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

---

User stories

- Player Perspective: As a user of this game, I will have the capabilities of adding a Space Shuttle to the Space Map, so that if I sink my opponent's Space Shuttles, I would increase my chances of winning and if my Space Shuttles are hit by my opponent, my chances of winning decrease. If a Space Shuttle is sunk, it counts as a hit against any possible ships on every coordinate occupied by the Space Shuttle on the corresponding OceanMap.
- Player Perspective: As a user of this game, I will have the capabilities of reviving two sunken ships by using my Lifesaver boost.
- 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 first new feature - Lifesaver boost
    - Create a new abstract class called Boost
        - Contains resources/power ups that user can use in order to help them throughout the game
    - Create a subclass that extends from Boost, called Lifesaver
        - Lifesaver can revive a full ship that has been sunk
- Implement second new feature - Spaceship
    - Create a new Map superclass called SpaceMap
        - Lies above UnderwaterMap and OceanMap
    - Create a new Ship class called SpaceShuttle
        - 1 x 10 size
        - Only moves 2 directions (North and South or East and West depending on orientation)
- Discuss attack requirements of SpaceShuttle
    - Can only attack a space shuttle with space laser
    - Once whole shuttle is sunk, all the ships that lie on maps underneath will be a "hit"
- Minor refactoring
    - Extracted fragments of code into separate functions to be more extensible

Time estimates and actual time spent

- Estimate:
    - 4 hours a week
    - 10 hours total for this milestone
- Time Spent This Week:
    - Wednesday, March 31:
        - 4 hours
    - Thursday, April 1:
        - 4 hours
    - Friday, April 2:
        - 4 hours

Reevaluate project risks

- Goal: Iteration 5:
    - Reviewing expectations, tracking progress, and completing remaining coding portions of project, like the two new features that would extend our Business Layer
    - 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, 2, 3, and 4
    - We will be testing new features and working on other requirements given to us by the instructors for Milestone 5.
- Evaluation:
    - We came up with ideas for 2 additional features from the milestone 5 write up and then implemented them in our code. We had to add new attributes and methods in our classes as well as do major refactoring so that our code adhered to design principles.
    - We had to reevaluate which features to implement based on our time constraints, so we did not end up implementing the GUI in this milestone
    - We added another "map" to our project which extended our Business Layer and allowed us to implement similar functions from other maps. We made sure to document requirements for the new type of ship that was created, and how it connects to other layers within our existing code. Additionally, our code has been refactored to become easily extensible.
    - Goal for next iteration: From our original expectations, we will be:
        - Implement a GUI for our battleship game.
        - Add one more feature that extends our business layer (some kind of "natural disasters" feature on each map)
        - Figure out how to implement command pattern for the undo/redo method

    - 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/1doyamzLg5sLwpaz_TbbyHpkrtI62rDwON3axN_ueSVM/edit

Resolved outstanding issues from last time:

- Wrote more tests for space laser class (was a concern brought up in interview grading)
- Came up with plan for command pattern for undo/redo
- Fixed code so that we can attack after moving fleet and still keep track of hit ships.
- Refactoring documentation
    - https://docs.google.com/document/d/1Aer1XSyy6-jqRHa87SdRJ0dRoMjDEz60L53i1xaEDKo/edit