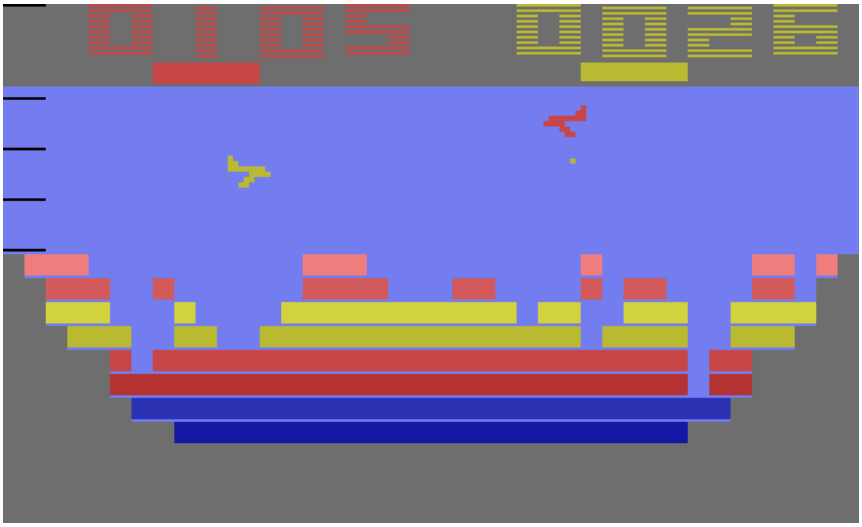
HOMEWORK 02:

Canyon Bomber



Purpose: To build a simple game in Mode 3 to further your understanding of: inputs to the GBA, collision detection and reaction, C basics, and drawing in Mode 3.

Instructions

For this homework, you will create a simplified version of the game *Canyon Bomber* in Mode 3. If you are unfamiliar with the game, you can watch this playthrough of Canyon Bomber. Additionally, you can play the provided `Example.gba` file, a complete version of the project which includes the base requirements and extra credit considerations. Review the **Base Requirements** section of this document for an explicit list of what we expect to see. The basic idea of our version of *Canyon Bomber* is as follows:

Base Requirements

Gameplay

Your game *must* have the following:

- The player should always move horizontally across the top of the screen, switching directions every time it reaches the edge of the screen.
    - The player should be at least 4x8 pixels in size.
    - The player rectangle should **not** wrap around the screen and continue moving in the same direction (this may happen if you move too far in either direction).
- While the SELECT button is held, the player's movement speed should be noticeably increased.
    - After releasing the SELECT button, the player's movement speed should return to normal.
- When the B button is pressed, the player should be able to drop a bomb from their current position.
    - The bomb should be no larger than 4x4 pixels in size.
    - The bomb's y velocity should cause it to move downwards.
    - The bomb's x velocity should match the player's x velocity **at the exact time it is dropped**. If the player speeds up, slows down, or changes direction while the bomb is falling, the bomb's x velocity should remain the same as when it was initially dropped.
    - The bomb should be completely erased from the screen once it reaches the left, right, or bottom edge of the screen.
    - **The player should not be able to shoot a new bomb if there is already a bomb falling on screen**. Once the current one has been erased, the player should be able to fire again.
- The map should include **at least 15** blocks of destructible terrain.
    - Each terrain block must be at least 8x8 pixels in size and should be positioned in a way where it is possible to hit them with a bomb.
    - If the bomb collides with a terrain block, that terrain block should be erased from the game. The bomb should **not** be erased after the collision and should be able to collide with multiple terrain blocks if they are in the bomb's path.
- The player should win the game once they have destroyed every block of terrain.
- The game should enter an end state once the player wins.
    - It must be clear that the game has ended; the player should not be able to play indefinitely. Preventing the player from moving or shooting is okay here.
- Only a minimal amount of flicker.

Code/files

Your *code* must have the following:

- **You must use the provided HW02 scaffold!**
- Multiple `.c` files.
- At least one new `.h` file that was not included in the original scaffold.
- A **`README.md`** file.
    - This should contain an **instruction manual** that tells a player (but most importantly, your grading TA!) how to play your game.
- Good organization (see **Tips** section below).
- Meaningful comments.

Extra Credit Considerations

Gameplay Mechanic - Falling Terrain

You can earn an extra 5 points by fully implementing this feature!

- Ensure that your terrain blocks are arranged in multiple rows stacked on top of each other.
- A terrain block should **not** fall downwards under either of the following conditions:
    - The block being checked is at the bottom of the canyon.
    - There is a terrain block directly underneath the block being checked.
- If neither of the above conditions is satisfied, the given block should fall directly downwards until at least one of them becomes true again.

This mechanic is deceptively difficult to implement and can cause game-breaking bugs if not done carefully! If you choose to attempt this extra credit, we **strongly recommend** finishing the base requirements first and saving a working version of your project beforehand.

Personal Flair

By adding extra features of your own, you can earn up to 5 extra points! If you choose to implement features beyond the scope of this assignment, **add a new section to your README.md file that details your additions in order to receive credit**. Some ideas:

- Animate the plane (player) and/or the bomb!
- Add lots more terrain blocks (without introducing flicker).
- Make the game look extra nice!

If you have questions about how many points a specific flair can earn, please ask Luke about it!

Tips

- **Start early.** Never underestimate how long it takes to make a game!
- For collision code, draw pictures. Graph paper is your friend.
- When splitting code between multiple files, put code that will be useful in multiple games in your lib file (e.g. `gba.h`), and code specific to this game in `main.c` or other files. Those other files should be specific to a concept (collision, movement, etc.).
- Organize your code into functions specific to what that code does. **Your `main()` should not be very long.** Use helper functions!
- Having `update()` and `draw()` functions that you call in `main()` is helpful.
- **Think carefully about which objects in your game have to be redrawn each frame and which do not**. Mode 3 only gives us time to edit about 2100 pixels in the video buffer per VBlank period, so minimizing setPixel usage is key!
- Make sure the order in which you call your functions takes into account waiting for VBlank at the correct times. This will help you minimize flicker.

Submission Instructions:

Ensure that **cleaning** and building/running your project still gives the expected results. **Please reference the last page of previous assignments for instructions on how to perform a "clean" command.**

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 HW02_LastnameFirstname, for example:

> “HW02_DelmanHoward.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.