DMS PROJECT

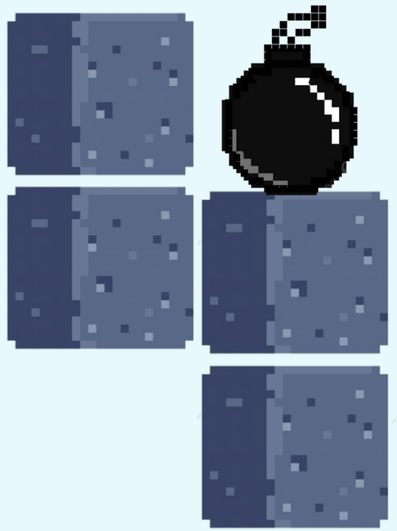
THE ULTIMATE MINESWEEPER GAME

START

SUBMITTED BY:
AREEBA TARIQ & PARKASH

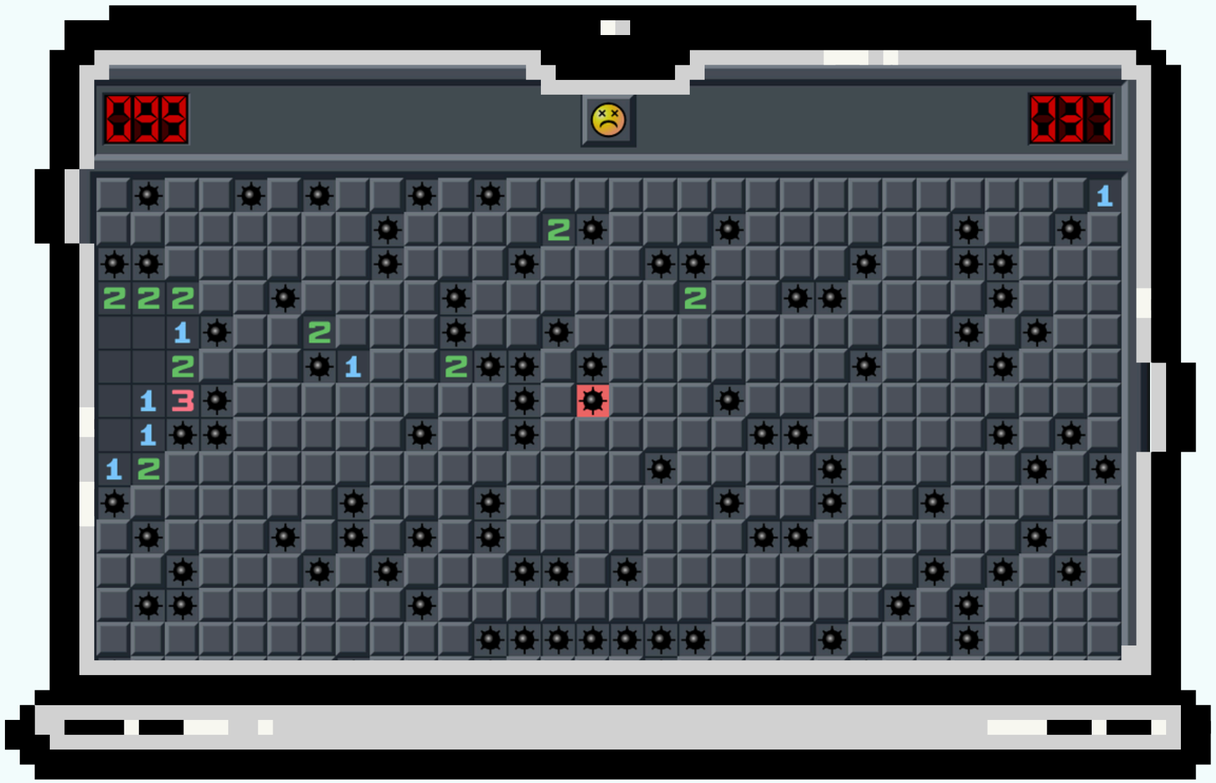
TABLE OF CONTENTS

INTRODUCTION



Minesweeper is a well-known single-player puzzle game where the objective is to clear a grid of squares without detonating any hidden mines. The player is provided with a grid of cells, some of which contain mines, while others indicate the number of adjacent mines. The game involves strategic thinking, where the player must make decisions based on the numbers revealed and avoid triggering mines.

