SESSION 7: MODULARIZATION (II)

GOAL:

- Practice the concepts studied in theory about modular programming (organizing the code into methods).

EXERCISE:

Implement a modularized program for a simplification of the minesweeper game with the following characteristics and functionalities:

- At the beginning of the program, the board will be initialized:
  - The board will be represented by a matrix (2D array) of NxM integers. The dimensions are to be entered from the keyboard, with the smallest matrix that can be created being 2x2.
  - The program will mark X positions of the matrix with the value 1 (bomb), obtaining these positions randomly. The value of X shall be entered by the user. The value of X has to be between 1 and (N*M)/2. The rest of the squares will initially contain the value 0 to indicate that they are available.
  - In case of wrongly entered values, the program will indicate that there is an error and will ask for the value again.
  - When assigning the squares with a bomb, it must be checked that the randomly selected position does not already have a bomb assigned to it. If it already has one, the program should try a different position until one that is without a bomb is obtained.
- The game is then started, which will end if the user uncovers a square with a bomb or if the user manages to uncover all the squares that do not have a bomb. To do this, the user will select a position on the board (it must be a valid position), and the program will check the selected square:
  - If it corresponds to a square with a bomb (1), this will be reported, the status of the board will be displayed and the program will terminate.
  - If it is an available square (with a 0):
    - it will be marked with a 2.
    - all the squares immediately around it will be examined to count the bombs (value 1) in them, then informing the user about that number of bombs around the selected square
    - The program will ask for another position on the board as long as there are available squares. If there are no available squares left, a congratulations message and the final state of the board will be displayed.
  - If it is a previously selected square (value 2), this case will be indicated to the user and the game will continue.

The methods developed in Assignment 6 must be used. You can make minor modifications to the code of Assignment 6 (renaming constants, methods, parameters, variables, modifying the messages displayed to the user) to adapt it to the context of the minesweeper game and facilitate the understanding of the code.

MILESTONE 3: SUBMISSION OF ASSIGNMENTS 6 AND 7: 22 December

EXAMPLE OF EXECUTION:

```
Enter the amount of rows for the board (greater than or equal to 2): 3
Enter the amount of rows for the board (greater than or equal to 2): 4
Enter the amount of bombs [1, 6]: 4

THE GAME STARTS:

Choose a square:
Introduce the row of the square [0, 2]: 0
Introduce the column of the square [0, 3]: 1
The square is available.
The selected square is surrounded by 2 bombs.

Choose a square:
Introduce the row of the square [0, 2]: 2
Introduce the column of the square [0, 3]: 1
The square is available.
The selected square is surrounded by 2 bombs.

Choose a square:
Introduce the row of the square [0, 2]: 1
Introduce the column of the square [0, 3]: 3
The square is available.
The selected square is surrounded by 2 bombs.

Choose a square:
Introduce the row of the square [0, 2]: 1
Introduce the column of the square [0, 3]: 0
The square is available.
The selected square is surrounded by 2 bombs.

Choose a square:
Introduce the row of the square [0, 2]: 0
Introduce the column of the square [0, 3]: 0
There is a bomb in the selected square.

Board:

1 2 0 1
2 0 1 2
1 2 0 0

End of the program
```