CMPT 103 – Project Milestone #2

General Information

Python version and IDE:	Python 3.3 / Wing IDE 101
Allocated lab time:	2 hrs and 50 min
Due date:	At the end of the lab period
Lab weight:	7%

Topics

Design and Implementation of Celtica

Before coming to the lab, please read the lab specification carefully.

The goals of this lab are twofold:

1. Build a graphical user interface of Celtica
2. Implement functions that can be used to detect legal moves

Submission

- ✓ **All code file (.py) should be submitted electronically** to your Lab Blackboard site.
- ✓ A portion of the total marks (20%) will be allocated for the programming style. For example, functions should be small; avoid writing duplicate code; names should be meaningful and descriptive; naming conventions should be followed consistently; code should be formatted properly; and comments should be accurate and well written.
- ✓ Comments are **required** for:
  - EACH program indicating the student name and program name.
  - EACH function indicating the function purpose, syntax (example usage of the function), parameters, and return value
  - Any block of code for which the purpose may be unclear (Note: you should always try to write clean code that can be understood easily without comments).

Assignment

For this project, please put all functions into a file called `Celticayour_initials.py` (e.g., `CelticaFL.py` where F and L are the first letter of your first name and last name). Please feel free to write helper functions if necessary.

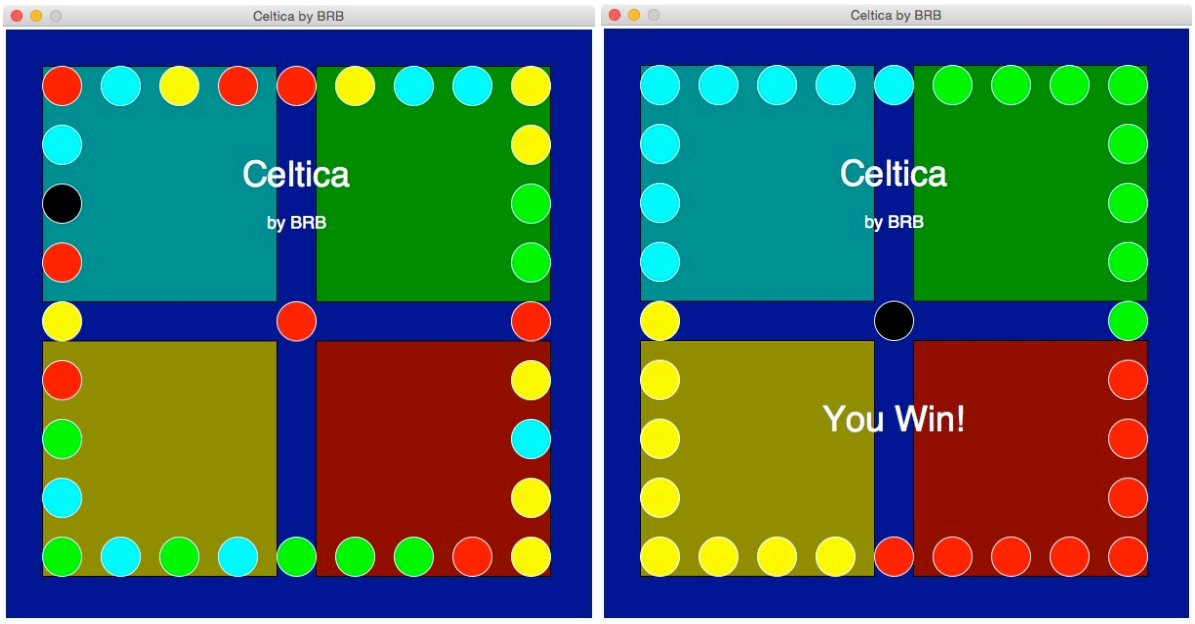

Figure 1: The winning configuration (shown above right) has each of the colours together in the four corners of the board and the open hole in the center.

Celtica (original design by B. Brookwell) is a game consisting of square track of 32 holes around with a single hole in the centre. The board is usually drawn in the form of a Celtic cross (hence the name). Each hole can hold a single marble (red, green, blue, or yellow). One hole is always left open. The player clicks on a circle beside the open/black hole, and the circles/colours exchange. A marble in the centre has four neighbours. The player clicks on circles until all the red, green, blue, and yellow circles are in their correct positions (see Fig. 1). Please see a demo in the lab to learn more about this game.

From the first project milestone, you have implemented the following functions (please revisit Project Milestone #1 description):

```
board = setup_game()
display(board)
exchange(board, first, second)
bool = is_game_over(board)
```

For the second milestone, please use the functions above as a starting point to think about your program design. However, feel free to modify the functions above and create additional functions to implement the features below:

1. [70 marks] The `display(board)` function from the first milestone shows the marbles in the Celtica game board using text. Modify this function so that it displays the Celtica game board in a graphical form. For example, given the initial configuration of the board from the `setup_game()`, the display function should show the contents of the board as illustrated in Fig. 2.

   Hints:
   - Include a `main` function to demonstrate that the `display` function works properly.
   - Declare global constant variables if appropriate.
   - Store `Circle` objects drawn in a suitable data structure (later, you'll need these objects to detect if a mouse click occurs inside a circle).

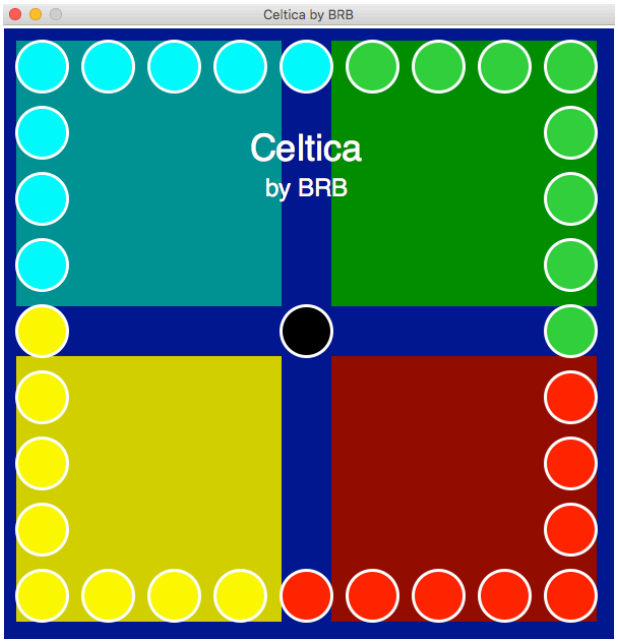

Figure 2: The initial configuration of the Celtica game board.

2. [30 marks] Write a function named `is_legal_move` that takes a Celtica game board and an index representing the location of a marble that is clicked on by the user. This function returns `True` if one of the neighbours of the clicked marble is the empty slot. Otherwise, it should return `False`.

Syntax:	`bool = is_legal_move(board, index)`
Parameters:	`board – Celtica game board` `index – index of the marble being clicked on`
Return value:	`True if the move is legal, False otherwise`

Hints:

- Being able to get a list of neighbours of individual marbles can help solve this problem. Note that some marbles have more than two neighbours.
- Following the Project Milestone #1 specification, here is the expected behaviour of the `is_legal_move` function:

```
>>> board = setup_game()
>>> is_legal_move(board, 23)
    True
>>> is_legal_move(board, 15)
    True
>>> is_legal_move(board, 7)
    True
>>> is_legal_move(board, 31)
    True
>>> is_legal_move(board, 1)
    False
>>> is_legal_move(board, 21)
    False
```