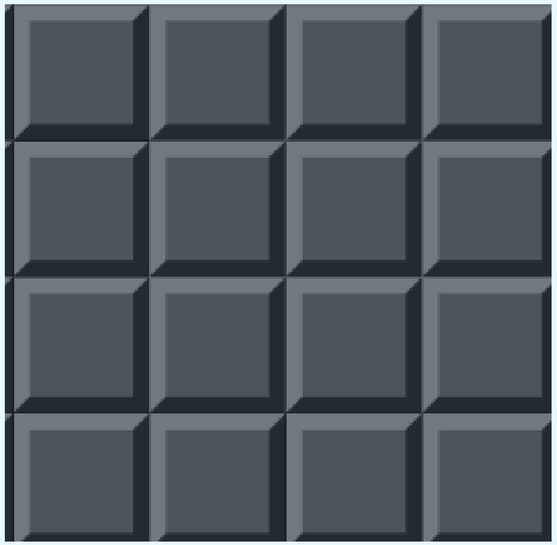
This project implements a text-based version of Minesweeper using Java, offering users a choice of three difficulty levels: Easy, Medium, and Hard. The program features a grid-based board represented using 2D arrays, where mines are randomly placed, and adjacent cells are evaluated for the number of mines surrounding them. Players interact with the game by revealing cells, flagging suspected mines, and requesting hints to aid their progress.



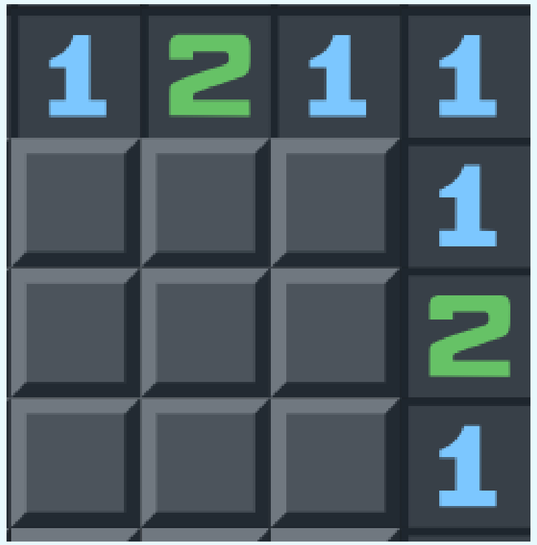
UNDERSTANDING THE MINESWEEPER GAME...

ABOUT THE GAME

The Game is Simple. There is a grid of squares within which are cells with randomly placed mines.



You take turns clicking any cell and it will randomly reveal one or more cells.



ABOUT THE GAME

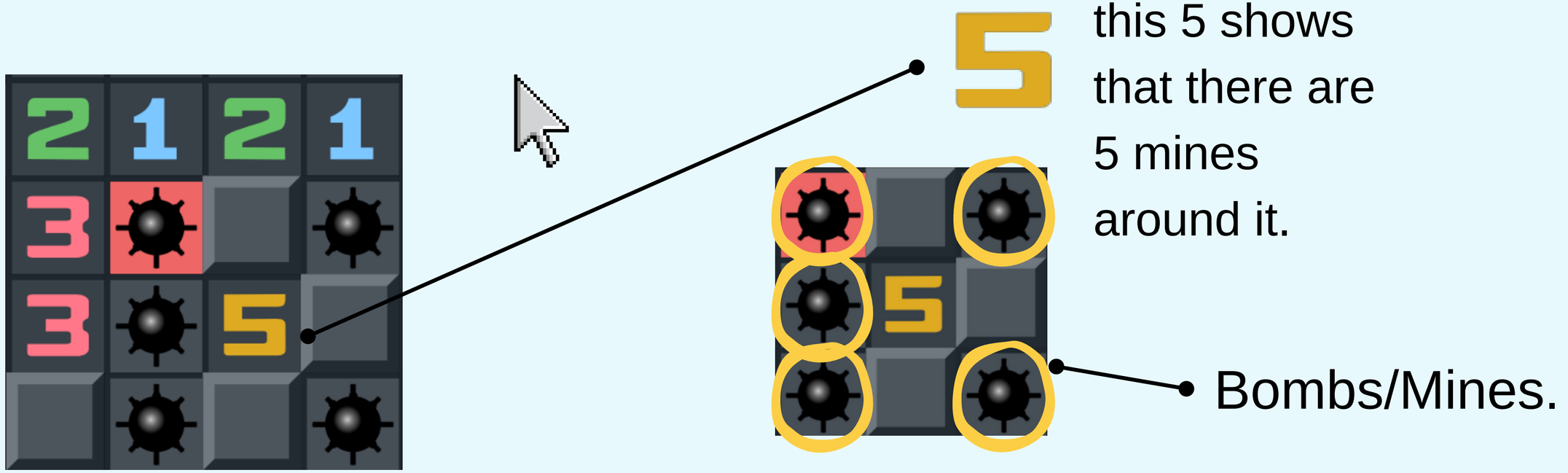
The revealed cells can either be safe cells with some digits written in them or they can be mines. and if you hit mines, you lose the game.

