completion
        if (check_over()):
            show_mines()
            print_mines_layout()
            print("Congratulations!!! YOU WIN")
            over = True
            continue
        clear()
```

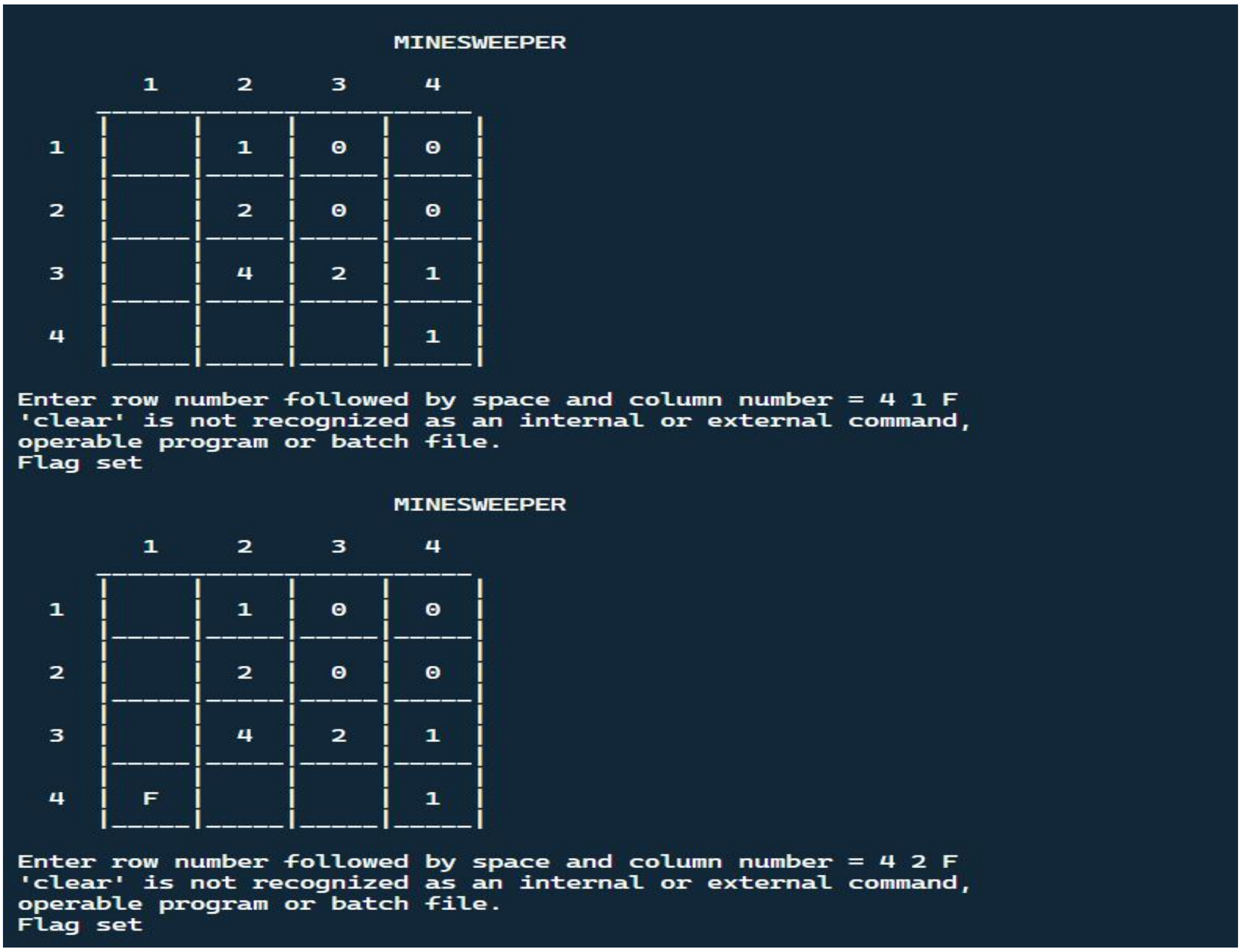
Output :



