**Winter 2016 - Cmpt 103**

**Section 41 – Lab 9 - Milestone 2 mark sheet**

**Student name: Metehan Dagsuyu Total: 90 marks**

**General: Code style [9 marks]**

* EXCELLENT reuse of milestone 1 (import CelticaMDLab5 as C1)!: **+1**
* ALL functions have properly documented function headers
* APPROPRIATE AMOUNT OF comments within the code
* GOOD spacing within the code
* POOR choice of variable names in most places: **-1**
  + Most of your variable names (yMarbleSize, sHalf, circleList, boardLocation) do not use the underscore approach (y\_Marble\_Size, s\_Half, etc.) which is what students are required to do in Cmpt 103
* ONE function is more than 20 lines: **-1**
  + display is around 46 lines long
  + find\_Index is around 26 lines long (not counted for this milestone)
  + You could create a function to add\_circle and call that to create and build your circle. Then you could add the returned circle to your circleList to shorten your code in the display function.

**Question 1: display [57 marks]**

* Function header for display:
  + The syntax should start with ‘circleList =’ as that is returned. You may wish to return circleColorList as well: **-0.5**
  + The return value is missing:  **-0.5**
* Problems with code:
  + There are two big problems in your program. In display, you define circleList as circleList = [[0]\*9]\*9 and circleColorList as circleColorList = [[""]\*9]\*9 but this is incorrect. Any changes to any position in one of the inner lists affects all of the inner lists as they are exact copies of one another. This is not what you want. Instead, do this circleList = [[0] \* 9 for i in range(9)] and this circleColorList = [ [""] \* 9 for i in range(9)]. They may seem the same but they are not. The second approach makes new lists each time so any changes are not propagated throughout the entire list structure which is what you need. This needs to be corrected for milestone 3: **-2**
* Problems saving the graphical circles into a structure:
  + circleList is not global and cannot be viewed by the main function. You return it to the main function from display but you don’t save it to a variable so it can be used later on: **-1**
  + The original milestone creates board which is a 1D list with 33 positions. For this milestone, you made a different representation. You started with board and then took each row and converted it to a string and then you stored each row into a list creating a 9 x 9 list. It works correctly but keeping in mind there will be some positions within the list that are empty as nothing will be stored there. To represent the circles you created circleList and circleColorList which are both 9 x 9 lists. circleList holds the actual graphical circles and circleColorList holds colors in string form (blue, lightgreen, yellow, red, and black).
  + Overall your approach works correctly. When I tested to see whether a particular circle could be changed, I found it could by accessing a circle within your list. HOWEVER, I played around a bit more to see what your find\_Index function was doing and, although it does take a point click and turn it into a position between 0 and 32, it also took some clicks that weren’t on any circles and turned them into a position within that range as well. You need to make sure this doesn’t happen for milestone 3.
  + If it’s easier for you, you can convert your find\_Index so that it returns a value between 0 and 81 or a tuple (0-9, 0-9). It doesn’t have to work with 0 – 32 if that’s not convenient for you. What’s more important is that you can make changes to the board during the game and propagate those changes to the circles. Think about how you wish to do this and what would work best for you.
* Function header for main:
  + Missing parameters and return value: **-1**

**Question 2: is\_legal\_move [24 marks]**

* Function header for is\_legal\_move:
  + The return value should start with ‘legalMove –‘ as that is what is returned.
* Does not properly determine the left neighbour for position 0 is 31. 0 has no left neighbour in your code: **-2**
* Does not properly determine the right neighbour for position 31 is 0. The right neighbour is at position ‘index + 1’ which is 32: **-1**
* Missing ‘else: return False’ within the ‘if index in (7, 15, 23, 31):’ section as there is an inner loop that needs to return False if the True conditions are not met: **-1**