SEC

Lab #1 - Treasure Hunt

- This lab will be graded. You can either submit this one or the Globbing Lab.
- The quality of your code will be graded.
- The quality of your design will be graded.
- Your submission has to be in Rust.
- Your code should be tested and you should provide your tests.
- We provide you with a template for displaying the board but you are not required to use it.
- You do **not** need to submit a report.

The goal of this lab is to implement a small treasure hunting game. The user has to search for a randomly generated treasure in a grid.

1 Game Rules

1.1 Initialization

- At startup, the player should be able to select the color of their character.
  Format: We accept color names (e.g., `blue`) and RGB inputs (e.g., `200,103,104`).
- The board has a size of 15x15 blocs.
- The player and the treasure are placed randomly on the board.

1.2 Actions

Once the game is initialized, the player can choose among three actions:

1. `Move`: following movement rules below.
2. `Search`: looks if the treasure is on their position.
3. `Quit`: leaves the game.

Format: Actions should be input either using their name (`Move`, `Search`, `Exit`) or using an integer representing their position in the menu.

1.3 Movements

A player can move on the board by entering the desired destination coordinate. The player can move **at most** four squares at the time. **Diagonal movements are allowed**. This means that he can go from the coordinate [0, 0] to the coordinate [4, 4].
Format: The coordinates `x` and `y` should be integers in base 10 or in base 16 (prefixed by '0x'). It must be entered in the following formats: `(x,y)` or `[x,y]`. Any spaces should be ignored. Possible errors are:

- Outside board.
- Bad format, if the enclosing format is not respected.
- Bad number of dimensions in provided coordinate (e.g., with (1, 2, 3)). The error should indicate the number of dimensions provided as input.
- A generic parsing error specifying the problematic part.

You are free to handle more errors.

1.4 Search

A player can look for the treasure at their current position.

- If the treasure is hidden there, we congratulate the player and leave the game.
- If not, display a message indicating the **distance to the treasure in blocs**. The distance should be the **shortest path** to the treasure (Euclidean distance).
- Save and display the searched position on the board.

1.5 Board

The board is displayed each turn (action request) using the `print()` function provided in the template. The board should display:

- The current position of the player
- Positions where the player has already looked for the treasure

1.6 Inputs

Entries must be validated and correspond to the different **Formats** described in this document. In case of an input error:

- Display an explicit error message
- Immediately request a new entry

1.7 Bonus

Bonus points will be given to any cool feature added (documented in a small readme).