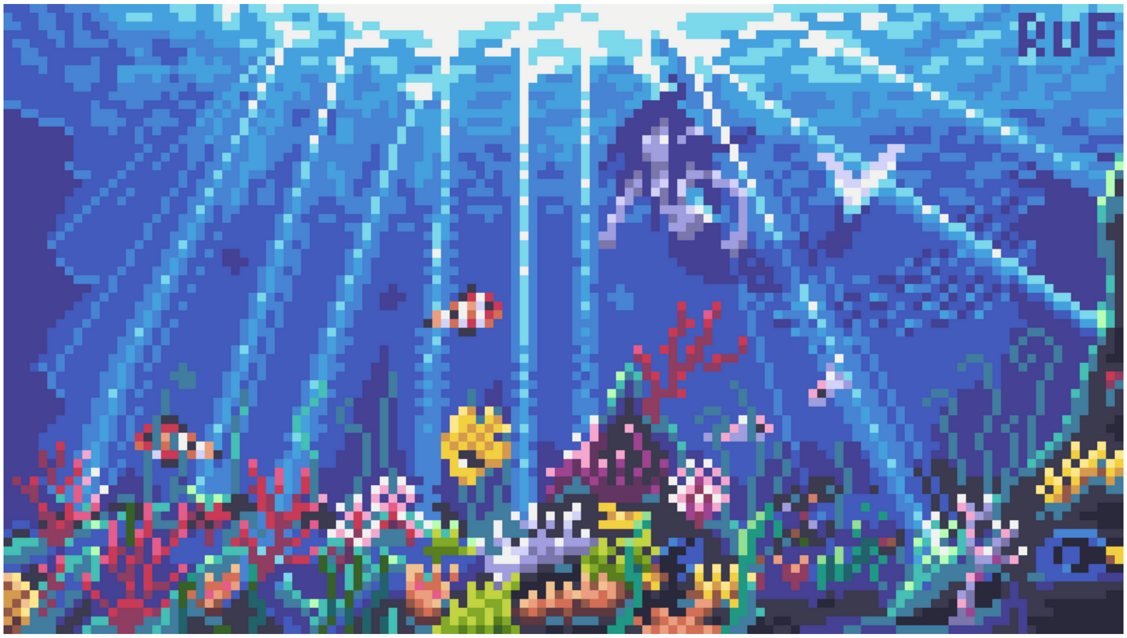
HOMEWORK 03

Extended Mode 3 Game



Purpose: To build a more complex game in Mode 3 to further your understanding of: structs, arrays, and pooling.

Instructions:

In this homework, you will be making a more complex game in Mode 3. **This must be something different than what you implemented for HW02 or any of the labs. Further, the game should relate to the theme of *Underground or Underwater.*** Examples of games you can use for inspiration are at the bottom of this PDF.

The design of this game must be more complex than *catching/dodging falling boxes* or basic *Pong*. We are leaving this one open ended so that you can push yourself creatively and see what fun game you can come up with! You are free to choose a game similar to the examples we provide (as long as you tie it to the Underground or Underwater theme), but you are also free to create your own original game (if you choose this option, ***please speak to a TA first*** so that we can ensure it is on the expected difficulty level).

Requirements:

Your *game* must use the following:

- At least **one struct**
- At least **one array**
- **Object pooling**
- A **state machine** including at least the following states:
  - Start,
  - Pause,
  - Game,
  - Win and/or Lose.
    - It is ok if your game is a survival based game, and therefore only has a lose state!
- You must be able to **navigate between the states** in some way (e.g. pressing the button START while on the Start state takes you to the Game state, winning the game while on Game states takes you to the Win state, etc.)
- At least **four moving objects**
- At least **four buttons** used for input
- **Collision that matters** (i.e. *something* must happen whenever two different objects touch each other -- think bullet+enemy, enemy+player, tomato+face, etc.)
- A **README.md** file
  - An instruction manual (of sorts) that tells a player how to play your game, including things such as:
    - What each button does (controls)
    - How to navigate your state machine
    - How to win and/or lose your game
    - Anything that is buggy or not completed
  - If you have little to no experience with Markdown syntax, here is a cheat sheet and a more thorough explanation.
    - There are plenty of other resources online! Feel free to reference as many as you like
  - We are not requiring that you fully explore all the possibilities that Markdown provides for formatting, however, we will award up to **2 points of extra credit** on this assignment for students who have clearly put effort into their README!
- Only a **minimal amount of flicker**
- Connection to the **Underground or Underwater theme**

Your *code* must have the following:

- At least **three .c files**
- At least **two .h files**
- Good organization (see tips below)
- Meaningful comments

Tips:

- **Start early**. Never underestimate how long it takes to make a game! We give *much* more time for HW03 than for HW01 and HW02 because ***we expect you to do a lot more work***.
- When splitting code between multiple files, put code that will be useful in multiple games in your HW03Lib.c, and code specific to this game in main.c or other files. Those other files should be specific to a concept (response to collision, a specific state of the state machine, etc.).
- Organize your code into functions specific to what that code does. **Your main method should not be very long at all!**
- Having update() and draw() functions that you call directly in main() or within another function being called in main() is helpful.
- Make sure the order of calling your functions takes into account waiting for vBlank at the correct times to minimize flicker.
- Build upon your HW02Lib.c and HW02Lib.h files that you might have created for HW02!
- Feel free to reference the Recitations files on Canvas to review concepts.
- Feel free to reach out to the TAs if you have any questions!

Submission Instructions:

Ensure that **cleaning** and building/running your project still gives the expected results.

Please reference previous assignments for instructions on how to perform a "clean" command if you need clarification.

Zip up your entire project folder, including all source files, the Makefile, and everything produced during compilation **(including the .gba file)**. Submit this zip on Canvas. Name your submission **HW03_LastnameFirstname**, for example:

“HW03_DaisyPrincess.zip”

It is your responsibility to ensure that all the appropriate files have been submitted, and that your submitted zip can be opened and everything cleans, builds, and runs as expected.