Project Description: Fishing Simulator
Fishing Simulator will be an arcade-style game where the objective is to catch as many high scoring fish as possible. The character has immediate control of the line's depth while the fish cross under the boat. The challenge arises from the player's limited amount of hooks and the threat of trash or unexpected creatures breaking the line. To add further complexity, larger fish will only bite the line if a smaller fish is on the hook. With the player's choice to reel up the fish or wait for a bigger fish, they risk losing their hook for a higher score.

Competitive Analysis: Ice Fishing from Club Penguin

Club Penguin's ice fishing is a time-based fishing game where the goal is to catch the maximum number of fish possible in a set time. The game features types of fish/debris that scare the fish off the line or lose the bait.

Features:

- Reeling the line
- Baiting a hook
- Hooking a fish
- Catching a fish
- Releasing a fish
- Losing bait
    - Shark - collision with hook
    - Jellyfish - collision with line
- Breaking line
    - crab - cuts line at the surface
- incrementation of fish/enemy types
    - After a certain time if more than X fish have been caught?
    - Text signaling the next part of the game
    - Quantity of fish increase
- End game
    - This game is a set amount of time and ends
- Music
- Sound effects
    - Caught fish
    - Shark going by
    - Jellyfish shock
    - Losing fish

Differences:

- GUI
    - Aesthetics
    - Bubbles?
    - home screen, directions, endgame, pause
- Catch bigger fish with small fish
- Score-based, not quantity based
- Play until you loose
- Progressively harder

Fishing Simulator will share many of the features in Ice Fishing from Club Penguin but also make reference to 'endless' games like flappy bird, doodle jump, and other 'play until you lose' games. Furthermore, the Fishing Simulator will have different fish types and advanced collision reactions based on what fish is on the line and if the collision is with the hook or the line itself

Structural Plan:

There will be 3 files: classes, mvc, and functions

classes: this file has all classes, subclasses, and their functions

- floaters
    - Fish
        - Big fish
        - Bigger fish
    - Enemy
        - Spikey fish/piranha/shark?
        - Crab?
        - Urchin?
        - Jellyfish
        - Squid?
        - Trash
    - Bonus
        - Extra hook
    - Text
        - New enemy
        - Instructions
        - Tips?

mvc: this file has the standard mvc functions and actually runs the game

- appStarted()
- keyPressed()
- mouseClicked()
- mousePosition?
- timerFired()
- redrawAll()
- runApp

functions: this file will have functions called in mvc that are not defined within a class

gameplay

- createFish()
- createEnemy()
- createBonus()

