

CSCI 141 Computational Problem Solving

Term Project: Tribe Bubbles

In this project, you have a great deal of freedom for creativity. We present minimum requirements for grade thresholds. The more features you add, the higher your maximum score is. We want you to implement the basic functionality of this game, then make it your own by researching how to add features that you think of.

The basic functionality is defined as follows:

1. This is a single-player game on an 8x8 grid.
2. The player places bubbles in the grid by clicking on the squares.
3. After each valid click, the computer chooses a random square (including (1) the square that the player has placed a bubble, (2) the square that the computer has placed a blocker, and (3) the empty square) in the grid to place the blocker.
4. The bubbles disappear when at least four of them are in a vertical, horizontal, or diagonal line.
5. The player receives points that equal to the number of disappeared bubbles in step 4. The points should be kept updated for each bubble placement.

Some additional features include:

6. If the player forms multiple lines of bubbles with a single bubble placement, her multiplier is increased by the number of simultaneous lines she forms. You are free to define the rule for increasing the multiplier as you like, but should explain how your multiplier works in your write-up.
7. If the player forms 0 or 1 line in one bubble placement, her multiplier should be reset to 1.

Here's how the points will work:

For a **maximum of 20 points**, your program must display an 8x8 grid and allow the user to click on the squares. When she clicks, the square should be marked to indicate that it is occupied.

For a **maximum of 30 points**, your program must include all features above, and must generate a blocker with a different color on a random square in the grid after each valid click.

For a **maximum of 40 points**, your program must include all features above, and must ignore clicks outside of the grid, on already-occupied cells, and after the game is over.

For a **maximum of 50 points**, your program must include all features above, and must also make the bubbles disappear when at least four of them are in a vertical or horizontal line.

For a **maximum of 60 points**, your program must include all features above, and must also make the bubbles disappear when at least four of them are in a diagonal line.

For a **maximum of 80 points**, your program must include all features above, and must also update and display the player's score on the screen.

For a **maximum of 90 points**, your program must include all features above, and must also incorporate the multiplier for multiple simultaneous lines of bubbles, resetting to 1 if she forms fewer than 2 lines of bubbles in a single placement.

The remaining **10 points are "dazzle points."** Show us what you can do now that you know the basics of Python programming. Think of some ideas to personalize and improve the game, research how to implement those ideas, and incorporate them into your program. This is your chance to shine. Note that without implementing your own additional features, the maximum score you can receive for this project is a low A-.

Here are some screen captures of the reference implementation. As always, these are examples, not prescriptions. You are free to stylize your program as you like. We chose to use circles, squares and [William & Mary's official colors](#), specifically William & Mary Green and Spirit Gold.

As we get questions of clarification, we may extend this specification. We will post the announcement via Piazza if the specs are updated.

EXAMPLES (NOT PRESCRIPTIONS)

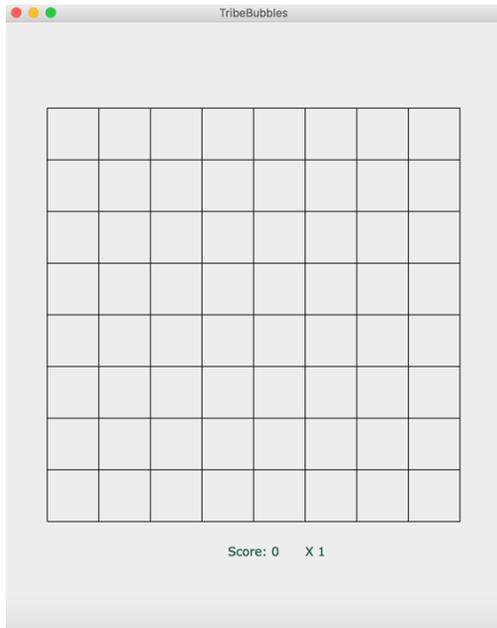


Figure 1: Program upon launch.

