Name: BinaryBuffs
Team Members: Sahand Setareh, Tahmina Ahmad, Divya Bhat, Tanvi Gopalabhatla

---

User stories

- Player Perspective: As a user of this game, I will have the capabilities of launching attacks that hit, miss, sink my opponent's ships, so that if I sink all my opponent's ships, I would win.
- Player Perspective: As a user of this game, I will have the capabilities of creating a fleet of ships that include a Battleship, a Destroyer, and a Minesweeper, and have the ability to place them on my board which will act as my defense against my opponent throughout the game.
- Player Perspective: As a user of this game, I should be able to hit the captain's quarters on an opponent's ship either once or twice and sink the entire ship. (If the ship is a minesweeper it will require 1 hit on the captain's quarters, and if the ship is a battleship or destroyer it will require 2 hits on the captain's quarters)
- Player Perspective: As a user of this game, if I have sunk at least 1 ship, I should be able to use a weapon called the sonar pulse twice within the game that allows me to see the status of the cells on a portion of my opponent's battleship grid.
- Creator Perspective: In order to create this game through Java, as the developer, we would implement the appropriate classes and methods allowing us to simulate the Battleship game according to the requirements and features we were given.

Planning game

- Implement Captain's Quarters feature
    - Create test for Captains Quarters feature
    - Create setCaptainsQuarters method
    - Create getCaptainsQuarters method
    - Created sinkShip method for status of ship (sunk or not sunk)
    - Edit attack functionality to accommodate the Captains Quarters feature for both armored (Destroyer and Battleship) and unarmored ships (Minesweeper) in the fleet
- Implement Sonar Pulse
    - Create a Weapons class
    - Create SonarPulse class → subclass of Weapons
    - Function to reveal adjacent coordinates of pulse target
        - Return array of coordinates that are occupied by ships as well as empty coordinates with nothing there
    - Keep a counter of sonar pulse uses → only can be used after at least 1 opponent ship sinks

    - Created helper function checkAvailability() to see if the Sonar Pulse feature is available to use yet
- Refactoring code
    - Change place ship input to take in coordinate and direction
    - Change to tests to be in multiple files → organize!

Time estimates and actual time spent

- Estimate:
    - 4 hours a week
- Time Spent This Week:
    - Monday February 22, 2021
        - 2 hours
    - Wednesday March 3, 2021
        - 3.5 hours

Reevaluate project risks

- Goal: Iteration 3:
    - Reviewing expectations, tracking progress, and completing remaining coding portions of project, like the two additional required features
    - This is a checkpoint to see what we have gotten done and how closely we are following the initial expectations and requirements set during iteration 1
    - We will be testing new features and working on other requirements given to us by the instructors for Milestone 3.
- Evaluation:
    - We got the two additional features from the milestone 3 writeup and then implemented them in our code. We had to add new attributes and methods in our classes as well as do a little refactoring so that our code adhered to the new requirements.
    - Goal for next iteration: From our original expectations, we will be:
        - Wrapping up the rest of the project assuming that it is the last milestone/iteration.
        - We initially said we would do our final round of testing and wrap up our code. However, since we have been testing throughout and will continue to test as we implement new features, we may not need the final round of testing.

Meeting minutes

- Public Link: https://docs.google.com/document/d/1-yP3ey-1XFvnBTOsuReMe4rvYLWhN_wZlpJxYQkzW3E/edit?usp=sharing

Resolved outstanding issues from last time:

- Created test classes for each of our classes
    - Each class we use to build this program has its own corresponding test class
- Changed our user input for placing ships based on our TA's feedback.
    - Refactored our code to take in a start coordinate and a direction. This is reflected in our input in our test cases.