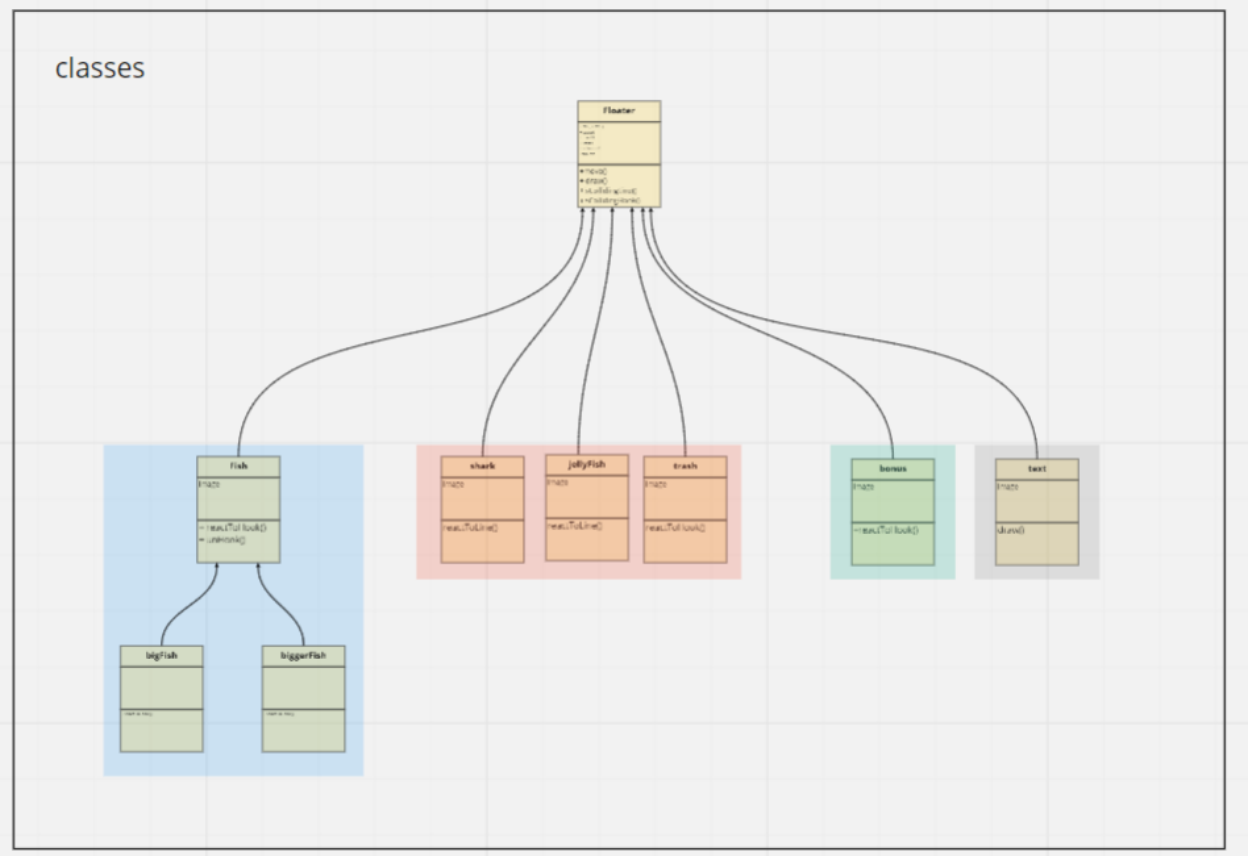
interface

- drawPause()
- drawStart()

Algorithmic Plan:

Collisions: use inheritance and double inheritance to optimize code and allow for testing of specific collisions based on the class the collider is (see diagram for more).

- All moving things that spawn in the water are floaters
- Fish can be caught and use the onLine parameter
    - Big fish only bite when a regular fish is onLine
    - Bigger fish only bite when a big fish in onLine
- Enemies are floaters that have specific effects on line and fish (not a specific class)
    - Jellyfish can shock line and scare the fish off, breaking the hook
    - Trash scares the fish off the line but does not break hook
    - Shark eats the fish and the hook
- Bonuses are floaters that can have a positive effect on the game
    - Add life (hooks remaining)



Progressive Difficulty: increase the quantity of floaters spawning and types based on score/time

The ratio of spawning types (can be done with random.choice) (approx.)

- Wave 1
  - Nothing 20
  - Fish 50
  - Trash 30

- Wave 2
  - Nothing 10
  - Fish 30
  - Trash 10
  - Shark 20
  - Bigfish 20
    - ** begin increasing fish speed and spawn rate

- Wave 3

    - Nothing (0 or 10)
    - Fish 30
    - Trash 10
    - Shark 20
    - Bigfish 20
    - jellyFish 10
    - biggerFish 10

- Wave 4
    - Continue gradually increasing the spawn rate
        - Decrease the amount of time required to spawn
            - Until n%k == 0 all the time (when k is 1)
            - n is the timePassed variable defined in app
            - K is the spawnRate variable defined in app
    - Change ratio of floater types spawning based on types of fish caught
        - Add enemy chances to spawn if enemy avoided count exceeds a certain amount
    - Continue Increasing the speed of floaters
        - Every X fish caught increase the speed range(based on how well the player is doing)
    - Continue increasing the spawn rate
        - Every X fish caught (based on how well the player is doing)
    - Spawn ratio
        - Spawn ratio becomes dynamic
        - The more of a specific enemy that crosses the screen successfully (by a set factor using mod), it is more likely for that enemy to spawn (sharks and jellyfish)
        - The more the player gets attacked by a specific enemy, the less likely it is for that enemy to spawn

The threshold for a new wave starting:

- Increases exponentially every time a new wave starts
- Bonuses (extra lives) spawn after the final wave is beaten and thereafter

Timeline Plan:

