Name: BinaryBuffs

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

---

User stories

- **Player Perspective:** As a user of this game, my game maps have the capability to randomly place a narwhal or shark under one of the ships, which will either grant me more uses for my sonar pulse, or bite off part of my ships, respectively
- **Player Perspective:** As a user of this game, at the beginning of every turn, there will be a random disaster that may occur on each of my maps. I will have the ability to see the locations of these disasters, and their effect on my current fleet.
- **Player Perspective:** As a user of this game, there will be a menu by which the user can play the game and interact with it. This menu feature allows the player to randomly place a fleet, or manually place ships on the grid, as well as show grid status, use different weapons at will, show score, use boosts, move the fleet, and surrender the game.
- **Player Perspective:** As a user of this game, the user should be able to generate a fleet of ships and place the fleet randomly on the grid, or manually place each of their ships.
- **Creator Perspective:** In order to create this game through Java, as the developers, we would implement the appropriate classes and methods allowing us to simulate a modified Battleship game. This game would be developed according to the requirements and features given to us by the instructors as well as according to new features we devise ourselves.

Planning game

- We decided to include special animals. Before every turn, the player may encounter a shark or a narwhal under one of their ships. A shark will bite off a part of your ship (similar to an attack on that cell), while a narwhal will give you an extra use for your sonar pulse.
    - MagicAnimals class
        - Jaws class- Bites part of ship
        - Narwhal class- Gives you 1 more use for sonar pulse
- There are 3 different types of disasters that can occur on each map - an asteroid field, a ghost zone, or a hurricane. These disasters are randomized before every turn, with a 10% chance of occuring during a player's turn.
    - Asteroid field- fires 10 randomly placed asteroids at the current player's space grid and has the chance of hitting a space shuttle in the area.
    - Ghost zone disaster- current player faces mysterious ghosts that take over the offensive grid.
        - These ghosts scramble the hit/miss data on the player's offensive grid randomly
    - Hurricane disaster- tosses and turns ships caught on its borders.

        - only occurs on the ocean map and moves the ocean maps ship in specific directions
- Disaster
    - Kept track of in maps
    - Ocean: hurricanes
    - Underwater: Ghost Zone
    - Scramble player's offensive grid
        - Randomly swap cell status
    - Space: Asteroids
        - Multiple random places that get hit
- Implement Game class and game functionality to create a working, fully functional game
- Refactoring old code to improve game features and functionality
- Game Functionality:
- Starting Menu (Composite Pattern)
    - Create random fleet
        - Which map choice (ocean, underwater, space)
        - Creates an object of type Fleet, Call placeFleet
    - Manually place ships
        - Deploy ship on certain coordinates
- While loop:
    - Random number generation to determine weather
        - 0-9 → good weather
        - 10 → hail storm/disasters
            - You suffer the consequences
    - If good weather, show menu
        - Current Turn Menu
            - Show grid status
            - Show score
            - Use Weapon
                - Bomb, Sonar Pulse
            - Use Boost
                - Lifesaver
            - Move Fleet
            - Undo/Redo Move Fleet
            - Surrender
                - Are you sure? (Y, N)

        - Ends game
    - If bad weather
        - Lose your turn

Time estimates and actual time spent

- Estimate:
    - 30 hours
- Time Spent This Iteration:
    - Wednesday 4/14 - 3 hours
    - Saturday 4/17 - 4.5 hours
    - Sunday 4/18 - 4 hours
    - Monday 4/19 - 8 hours
    - Wednesday 4/21 - 12 hours
    - Thursday 4/22 - 12 hours
    - Friday 4/23 - 8 hours
    - Saturday 4/24- 8.5 hours
    - Sunday 4/45- 1.5 hours
- Total Time: 61.5 hours

Reevaluate project risks

- Goal: Iteration 6:
    - 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.
- Evaluation:
    - We came up with many different ideas in order to add more features to extend our business layer and built on old features to implement proper design patterns
        - Added Composite and Command design patterns to existing code
    - We implemented a disasters feature: there are 3 different types of disasters that can occur on each map - an asteroid field, a ghost zone, or a hurricane. These disasters are randomized before every turn, with a 10% chance of occuring during a player's turn.

- We implemented a MagicalAnimals class to add in Jaws and Narwhal features.
- We realized that a GUI didn't add to our business layer and with the time constraints, a simple UI design would be appropriate. Therefore, we implemented the actual game functionality so that we could have a game with a UI that works and takes in user input, so that a user can interact with it and actually play it.
- We drastically refactored our code for attacking under space shuttle, comparing string user input, checkAvailability (in the Bomb class), and more.
- We tested and debugged the game relentlessly to ensure perfect execution in all cases.

Meeting minutes

- Public Link: https://docs.google.com/document/d/14MNHPKBpepE9ZuiCB5A9eBQMZyW5SnTS3O3DxPzvuFY/edit?usp=sharing

Resolved outstanding issues from last time:

- Refactoring documentation
  - See Appendix:
    - https://docs.google.com/document/d/1CsOioenzlVfwhtwyt4mkotTVXdGkWtbcJaqJyfZYKtc/edit?usp=sharing