University of Batna 2
Department of Computer Science
DSA - Data Structures and Algorithms

Project - Mastermind

In this game, two players compete: codemaker and codebreaker. The codebreaker must guess, by successive deductions, the color and position of 5 pawns hidden behind a screen placed by the codemaker.

Game principle

- The codemaker starts by placing 5 random pawns, taken from 8 colors, behind the screen.

- To guess this combination, the codebreaker places/chooses 5 pawns.
  - If one of the proposed pawns corresponds by its **color and position** to a hidden pawn, the codemaker indicates it by signaling a **well-placed pawn**.
  - If one of the pawns matches only by its **color**, the codemaker indicates it by signaling a **misplaced pawn**.
  - The codebreaker keeps suggesting new combinations until the codemaker responds with **5 well-placed pawns** or the maximum number of attempts is reached.

Assigned work

```
const int NUM_PAWNS    = 5;
const int NUM_COLORS   = 8;
const int NUM_ATTEMPTS = 10;

enum color_t{RED, GREEN, BLUE, YELLOW, BLACK, WHITE, GRAY, PURPLE};

// typedef enum color_t board[NUM_PAWNS];
```

With the above data structure, write the following three functions:

1) To retrieve the proposal of the codebreaker.

```
void read_proposed_combination(enum color_t board[]);
```

2) To compare the combination of the codemaker with the proposal of the codebreaker, and deduce both the number of well-placed and misplaced pawns.

```
void evaluate_proposed_combination(enum color_t hidden_combination[],
                                   enum color_t proposed_combination[],
                                   int *num_well_placed_pawns,
                                   int *num_misplaced_pawns);
```

3) To run the game until the codebreaker guesses the hidden combination, or the maximum number of attempts is reached.

```
void game();
```

Note that you can add other functions to your program if needed. For instance:

```
void generate_hidden_combination();
```