CMPS4143: Contemporary Programming Languages
Major Programming Assignment 4
Due: Tuesday, Sept 27 2022 75 points

PURPOSE: To develop a windows application that implements an interactive game; to use a variety of GUI components; to use a message box; to write a program using more than one class; to use two dimensional arrays and array methods; to use Random class; to use a for loop and a do-while loop; and to use the ? operator. *Remember you will be graded on applying the concepts learned in class.*

PROBLEM: "**Find the Tropical Island" Game**

This game lets you travel to an exotic island (perhaps one in the Caribbean) …but only if you can find it. You are flying an airplane and are on your way to this paradise when the navigation system breaks down and looses part of its ability to communicate with you (no doubt due to a hardware problem). The only way to navigate now is through trial (qualified guessing) and error, accompanied by some feedback from the navigation system, which now only provides a hint of the island's location relative to that of each guess. A guess consists of two numbers (row and column) specifying a point on the map. When the guess is entered into the navigation system, it will tell you whether the island is

- To the East or the West of the point (guess)
- To the North or the South of the point (guess)
- If the row of the point (guess) is the same as that of the island's location (on the same horizontal line)
- If the column of the point (guess) is the same as that of the island's (on the same vertical line)

The trial and error will continue until you enter the coordinates that match those of the island's location.

We want a program to simulate the situation just described. The user is initially shown a map of the area where the island is located, but, due to the faulty navigation system, only water (in the form of the symbol ~) can be seen here initially. The user determines the size of the map at the start of the game by entering the number of rows and columns. Each point on the map is identified with a row index and a column index. Here is an example of the map

```
  0123456789
0 ~~~~~~~~~~
1 ~~~~~~~~~~
2 ~~~~~~~~~~
3 ~~~~~~~~~~
4 ~~~~~~~~~~
```

The location of the island is chosen randomly and secretly by the program. The island is thus hiding behind one of the "waves" (~).

The user is now ready to enter guesses. Each guess must consist of a location with a row and a column number. The feedback accompanying each guess is to be provided in the following way.

Every second guess is evaluated in terms of whether the island is located to the north or the south of the provided guess. All other guesses are evaluated in terms of east/west.

After each guess, a new map is displayed with a new letter printed in the location of the guess. An N on the map indicates that the island is to the north of the guess, an S to the south, an E to the east, and a W to the west. IF the guess happens to have the same row as the island during a north/south evaluation, an R (For Row) is printed on the map. Similarly, if the guess during and east/west evaluation is found to have the same column as the island, a C is printed on this location. When the pilot (user) through luck (and hopefully clever logical deductions) finally enters a location matching that of the island, an I is printed in this position and the pilot is congratulated along with a message telling her how many (or few) guesses she used in total. The game is then terminated and the user can play another game.

Example of a completed game.

```
  0123456789
0 ~~S~C~~S~~
1 ~E~~~~~~W~
2 R~~~I~~R~~
3 ~E~~~~~~W~
4 ~~N~C~~N~~
```

DESIGN HINTS: Utilize at least two classes – NavigationSystem and FindTheIslandGame (which controls the main flow of the game).

NavigationSystem class – utilizes the two-dimensional map. It further holds the secret location of the island and also keeps track of the number of guesses entered by the user. Methods include:

- A constructor to perform the following tasks:
  - Create the new map with the number of rows and columns specified by the user and assign its reference to the map instance variable.
  - Randomly generate the secret location of the island and store the coordinates for later comparisons to the guesses.
  - Fill the map with "waves" (~) for the first view.
  - Set the guess counter to zero.

- PrintMap - Printing out the map requires quite a few lines of code due to headings alignments and for loop. Called from outside the NavigatorSystem, as well as from within.
- EvaluateGuess – evaluates guess and provides the feedback by printing a new map, returns a bool reflecting whether guess is correct or not.

FindTheIslandGame -

- Main method – controls the overflow of the game. It starts a new game, lets the user enter guesses until the island is found, and will repeat until the user quits the program. It presents the guesses of the user to the NavigationSystem for evaluation and feedback.

INPUT: User should be able to

- Enter number of rows and columns for the map size.
- Enter a row and a column number for their guess (and do this over and over until they find the island). ***Must be within limits of map size!!!***
- Indicate whether they want to play another game or submit a guess for current game (may need to be enable or disabled at appropriate times).

OUTPUT: Output is all done on a form. The form should display

- Map size
- Instructions to play the game
- Randomly generated map for a game – after each guess it will need to be updated. Use Courier New Font.
- User's guess
- Response to the user's guess
- Button(s) to play a new game or submit a guess – I'll give you some leeway on this

TURN IN all materials in a 9x12 envelope:

1) Print out of documented Source code
2) Screen dump(s) of image(s) when running
3) Application (.exe file) on some storage media. The application is in a debug folder.