Introduction

Title: Ant Simulation

Group Members: John Wiesner

Overview:

Many hours have gone into this project and in return a fully fleshed out game with a downloadable executable was produced. This project is based on the concept of Langton’s Ants: if an ant was the size of a pixel, the ant would turn and move left/right dependent on what color pixel it was on. This creates a pattern like path on a screen. Taking the interactive pixel elements from a game called The Powder Game and merging them with the Langton’s Ant concept I developed the game/simulation that is presented in this documentation.

Application design:

The only 3rd party API used to create this application game was Pygame. My self-goal and challenge for building this game was to make the entire game from scratch meaning using no other 3rd party code, graphics, nor Pygame examples (Coping a tutorial and changing a few things). Setting these rules provided a more open-ended development and game design process that would otherwise be influenced by a Pygame tutorial or non-original game.

There are three main components of an enjoyable playable sandbox game: Simplicity, Creativity, and Learnability. Developing a program that satisfying these requirements could take many years to prefect. Therefore, I chose to take a step back and look at what other popular sandbox games have done and tried to adapt them to my constraints.

Before placing buttons, text and boxes over the screen, I decided that a simple left sided menu would be the best for this game design (Similar to The Powder Game layout). All the buttons would be located on one side and there would be no need for searching the screen to find the correct button.

Below is a final piece of art concept for the menu layout:

This is what the menu looks like in the finished game:

Having this simple layout saved time when coding because there were no adjustments needed and code did not need to be rewritten. All buttons are organized into categories and have corresponding colors to make it simple for a user to distinguish functionality.

The learnability of this game applies to the way the ants interact with each other on screen. Each ant has its own properties and it is up to the user to discover what they are. There is no in-game description what to do nor how the ants work. This curiosity is what draws users’ attention and makes them want to keep playing. For the minds who want to know more about each ant’s properties there is a help button that brings you to a website showing what each ant is capable of.

This game provides its users with an opportunity to be creative. The user is in control of how many ants are on screen, which ants live, which ant paths should stay on the screen, how fast the simulation should run, and even how some of the ants can move. Using the provided existing ants a user can create their own ants by changing the ant step properties. Many unique patterns can be created by doing so creating designs like the one below:

Image of weird ant here

Overall this Python application/game meets the three previously set requirements defined above in a respectful and elegant way. This hopeful provides all users with many hours of enjoyment and delight.

High-level diagram of the structure and flow of the application:

Diagram here

1). Provide a description and overview of your project. Provide a description of your application design and details about the application software, possibly including a high-level diagram or description of the structure and flow of the application. (1-2 pages)

What was accomplished:

I knew very little going into this project: no Pygame experience nor how to use python classes properly.

Talk about

The different classes made

Game art

Sounds

Toughest parts to code

Successes to those parts

Most enjoyment parts

2) Project Results: Include a description of what you accomplished. Include specific successes and pitfalls in your project experience. Provide specific outputs or screen shots as part of the explanation of your results. Provide suggestions for future potential improvements. (1-2 pages)

**Division of Work:**

Even though I worked alone I still divided the project into phases.

Phase 1:

Create a design on what the game should look like and what the game should feel like. This is thinking about what the user experience would be if certain features would be included.

Phase 2:

Create the necessary Pygame objects to have a functional game. This includes: mouse over buttons that enlarge when the user hovers over them, moveable images on screen that also scale properly to the window size, and clickable and editable text boxes that a user can edit.

Phase 3:

Slowly create all the game logic and simulation effects. The core of the game.

Phase 4:

Polish the game layout. Making sure the are no broken buttons and all features are displayed and are functional. This also ensures the game is able to work in full screen and scales properly.

Phase 5:

Add game art to make the presentation better and the user feel like the game is a true game and not just a proof of concept.

Phase 6:

Add sound a music

Phase 7:

Test the game like crazy to discover any bugs and fix them.

Phase 8:

Game release

4) Bibliography:  Provide appropriate references for all articles, websites, and all code libraries you used in your project. (1 page)

**References:**

**Topics that gave me inspiration:**

Langton's ant: <https://en.wikipedia.org/wiki/Langton%27s_ant>

Powder Game: <https://dan-ball.jp/en/javagame/dust2/>

**Libraries\Programs:**

Pygame: <https://www.pygame.org/news/>

Cx\_Freeze: <https://anthony-tuininga.github.io/cx_Freeze/>

**Tutorials/Code/Assets:**

Pygame Tutorials: <https://www.youtube.com/watch?v=K5F-aGDIYaM>

Interactive Textbox: <https://stackoverflow.com/questions/46390231/how-to-create-a-text-input-box-with-pygame>

Background music: <http://soundimage.org/>

**Inspirations for ant types:**

Fire ant:

<https://en.wikipedia.org/wiki/Fire_ant>

Zombie ant:

<https://en.wikipedia.org/wiki/Ophiocordyceps_unilateralis>

<https://www.wikihow.com/Walk-Like-a-Zombie>

Crazy (Acid) ant:

<https://en.wikipedia.org/wiki/Rasberry_crazy_ant>

Add link to Repo or Dropbox:

Download game exe: <https://mrjohnweez.weebly.com/ant-simulation.html>

I have always loved the [Powder Game](https://dan-ball.jp/en/javagame/dust/) and [Langton's Ant](https://en.wikipedia.org/wiki/Langton%27s_ant) concept so I wondered if it would be possible to combine the two. I never found the perfect engine that has an easy way to manipulate pixels on the screen until I started programming in python. I decided to make a test simulation.  
  
This simulation started out as a basic proof of concept and was a way for me to learn Python and the API Pygame for a future class project at the end of a college semester. Overtime, I just added features here and there while I was studying in college and it began to grow into a game/simulation like program. Towards the end of the semester I was able to use my basic simulation for my final Python project! This allowed me to pour way more time into it then I had before.

It took many hours to create this simulation; mainly due to learning Pygame as I went but also due to the fact that I wanted to create everything from scratch (Not using 3rd party code other than Pygame/Python APIs). This includes all the sound effects you hear in the game which were made using my laptop and stuff around my desk. The music was downloaded from an opensource website (Credited in game). All art was created and is owned by me.