**FRONT-END ENGINEER**

### **Challenge Project: Find Your Hat**

const prompt = require('prompt-sync')({ sigint: true });

const hat = '^';

const hole = 'O';

const fieldCharacter = '░';

const pathCharacter = '\*';

class Field {

  constructor(field) {

    this.field = field;

    this.currentPosition = this.getRandomStartPosition();

    this.field[this.currentPosition[0]][this.currentPosition[1]] = pathCharacter;

  }

  print() {

    for (const row of this.field) {

      console.log(row.join(''));

    }

  }

  isOutOfBounds(row, col) {

    return row < 0 || col < 0 || row >= this.field.length || col >= this.field[0].length;

  }

  isHole(row, col) {

    return this.field[row][col] === hole;

  }

  isHat(row, col) {

    return this.field[row][col] === hat;

  }

  move(direction) {

    const newRow = this.currentPosition[0] + direction[0];

    const newCol = this.currentPosition[1] + direction[1];

    if (this.isOutOfBounds(newRow, newCol)) {

      console.log('Out of bounds! Try again.');

    } else if (this.isHole(newRow, newCol)) {

      console.log('Oops! You fell into a hole. Game over.');

    } else if (this.isHat(newRow, newCol)) {

      console.log('Congratulations! You found the hat. You win!');

    } else {

      this.field[this.currentPosition[0]][this.currentPosition[1]] = fieldCharacter;

      this.currentPosition = [newRow, newCol];

      this.field[newRow][newCol] = pathCharacter;

    }

  }

  getRandomStartPosition() {

    const row = Math.floor(Math.random() \* this.field.length);

    const col = Math.floor(Math.random() \* this.field[0].length);

    return [row, col];

  }

  static generateField(height, width, holePercentage) {

    const totalTiles = height \* width;

    const numHoles = Math.floor(totalTiles \* (holePercentage / 100));

    const field = new Array(height).fill(null).map(() => new Array(width).fill(fieldCharacter));

    let hatRow, hatCol;

    do {

      hatRow = Math.floor(Math.random() \* height);

      hatCol = Math.floor(Math.random() \* width);

    } while (hatRow === 0 && hatCol === 0);

    field[hatRow][hatCol] = hat;

    for (let i = 0; i < numHoles; i++) {

      let holeRow, holeCol;

      do {

        holeRow = Math.floor(Math.random() \* height);

        holeCol = Math.floor(Math.random() \* width);

      } while (field[holeRow][holeCol] !== fieldCharacter);

      field[holeRow][holeCol] = hole;

    }

    return field;

  }

  play() {

    while (true) {

      this.print();

      const direction = prompt('Which way? ').toLowerCase();

      let moveVector;

      switch (direction) {

        case 'up':

          moveVector = [-1, 0];

          break;

        case 'down':

          moveVector = [1, 0];

          break;

        case 'left':

          moveVector = [0, -1];

          break;

        case 'right':

          moveVector = [0, 1];

          break;

        default:

          console.log('Invalid direction. Use "up", "down", "left", or "right".');

          continue;

      }

      this.move(moveVector);

    }

  }

}

const generatedField = Field.generateField(6, 8, 20); // Adjust the dimensions and holePercentage as needed

const myField = new Field(generatedField);

myField.play();