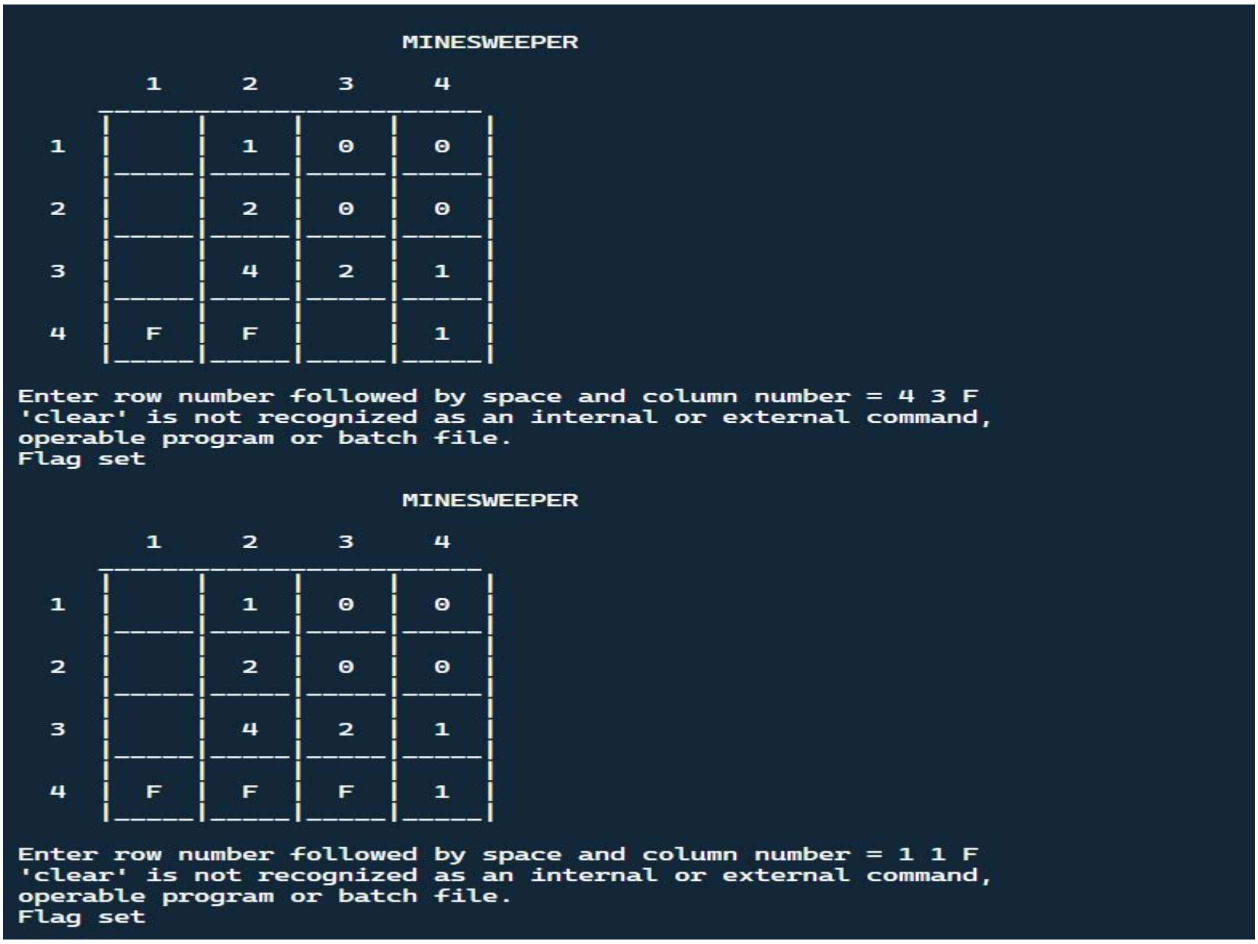
```
PS D:\SEM-5\AI\EXPERIMENTS> python -u "d:\SEM-5\AI\EXPERIMENTS\minesweeper.py"
Instructions:
1. Enter row and column number to select a cell, Example "2 3"
2. In order to flag a mine, enter F after row and column numbers, Example "2 3 F"

