Mini-Project 4 Write-up and Reflection

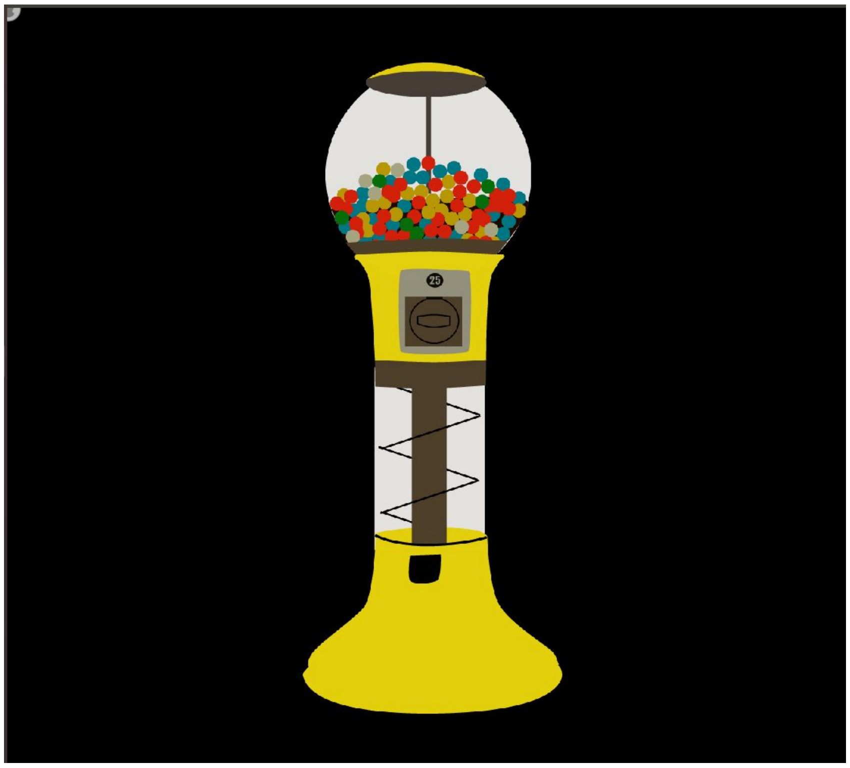
Authors: Emma Mack and Gabriella Bourdon

Project Overview

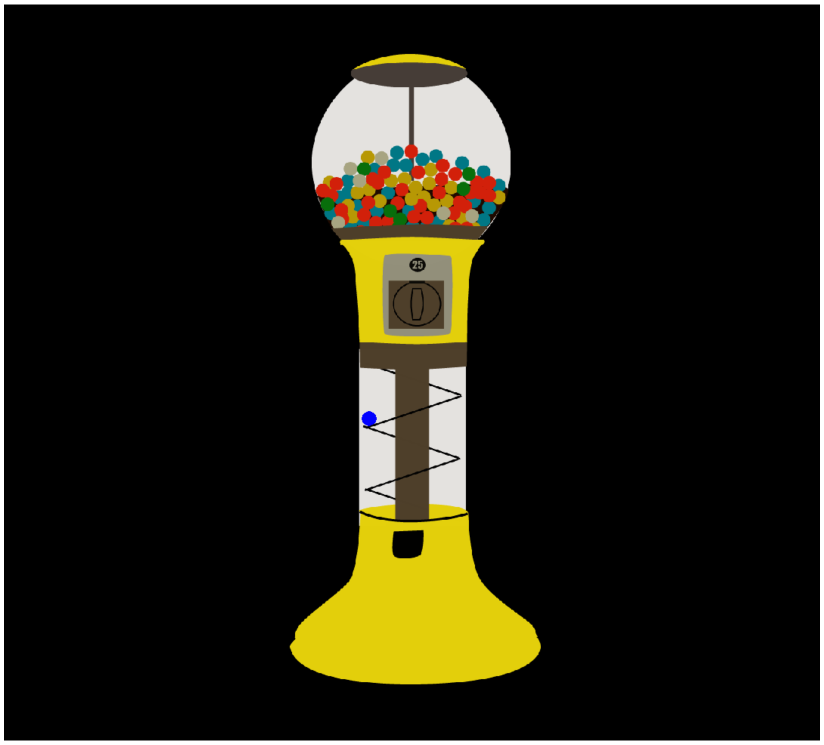
The idea of our project is to have an interactive animation of a gumball arcade machine (part art, part game), with both the quarter dispenser and gumball retrieval serving as event-driven points of interaction. From the user's perspective, they will first see an image of a gumball machine, click on the dispenser to insert a quarter into the slot (that is currently following their mouse cursor), watch the gumball roll down the machine in a zig-zag pattern until it reaches the bottom, in which the user must click again for the gumball to turn into an image of an animal.