Revealed Safe Cells.

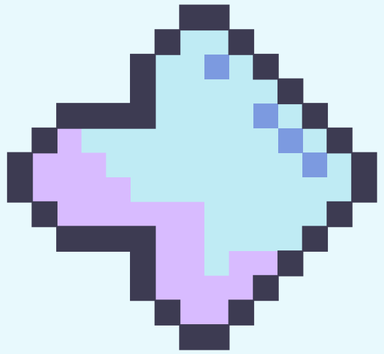
Revealed Mines (Game Over).

ABOUT THE GAME

Revealed Cells that has digits are helpful to predict our next moves.



WORKING OF GAME IN JAVA PROGRAM...



WORKING OF GAME IN JAVA PROGRAM

1. Game Setup

Menu Options:

- New Game: Starts a new game.
- Quit: Exits the program.

- **Choose Difficulty:**
  - Users select a difficulty level: Easy (6x6, 4 mines), Medium (12x12, 10 mines), or Hard (18x18, 20 mines).
  - The selected size and number of mines configure the game grid.

WORKING OF GAME IN JAVA PROGRAM

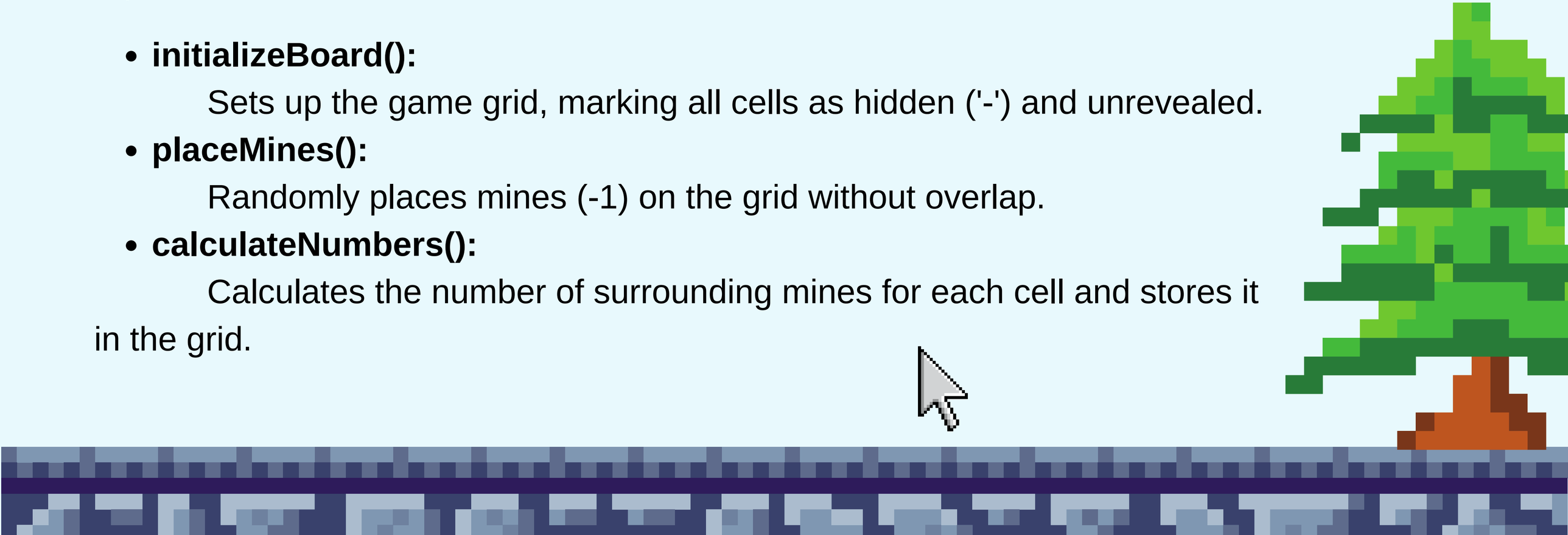
2.Board Initialization

- **initializeBoard():**

  Sets up the game grid, marking all cells as hidden ('-') and unrevealed.
- **placeMines():**

  Randomly places mines (-1) on the grid without overlap.
- **calculateNumbers():**

  Calculates the number of surrounding mines for each cell and stores it in the grid.