                        MINESWEEPER

        1     2     3     4
     ________________________
     |     |     |     |     |
  1  |     |     |     |     |
     |_____|_____|_____|_____|
     |     |     |     |     |
  2  |     |     |     |     |
     |_____|_____|_____|_____|
     |     |     |     |     |
  3  |     |     |     |     |
     |_____|_____|_____|_____|
     |     |     |     |     |
  4  |     |     |     |     |
     |_____|_____|_____|_____|

Enter row number followed by space and column number = 2 2
'clear' is not recognized as an internal or external command,
operable program or batch file.

                        MINESWEEPER

        1     2     3     4
     ________________________
     |     |     |     |     |
  1  |     |     |     |     |
     |_____|_____|_____|_____|
     |     |     |     |     |
  2  |     |  2  |     |     |
     |_____|_____|_____|_____|
     |     |     |     |     |
  3  |     |     |     |     |
     |_____|_____|_____|_____|
     |     |     |     |     |
  4  |     |     |     |     |
     |_____|_____|_____|_____|

Enter row number followed by space and column number = 4 4
```



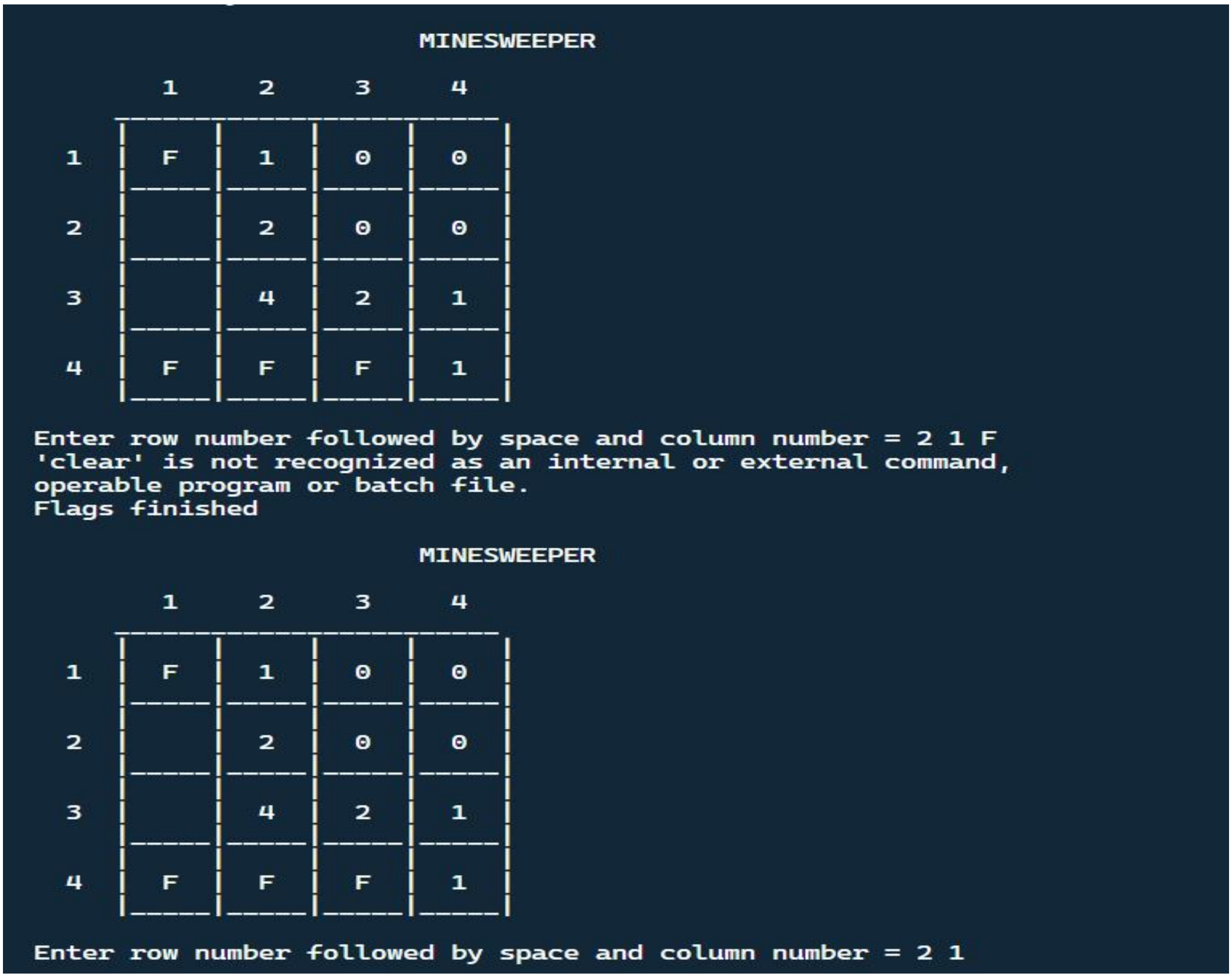
```
                           MINESWEEPER