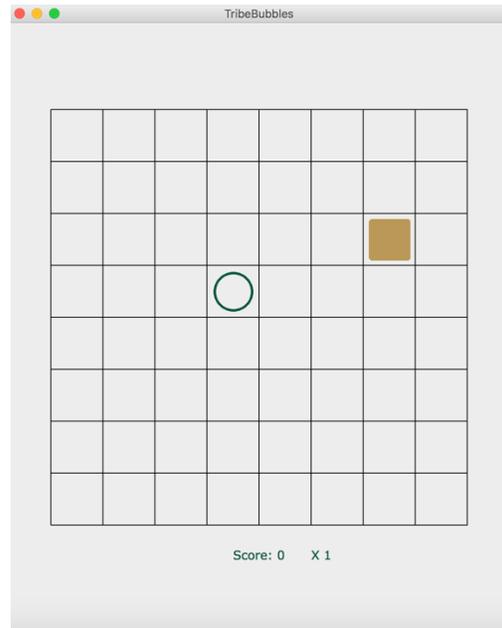


Figure 2: Program after the player clicks on the square 3, 3. Note that the program randomly blocks a square using the yellow blocker.

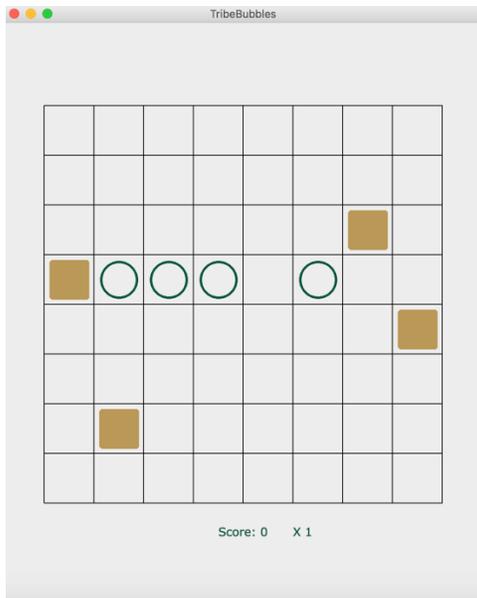


Figure 3: Program after three more valid clicks. Note that score and multiplier are unchanged.

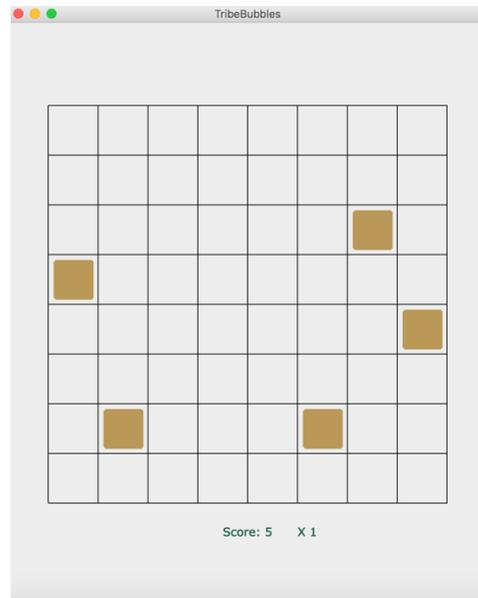


Figure 4: Following Figure 3, the player clicks on the square 3, 4 by forming a row of five bubbles. The bubbles disappear, and the score becomes 5 points.

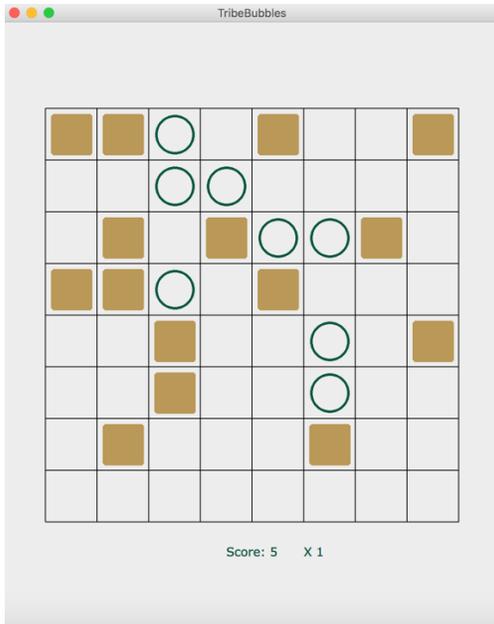


Figure 5: After a few more clicks, the player tries to form more than one line of bubbles in a single placement to gain more points. Now she has one good chance by clicking on the square 3,5.