Results

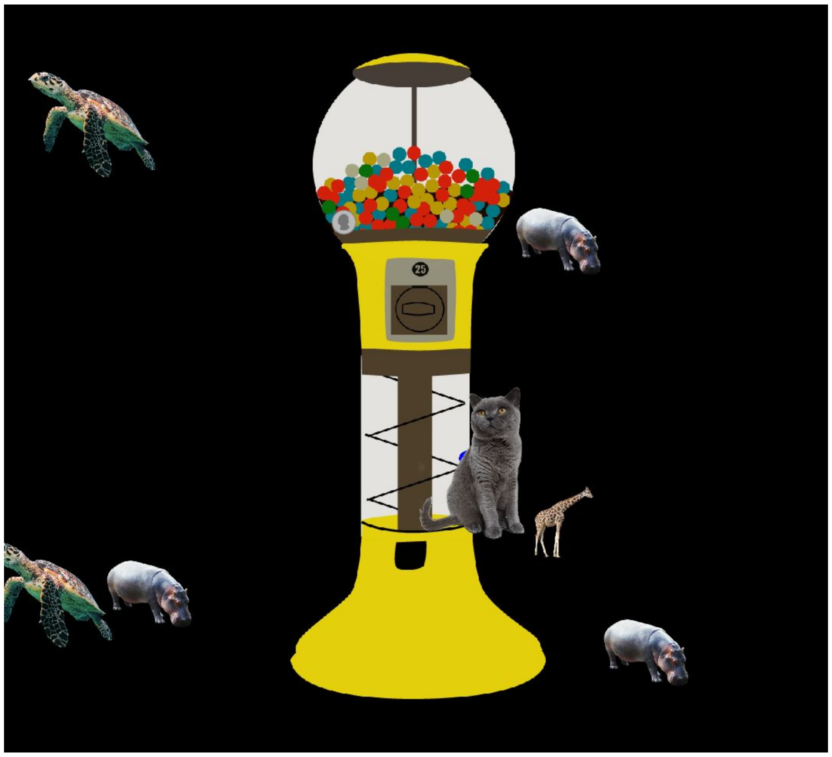
The final product begins with a drawing of a gumball machine. A quarter follows the mouse from the start of the program. If the mouse scrolls over and clicks within the gray dispenser box, the quarter darkens in color and triggers a gumball to roll down the chute. Any clicks outside the box will be out of range, and the animation will not start.



Initial screen. Quarter follows mouse.



A click on the dispenser turns the knob and sends a gumball rolling down the chute.



Each gumball contains a surprise, which build up on the screen as you release more and more gumballs!

