# HOMEWORK 01


# Looping Animation

**Purpose:** To use a vertical sync function and set pixel to further your understanding of:

- VBlank and HBlank
- Drawing and Erasing pixels in Mode 3
- Working with an infinite while loop

---

## Instructions:

Before you can make games, you need to get a solid grasp on displaying imagery to the screen. To that end, you will create a looping animation of several frames in Mode 3. You will also implement button controls to switch between animation frames.


## Animation

Your animation should fit into one of the following themes:

- **Robots / Machines**

  **OR**

- **Video Games**

We plan to select the best submissions and make a compilation of them to be shown during recitation and potentially posted to the class Canvas page. If you don't mind your submission being potentially put into that compilation, **please comment on your submission saying something like "I'm okay with my submission being shown to the class." If you want to be anonymous, please include that in your comment as well by saying something like "I want to remain anonymous."**

Submissions will be selected on the following criteria:

- Creativity
- Application of one of the two themes (we're pretty lenient here)
- Complexity

**There is no prize for being selected other than pride.** You can get a perfect grade on this assignment by meeting the criteria in the Requirements section. The primary purpose of this assignment is to help familiarize you with drawing in Mode 3 — making something artistically pleasing is secondary!

## Button Controls

You should implement the following controls:

- **A:** Advance to the next frame.
  - If there is no next frame, this button should cause the program to return to the first frame in the sequence.
- **B:** Return to the previous frame.
  - If there is no previous frame, this button should cause the program to show the last frame in the sequence.

## Clarifications

- A and B refer to GBA buttons! They are not mapped to the A and B keys on the keyboard by default!

- For advancing and returning through frames, think of the current frame as a number ranging from [0, <number of frames>). Pressing B should increment that number and change it to 0 if the number leaves that range. Pressing A should decrement the number and change it to the upper end of the range if the number becomes negative.
- You ***cannot*** make an "animation" that is a bouncing ball or anything similar! We will not watch your program for 5 minutes waiting to see if it technically loops or not! Brickless Breakout will receive **0 points**!

---

## Requirements:

- Follow **good code architecture**
    - Not a bunch of set pixel calls in main
- The animation must **loop forever**
    - The animation does not necessarily need to end where it started. You can just cut back to the first frame of the animation after the last frame. For example, you could have a cat jump off of a table, then have the animation restart with the cat back on the table again.
- A **vertical sync function** must be used
- There must be at least **four meaningfully different frames** of animation
    - This means the shape or position of something in your animation must be different in each frame — just changing the color alone is not enough!
- There must be at least **three different colors**
- There must be a **minimal amount of flicker** (ask a TA if you are unsure!)
- Your submission must relate to **one of the two themes**
    - We will be lenient with this!
- Pressing the **A and B buttons** must move to the previous and next frame, respectively

## Tips:

Consider writing a separate function for every frame and calling these functions in an endless loop. Or, write a procedural animation that resets itself so that it also looks like it loops. Either is fine, and it's up to you how you want to organize your code, so long as there is meaningful organization.

- If you are having trouble with flicker, try drawing fewer pixels per frame. Don't draw a pixel unless you actually need to!
- If you need help getting started, drawing out a few frames on a 240x160 canvas in a simple paint editor or on graph paper may help.
    - You can use Microsoft Paint:
        - Open a new project, select "Resize"
        - Select "By Pixels"
        - Uncheck "Maintain Aspect Ratio"
        - Set "Horizontal" to 240 and "Vertical" to 160
- Make sure your moving elements/shapes don't "wrap around" the GBA screen. In other words, check your boundaries!

---

## Submission Instructions:

Ensure that cleaning and building/running your project still gives the expected results. Please reference the last page of this document 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 Lab01_FirstnameLastname, for example:

"Lab01_EdsgerDijkstra.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.

Please leave a comment on your submission which addresses **both**:

- Whether you are okay with your submission being shared with the class
- Whether you wish to remain anonymous or to have your name shared along with the submission

Example: "I am okay with my submission being shared with the class. I do not wish to remain anonymous."