- ❑ **Monday (6/22)**
    - ❑ finalize project proposal
    - ❑ finalize storyboard

- ❏ Preliminary code
    - ❏ Classes
    - ❏ Movement features
    - ❏ Catching features
- ❏ TP1 8 pm

❏ **Tuesday (6/23)**

- ❏ Working demo
- ❏ Enemies, big fish, etc.
- ❏ Update design docs
    - ❏ TP2 Update (design changes made/foreseen)

❏ **Wednesday (6/24)**

- ❏ User features
    - ❏ Clean interface
    - ❏ Start game screen
        - ❏ Play
        - ❏ graphic
    - ❏ Pausing
        - ❏ Screen
        - ❏ Restart
    - ❏ End game screen
- ❏ Make objects look like what they are
- ❏ TP2

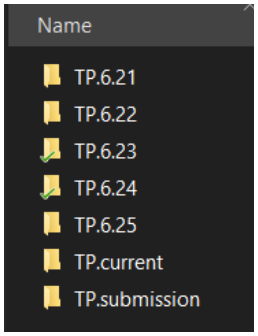
❏ **Thursday (6/25)**

- ❏ Finish all features and code work
- ❏ Plan project demo
- ❏ Record project demo

❏ **Friday (6/26)**

- ❏ Assemble codebase
    - ❏ All python files and any other files (images, music files, etc.)
- ❏ Assemble design documentation
    - ❏ Readme.txt
    - ❏ Demo-video.txt
        - ❏ Link to demo
    - ❏ Project proposal
    - ❏ storyboard
- ❏ TP3 Update
    - ❏ TP2 Update (design changes made)
- ❏ TP3 3pm
- ❏ Live demo plan

Version Control Plan:
The current file is saved in TP.current while copies of working or progress code are saved into the dated folders and automatically uploaded to the google drive.



TP2 Update:

Revised algorithm: increasing the difficulty as the game is played

- The ratio of floater types spawn was tweaked to be more challenging
- The threshold for starting a new wave was revised to be a ratio of fish caught and time
- Dynamic speed
    - Range of floaters speed is increased as fish are caught
- Final wave changes
    - More specific enemy instances spawn if you avoid more of them (ex. If you avoid more jellyfish, more jellyfish will spawn to challenge you!)
    - Less specific enemy instances are likely to spawn if you are hit by them
    - All floaters get continually faster (as you catch fish)
    - More floaters spawn (as you catch fish)
- Increased spawn rate only occurs after the second round

Bonus revision

- Instead of spawning like other floaters, a bonus (i.e. a 1-up) spawns for every ‘wave’ passed after wave 4 (meaning as the wave threshold is increased and met a 1-up will spawn
    - Supports ‘endless’ wave concept

Revised floater type balance:

- Sharks should be faster than regular floaters!
- Speed is randomized between a now dynamic range

MVP breakdown

- Functional game (reeling in fish, keeping track of a score, ending the game at some point)
- Tracking user stats and having an increase in difficulty/procedural generation based on those stats
    - fish caught - changes speed, spawn rate
    - score - changes the ratio of spawn and spawn types
    - enemy count - increases enemies based on which ones are avoided
        - * enemies not avoided become less likely to appear
- Fish, enemies, and obstacles (at least two of each)
    - one obstacle, three enemies, three fish, one bonus
- Good GUI
    - Stylize as fish
    - End game
    - Track score
    - Track lives
    - Pause
    - Wave messages

Things done, not listed:

- Floater bonus class: increase lives (hook count)
- Loose hook feature
    - Collisions (fish, enemies, bonuses) only take place if there is a hook on the line
- Rehook line at the surface
- edge/margin resolve to avoid lag and be unnoticeable
- Class hierarchy (advanced collision testing, double inheritance)
    - Catch a smaller fish in order to catch a bigger one
    - Line collision vs hook collision
    - Fish release (scare the fish) vs. eat (make the fish disappear)
- Stylize
    - Draw scenery
    - Draw hook/player
    - Animate fish
        - Animate on line
    - Fish orientation/reel/direction

Next steps:

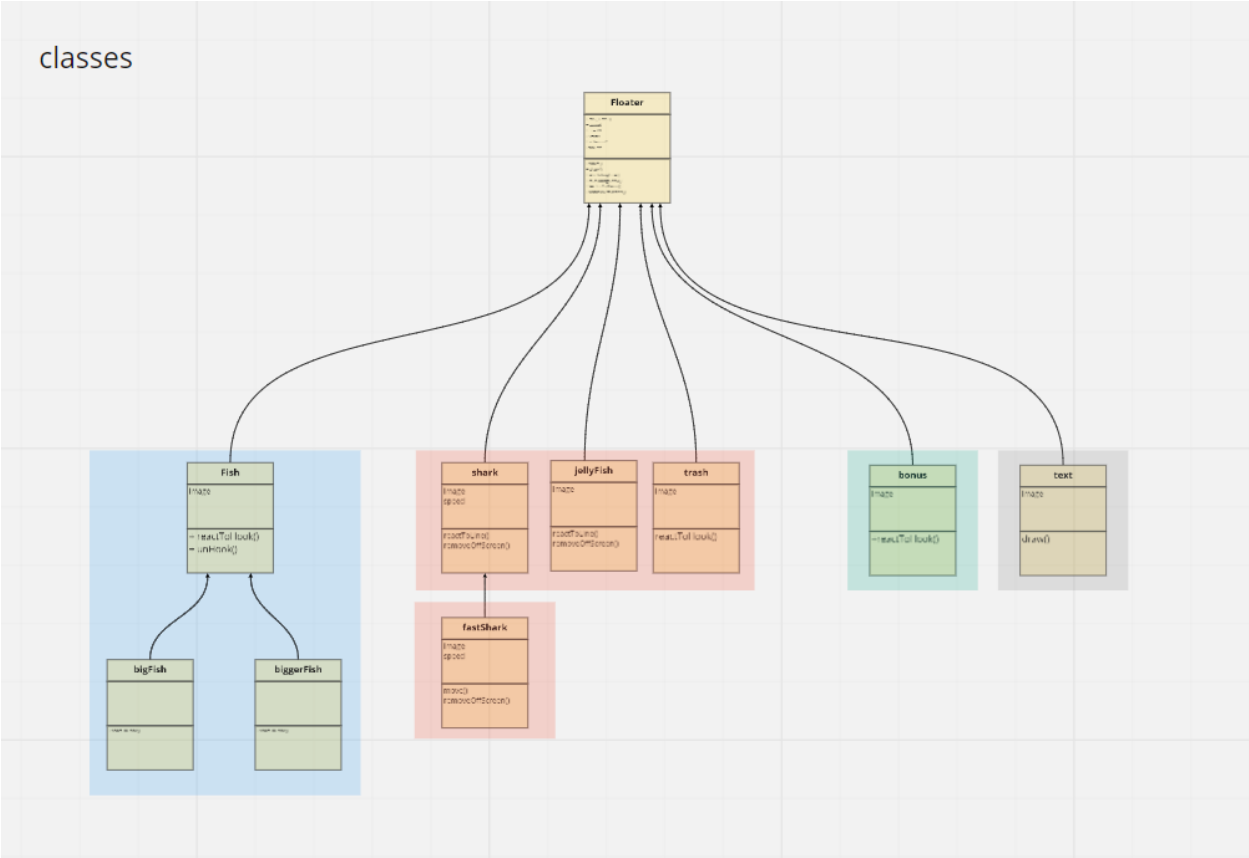
- Modal apps?
- Physics - slower for bigger fish?
    - Pull up on line

- Intensity of reel
- When fish collides hook,
- Click to catch

TP3 Update:

Revised class code organization/methods:

- New dynamic enemy: fast shark
  - As a subclass of shark, this enemy can hone in on a fish that is currently hooked
  - Will only attack if it can see a fish on the line
  - Can only see in front of itself



- Modal apps
  - gameMode
    - Regular hooking of fish and start parameters
  - clickToPlayMode
    - Implements a difficulty of catch feature
  - homeScreen
    - Has buttons that allow the user to choose the game mode