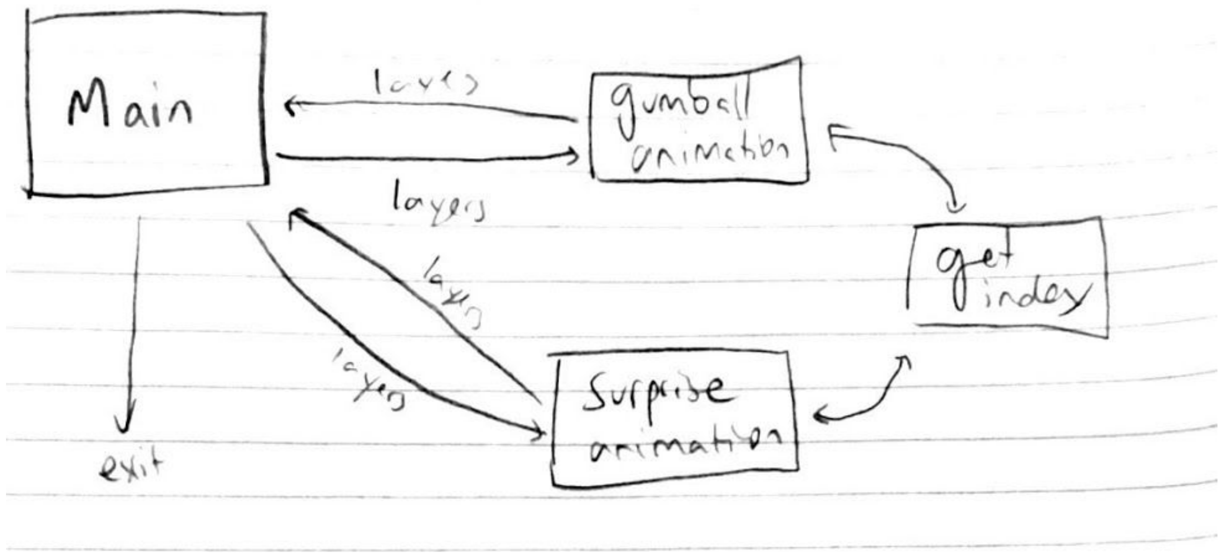
When the gumball lands in the bottom of the machine, another click sends it flying in a random direction, and it then expands into a random animal-themed surprise and plays a celebratory sound.

Implementation

Each object that is displayed on the screen: the layers of the machine, the gumballs, the quarter, and the surprises, are their own distinct classes. They are contained within a list of layers which can be re-ordered to reflect which elements are on top.

The main loop draws the layers in order each clock cycle. It also checks for clicking in certain areas, which triggers the recording of t_start and changes boolean flags so that an animation starts playing. If an animation is triggered, an animation function is called, which modifies the layers slightly each clock cycle. The animation functions keep track of time, and what changes to make, by calculating the time since the start of the animation.

We had to make many design decisions in the creation of this project. An example of a decision that was successively iterated upon is how to encode the information in the layers. Each image has its own coordinates, a position within the layers, and in the case of the gumball, a color. Originally, layers list was its own object, but it didn't need to be, since it has the same functionality as a normal list. Then, the location of each element was encoded in tuples in the layer list. The code got unreadable at that point, though, so we made each layer an object.



Reflection

The scope was very appropriate. We finished without too much stress, but we were definitely working hard the whole time and learning a lot.

We tested each change by playing through the game and seeing what errors came up. Doctests would not have made sense for the most part, since the functions were not outputting text.

Emma learned how to make high-level design decisions to maximize efficiency and code clarity such as what to put in what classes and what functions to call where. She also learned how to debug code where the actual bug can be several functions removed from where the error message is. Gabriella learned how rotations in animation are structured after working on the dispenser rotation, more functionality with classes as well as interaction between imported images, layers, and animation. However, it would have been good to know how to use modules (different .py files) and import them so as to minimize merge errors from the beginning. If we had to do this project again, we might consider utilizing class inheritance. Something that aided our design process was observing our friends playing our animation to see how they expected the animations and clicking infrastructure to operate. That is why the color of the quarter changes - to indicate that when the quarter darkens, the user should click.

We divided up work by category -- Emma mostly did the animations, and Gabriella handled the clicking infrastructure and getting the quarter image to follow the mouse. We met pretty frequently to check in and make sure we were both on the same page. We ran into a few problems wherein one of us had done more coding in python before, and tended to run ahead with structural decisions without consulting the other first. We were able to reach a middle ground eventually and teach each other what we needed to know. Next time, we should, on a piece of paper, make structural decisions together before starting to code.