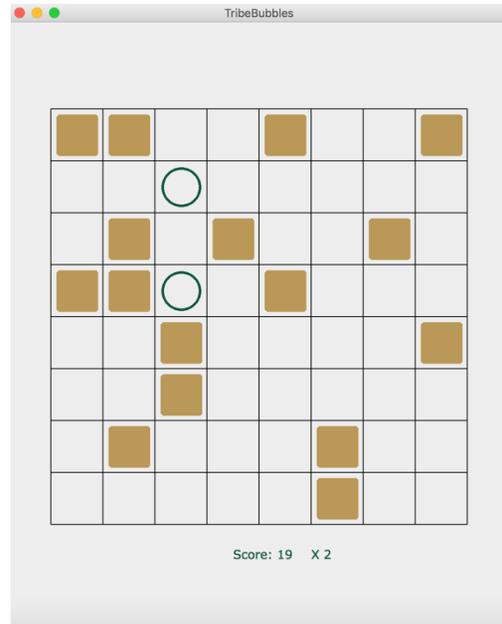


Figure 6: After clicking on the square 3,5, the two lines of bubbles disappear. Note that the score is increased by 14 points: there are in total 7 bubbles disappeared, since they are in two lines, the multiplier is increased to 2, and $7 \times 2 = 14$ points.

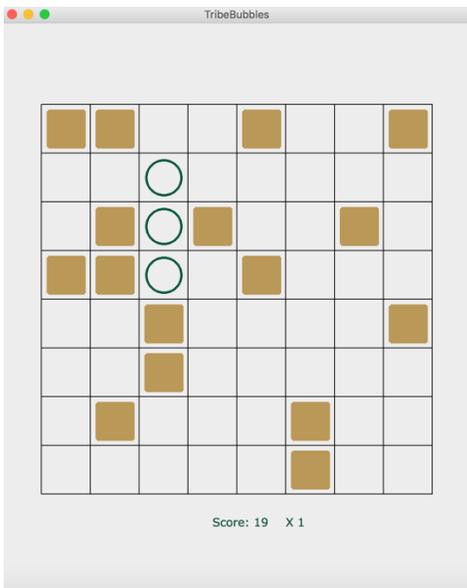


Figure 7: Following Figure 6, the player clicks on the square 2,2. Note that since the player does not form multiple lines, the multiplier is reset to 1.

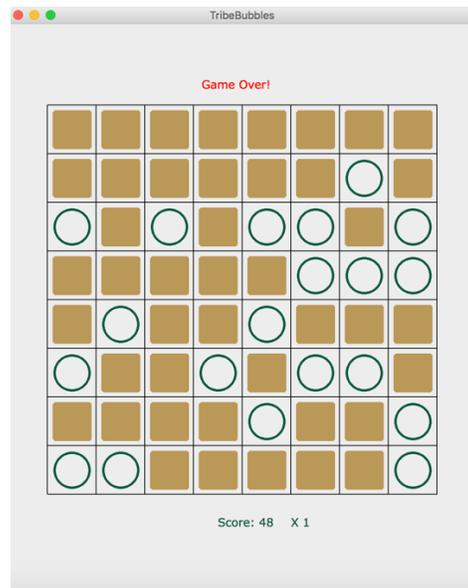


Figure 8: Play continues until all 64 squares are occupied.

SUBMISSION EXPECTATIONS

930xxxxxxx.pdf: A brief write-up describing the design of your program and difficulties you encountered in this project and how you resolved them. Your write-up should also include a justification for dazzle points, if you attempt any, and a description of how to play your game (including descriptions of scoring and multipliers). Use your W&M ID number for the name of the file.

TribeBubbles.py: The file containing your game. A skeleton file is not provided, but you can use files from previous projects and lectures to build from. You may need to import additional modules. **If you encounter trouble importing a particular module, post the problem on Piazza, and we will direct you to the correct import line.**

Dazzle.zip: A zip file containing any other files your program needs in support of your dazzle point. This file is not required for the specified behavior; you may need it depending on your dazzle approach. We will place the contents of this zip archive in the same folder as your TribeBubbles.py. Your program should assume that any additional files are in its same directory.