         1     2     3     4
       _______________________
      |     |     |     |     |
  1   |     |     |     |     |
      |_____|_____|_____|_____|
      |     |     |     |     |
  2   |     |  2  |     |     |
      |_____|_____|_____|_____|
      |     |     |     |     |
  3   |     |     |     |     |
      |_____|_____|_____|_____|
      |     |     |     |     |
  4   |     |     |     |  1  |
      |_____|_____|_____|_____|

   Enter row number followed by space and column number = 3 2
   'clear' is not recognized as an internal or external command,
   operable program or batch file.

                           MINESWEEPER

         1     2     3     4
       _______________________
      |     |     |     |     |
  1   |     |     |     |     |
      |_____|_____|_____|_____|
      |     |     |     |     |
  2   |     |  2  |     |     |
      |_____|_____|_____|_____|
      |     |     |     |     |
  3   |     |  4  |     |     |
      |_____|_____|_____|_____|
      |     |     |     |     |
  4   |     |     |     |  1  |
      |_____|_____|_____|_____|

   Enter row number followed by space and column number = 1 4
   'clear' is not recognized as an internal or external command,
   operable program or batch file.
```



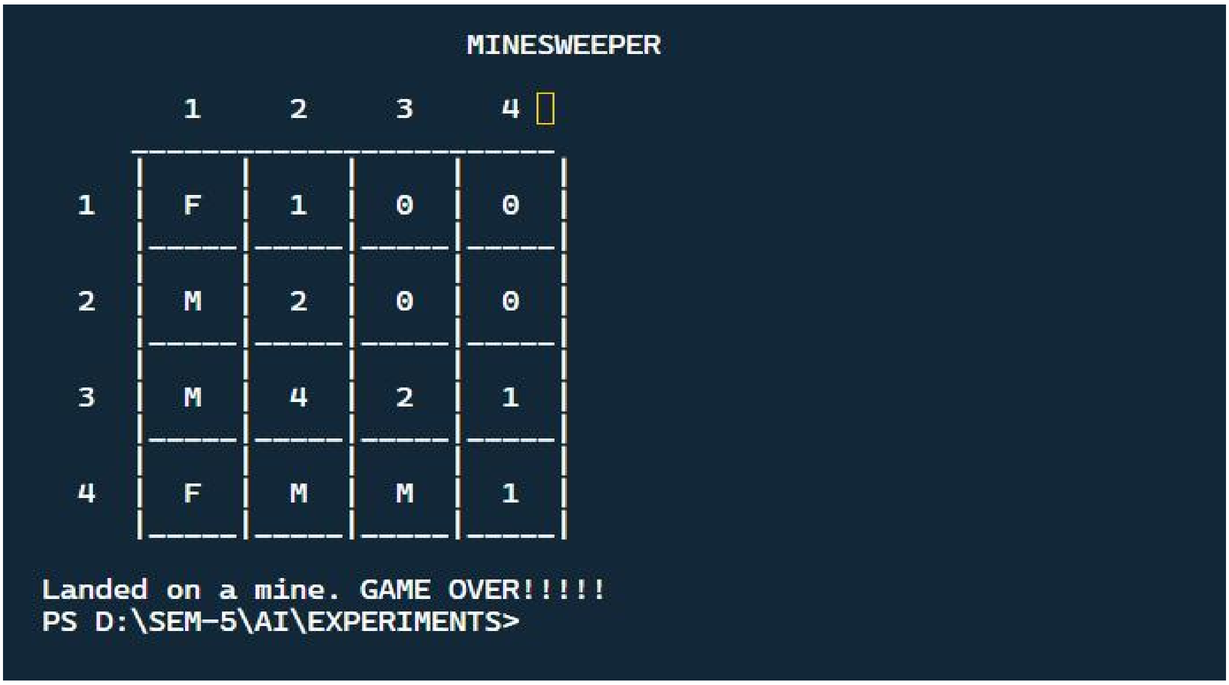
```
                           MINESWEEPER