WORKING OF GAME IN JAVA PROGRAM

3. Gameplay Mechanics

- The game loop continues until the user wins or loses.
- Users provide input in the form of actions:
  - **Reveal (R row col):** Reveals the specified cell.
  - **Flag (F row col):** Marks a cell with a flag.
  - **Hint (H):** Reveals a safe cell automatically.

- **revealCell():**
  If the cell has no surrounding mines, it reveals all connected safe cells recursively.
- **flagCell():**
  Toggles a flag on or off for a cell.

WORKING OF GAME IN JAVA PROGRAM

4.Game End Conditions

- **Game Over:**

  If the user reveals a mine, the game ends, displaying all mines on the board.

- **Win Condition:**

  The user wins when all safe cells (non-mine cells) are revealed.

  - checkWinCondition(): Checks whether all non-mine cells are revealed.

WORKING OF GAME IN JAVA PROGRAM

5.Utility Functions

- **displayBoard():**

  Displays the current game board, showing cells as hidden, flagged, or revealed.

- **hintSystem():**

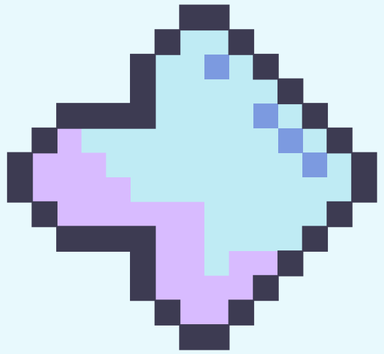
  Helps the player by automatically revealing a safe cell.

WORKING OF GAME IN JAVA PROGRAM

6.Final Display

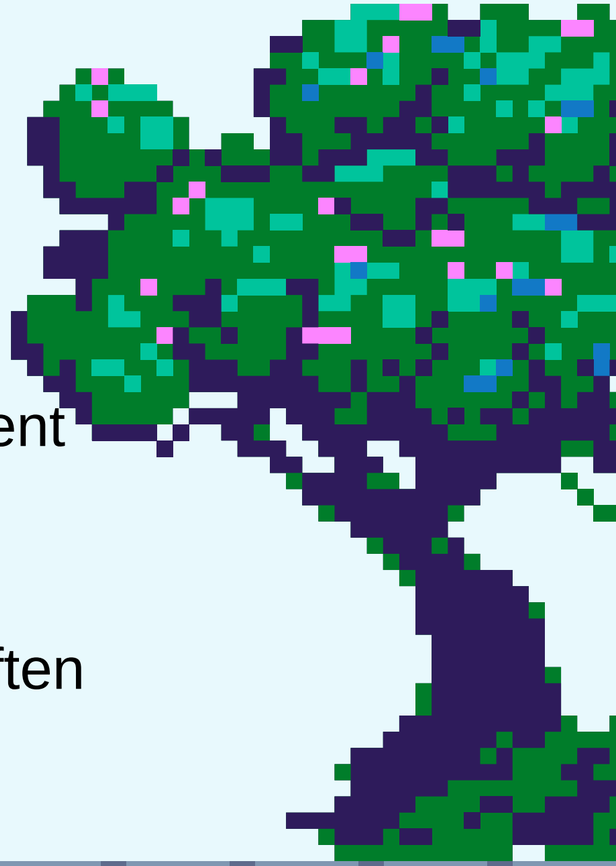
- At the end of the game (win or loss), all cells are revealed.
- The total time taken to play the game is displayed.

DISCRETE MATHEMATICAL CONCEPTS IN PROGRAM...



DISCRETE MATHEMATICAL CONCEPTS IN PROGRAM

Our Minesweeper Program uses many discrete mathematical structures and implementations.



1. MATRIX REPRESENTATION AND INDEXING

· The program uses a 2D matrix (minefield, board, revealed) to represent the game board, which is a classic discrete structure.

· Matrix traversal, as seen in the initializeBoard(), placeMines(), and calculateNumbers() methods, involves adjacency checks, a concept often used in graph theory and discrete math.

DISCRETE MATHEMATICAL CONCEPTS IN PROGRAM

2. LOGICAL ADJACENCY CALCULATION

· In the calculateNumbers() method, the logic to determine the number of surrounding mines for each cell involves iterating over neighboring cells using nested loops. This can be linked to concepts in graph theory where nodes are connected to adjacent nodes.

· The bounds-checking logic ensures cells are within valid indices, showcasing conditional logic derived from set membership.

DISCRETE MATHEMATICAL CONCEPTS IN PROGRAM

3. RECURSIVE EXPLORATION

· The revealCell() method recursively reveals connected empty cells. Recursion is highly used programming technique and is a very important discrete mathematical structure as well.