                - Toggle program - only one can be selected and when one is selected the other is deselected
            - Button for instructions
            - Animated graphic
            - title
- pop-Ups
    - Instructions
        - Displays how to play info has exit button
            - Enemy types and what they do
                - Trash
                - Shark
                - Fast sharks
                - jellyfish
            - Fish types and point values, using small fish as bait
                - Fish
                - Big fish
                - Bigger fish
            - Click to catch
        - Can be exited by unpausing when playing the game
        - Only accessible from the home screen of pause popup
    - Pause screen
        - Provides the user the options to check how to play info, restart the game, or go back to the home screen
- Button class
    - New file ‘buttons.py’
    - Buttons are objects that can be drawn and clicked
    - Buttons are graphic ways of communicating with the user
    - Can be used as a toggle and features a highlight representation
- Refined code org.
    - Functions moved into secondary ‘functions.py’ and simplified into a helper to avoid duplicate code or allow code to be accessible across modes
- clickToPlayMode
    - Timing bar appears when a fish is near the hook
    - If the fish is clicked when the timing bar has its arrow in green, hook the fish
    - The size of the green (represents hook ease - the smaller the harder it is to hook the fish as it
    - Size of green is relative to the size of fish and speed of the fish
        - The green represents the likelihood of hook a fish
        - As the game progresses through the waves, the fish get faster. Similarly, this ‘green’ will get progressively smaller
        - The size of the green is based upon the ‘weight’ of the fish and its current speed
        - You must click the fish to hook it

    - New variables in fish class and app to prevent hooking a fish twice and control the game modes
- Algorithm revision
    - Dynamic spawn location
        - If the player catches more fish near the surface of the water, it is more likely a fish will spawn lower to the bottom (as well as the opposite)
        - Spawn range decreases until it reaches a minimum width (a quarter of canvas size in this case) after that, the zones increase but the one where fish are not being caught increases faster, keeping the same logic while the fish 'condense' and 'spread out' when trying to avoid the hook
    - Dynamic fish spawn type
        - If the player catches a fish the fish of the next weight tier is more likely to spawn
        - Careful to maintain a ratio of fish>bigFish>biggerFish
        - Only in effect after wave 3 (similar to dynamic enemy spawn ratio)
- Highscore
    - The game now keeps track of a high score and player name in a local file
    - The game also asks the player for their name if they get a new high score
- Expert mode
    - Game mode introduced that skips the first introductory waves of enemies
    - Reworked interface to accommodate the new option

Final Term Project Main Features:

Game Modes
- Amateur
    - Introductory waves
    - A fish is caught automatically when it hits the hook
- Beginner
    - A similar progression of difficulty/introductory waves as an amateur
    - Must click at the right time to hook a fish
    - A fish is more difficult to hook when it is larger or faster
- Expert
    - No introductory waves
    - The game gets progressively harder instantly

Progressive difficulty
- The first two modes incorporate a 'learn to play' first portion of the game
    - Fish and enemy types are gradually introduces based on how the player is doing (score)
- After these initial waves, the game becomes progressively harder
    - Specific enemy types are more likely to spawn when avoided and less likely to spawn if they break your line!

    - Fish are progressively less likely to spawn (compared to enemies) based on how many specific enemy types are avoided
- Based on the fish caught, the sea creatures move faster
    - This rewards players who gamble their fish and wait for bigger fish as the speed is quantity based, not score base
- When a fish is caught, the next weight tier of that fish is more likely to spawn
- Depending on where fish are being caught fish will avoid that area
    - If a lot of fish are being caught near the surface, fish are more likely to appear deeper in the water. The opposite is true if more fish are being caught near the bottom!
- When playing in beginner or expert mode the difficulty of hooking a fish increases as you catch fish
    - This is based on the fish's weight and the speed it is moving when it is near the hook

Types of spawnings

- Things to catch
    - Regular fish
    - Big fish
        - Only bites the hook when a regular fish is on the line
    - Even bigger fish
        - Only bites the hook when a big fish is on the line
- Things to avoid
    - Trash
        - Will scare a fish of the line if it hits the hook
    - Shark
        - Eats the fish and breaks the hook
    - Hungry shark
        - Similar to a regular shark but will chase a hooked fish
    - Jellyfish
        - If it hits the line will shock the hook and fish off
- Bonus
    - Extra life
        - Rare, usually spawns in the drift of jellyfish tentacles

Secondary features

- User interface
- Animation
- leaderboards