         1      2      3      4
      ___________________________
     |      |      |      |      |
  1  |      |  1   |  0   |  0   |
     |______|______|______|______|
     |      |      |      |      |
  2  |      |  2   |  0   |  0   |
     |______|______|______|______|
     |      |      |      |      |
  3  |      |  4   |  2   |  1   |
     |______|______|______|______|
     |      |      |      |      |
  4  |      |      |      |  1   |
     |______|______|______|______|

Enter row number followed by space and column number = 4 1 F
'clear' is not recognized as an internal or external command,
operable program or batch file.
Flag set
                           MINESWEEPER

         1      2      3      4
      ___________________________
     |      |      |      |      |
  1  |      |  1   |  0   |  0   |
     |______|______|______|______|
     |      |      |      |      |
  2  |      |  2   |  0   |  0   |
     |______|______|______|______|
     |      |      |      |      |
  3  |      |  4   |  2   |  1   |
     |______|______|______|______|
     |      |      |      |      |
  4  |  F   |      |      |  1   |
     |______|______|______|______|

Enter row number followed by space and column number = 4 2 F
'clear' is not recognized as an internal or external command,
operable program or batch file.
Flag set
```



```
                                MINESWEEPER

          1       2       3       4
      _________________________________
     |       |       |       |       |
 1   |       |   1   |   0   |   0   |
     |_______|_______|_______|_______|
     |       |       |       |       |
 2   |       |   2   |   0   |   0   |
     |_______|_______|_______|_______|
     |       |       |       |       |
 3   |       |   4   |   2   |   1   |
     |_______|_______|_______|_______|
     |       |       |       |       |
 4   |   F   |   F   |       |   1   |
     |_______|_______|_______|_______|

Enter row number followed by space and column number = 4 3 F
'clear' is not recognized as an internal or external command,
operable program or batch file.
Flag set
                                MINESWEEPER

          1       2       3       4
      _________________________________
     |       |       |       |       |
 1   |       |   1   |   0   |   0   |
     |_______|_______|_______|_______|
     |       |       |       |       |
 2   |       |   2   |   0   |   0   |
     |_______|_______|_______|_______|
     |       |       |       |       |
 3   |       |   4   |   2   |   1   |
     |_______|_______|_______|_______|
     |       |       |       |       |
 4   |   F   |   F   |   F   |   1   |
     |_______|_______|_______|_______|

Enter row number followed by space and column number = 1 1 F
'clear' is not recognized as an internal or external command,
operable program or batch file.
Flag set
```



```
                                MINESWEEPER

          1       2       3       4
      _________________________________
     |       |       |       |       |
 1   |   F   |   1   |   0   |   0   |
     |_______|_______|_______|_______|
     |       |       |       |       |
 2   |       |   2   |   0   |   0   |
     |_______|_______|_______|_______|
     |       |       |       |       |
 3   |       |   4   |   2   |   1   |
     |_______|_______|_______|_______|
     |       |       |       |       |
 4   |   F   |   F   |   F   |   1   |
     |_______|_______|_______|_______|

Enter row number followed by space and column number = 2 1 F
'clear' is not recognized as an internal or external command,
operable program or batch file.
Flags finished
                                MINESWEEPER

          1       2       3       4
      _________________________________
     |       |       |       |       |
 1   |   F   |   1   |   0   |   0   |
     |_______|_______|_______|_______|
     |       |       |       |       |
 2   |       |   2   |   0   |   0   |
     |_______|_______|_______|_______|
     |       |       |       |       |
 3   |       |   4   |   2   |   1   |
     |_______|_______|_______|_______|
     |       |       |       |       |
 4   |   F   |   F   |   F   |   1   |
     |_______|_______|_______|_______|

Enter row number followed by space and column number = 2 1
```



```
                          MINESWEEPER

          1     2     3     4
       _______________________
       |     |     |     |     |
   1   |  F  |  1  |  0  |  0  |
       |_____|_____|_____|_____|
       |     |     |     |     |
   2   |  M  |  2  |  0  |  0  |
       |_____|_____|_____|_____|
       |     |     |     |     |
   3   |  M  |  4  |  2  |  1  |
       |_____|_____|_____|_____|
       |     |     |     |     |
   4   |  F  |  M  |  M  |  1  |
       |_____|_____|_____|_____|

  Landed on a mine. GAME OVER!!!!!
  PS D:\SEM-5\AI\EXPERIMENTS>
```