DISCRETE MATHEMATICAL CONCEPTS IN PROGRAM

4. LOGICAL CONDITION EVALUATION

There are so many, SO MANY logical conditions used in the Program. they are literally everywhere. from the loops to if statements and the boolean values and also in normal block of codes. Next Slide shows all the evidences.

4. LOGICAL CONDITION EVALUATION

```
for(int i=0;i<SIZE;i++)
{
    for(int j=0;j<SIZE;j++)
```

```
minefield[row][col] = -1;
placedMines++;
```

```
if(minefield[i + x][j + y] == -1)
```

```
if(minefield[i][j] == -1)
```

```
while(placedMines < MINES)
```

```
board[i][j] = (minefield[i][j] == 0) ? ' ' : (char) ('0' + minefield[i][j]);
```

```
private static boolean[][] revealed;
3 usages
private static boolean gameRunning = true;
```

```
board[i][j] = (minefield[i][j] == 0) ? ' ' : (char)
```

```
if (row < 0 || row >= SIZE || col < 0 || col >= SIZE)
```

```
if(row+i>=0 && row+i<SIZE && col+j>=0 && col+j<SIZE)
```

```
gameRunning = false;
```

```
board[row][col] = (minefield[row][col] == 0) ? ' ' : (char) ('0' + minefield[row][
```

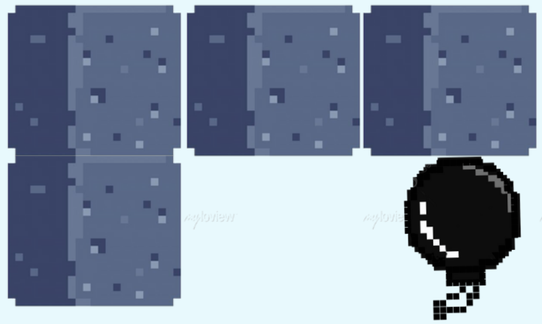
```
if (!isNumeric(parts[1]) || !isNumeric(parts[2]))
```

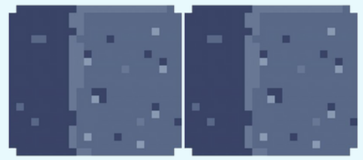
```
if(minefield[i][j] != -1 && !revealed[i][j])
```

```
int row = random.nextInt(SIZE);
int col = random.nextInt(SIZE);
if(minefield[row][col] != -1)
```

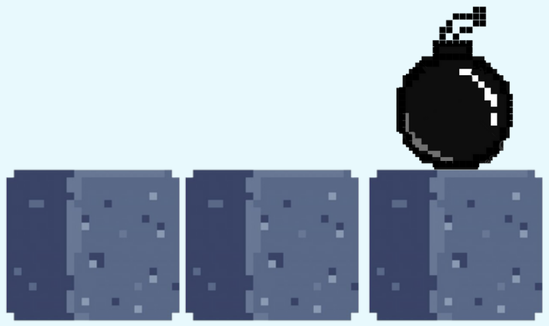
```
if(i+x>=0 && i+x<SIZE && j+y>=0 && j+y<SIZE)
```

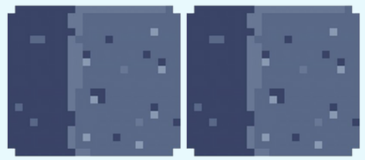
```
if (parts.length < 1 || parts.length > 3)
```

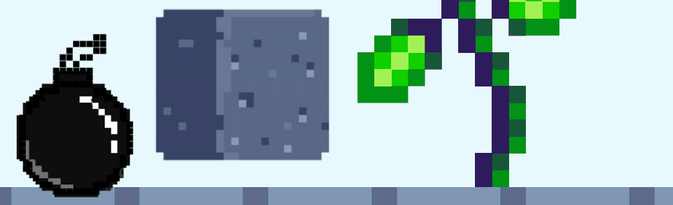




i really meant it when i said "SO MANY".







DISCRETE MATHEMATICAL CONCEPTS IN PROGRAM

5. PROBABILITY

· While not explicitly computed, placing mines randomly in placeMines() introduces a probabilistic aspect. This involves to analyze configurations or probabilities of certain cell states.

CONCLUSION

In conclusion, the Minesweeper game not only meets the requirements of a fun and challenging puzzle game but also serves as a robust learning tool for exploring discrete mathematics and programming concepts. It can be expanded further by adding more advanced features such as a more sophisticated hint system, difficulty scaling, and the ability to save and load game progress.


and that's it.

THANK YOU!