CMPT 103 – Project Milestone #1

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:	3%

Topics

Design and Implementation of Celtica

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

The goals of this lab are threefold:

1. Understand how Celtica is played
2. Decide on the basic data structure
3. Create the data structure and implement some key functions in the game

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.

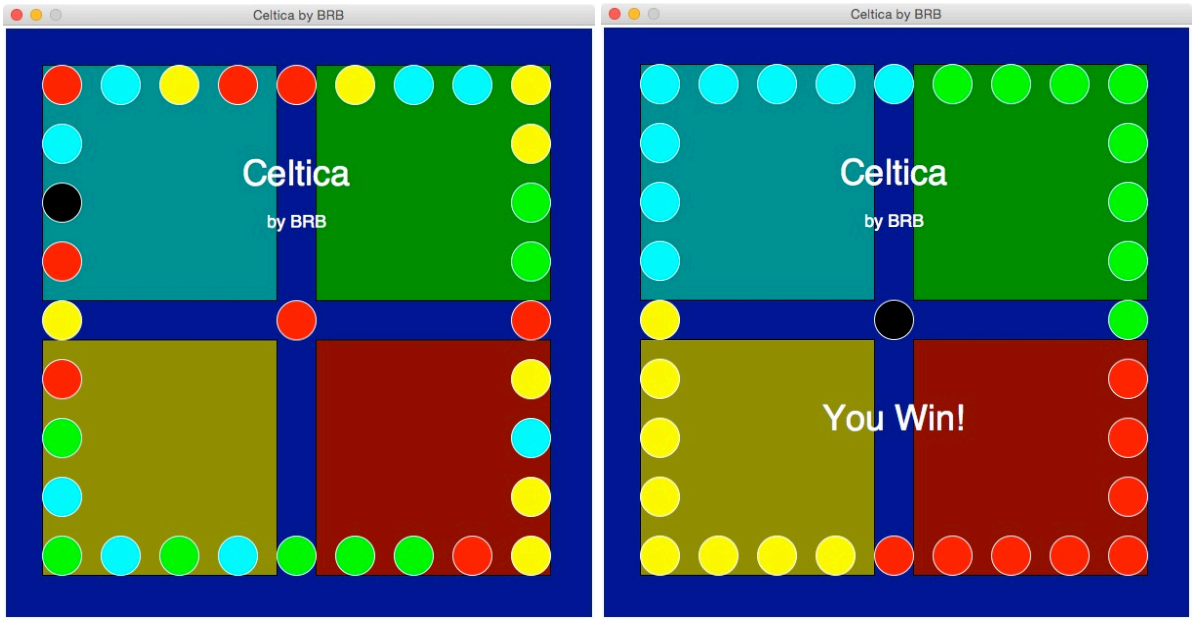

Figure 1: The winning configuration (shown above right) has each of the colours together in the four corners of the board and the open hole in the center.

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.

1. [20 marks] Create a function named setup_game that creates and returns a list containing the marble colours in each of the holes. Make the first eight be G (for green), the next eight be R (for red), the next eight be Y (for yellow), and the next eight be B (for blue). Add one extra value X to represent the open slot in the centre.

Syntax:	`board = setup_game()`
Parameter:	`None`
Return value:	`board – a list of characters representing the colour of the marble in each hole of the Celtica game board`

```
>>> board = setup_game()
>>> print(board)
    ['G', 'G', 'G', 'G', 'G', 'G', 'G', 'G', 'R', 'R', 'R', 'R', 'R', 'R',
```

Note: the screenshot above does not show all contents of the board.

2. [50 marks] Create a function named `display` that displays the marbles in the Celtica game board using text (you'll build the graphical interface later). Display the values in the form of a square. For example, given the initial configuration of the board from the `setup_game` above, the `display` function should display the contents of the board as follow:

```
BBBBBGGGG
B       G
B       G
B       G
Y   X   G
Y       R
Y       R
Y       R
YYYYRRRRR
```

Syntax:	`display(board)`
Parameter:	`board – a list of characters representing the Celtica game board`
Return value:	`None`

3. [15 marks] Write a function named `exchange` that swaps the colours in two locations of the Celtica game board.

Syntax:	`exchange(board, first, second)`

Parameters:	`board – Celtica game board where the exchange takes place on`
	`first – index of the first marble being exchanged`
	`second – index of the second marble being exchanged`

Return value:	`None`

```
>>> board = setup_game()
>>> print(board)
    ['G', 'G', 'G', 'G', 'G', 'G', 'G', 'G', 'R', 'R', 'R', 'R', 'R',
>>> exchange(board, 0, 9)
>>> print(board)
    ['R', 'G', 'G', 'G', 'G', 'G', 'G', 'G', 'R', 'G', 'R', 'R', 'R',
```

4. [15 marks] Write a function named `is_game_over` that determines whether or not a Celtica game board contains the winning configuration.

Syntax:	`is_over = is_game_over(board)`
Parameter:	`board – Celtica game board`
Return value:	`True if the board contains the winning configuration, False otherwise`