Kathryn Curley
Daniel Gilbert
Rebecca Loredo
Daniel Shats

CAP3032 Final Report - Hot_budr Snake

Overview

Essentially, hot_budr_snake is our take on the classic snake game. It's a twist that takes all the fun of chasing squares with a pixelated snake, and makes you chase good ol' hot butter instead. But that's not all. In addition to being on the lookout for hot_budr, you need to be careful to avoid the dangerous instances of not_budr located on the game map. Also, running into the walls or colliding with yourself will put an end to your game as well. Cheerful 8-bit music will play in the background during the menu and sound effects will play during the game when you eat butters or die. Moreover your current score will be displayed in the upper corner while playing.

Our game will contain 3 seperate levels: easy, medium and hard. As levels increase in difficulty both the initial speed and number of obstacles increase as well.

Easy mode is based off of the typical snake game. There are no obstacles on the playing screen, only hot_budr that should be collided with in order to grow your snake and in turn increase your score. Beware, the more budr you collect the faster the game play will get and the more difficult the game will become. Avoid the playing field edges, they are considered to be walls and will cause you to lose the game if collided with.

Medium mode is where things begin to get interesting. Here not only will you be challenged with the regular difficulty of snake, but you will also have a snake with an increased velocity. In this mode you will come across four not_budr obstacles which you will need to avoid while trying to collide with the hot_budr to increase your score. The snake speed will get fast as it size increases.

Hard mode is the most difficult mode the game offers. Now you will come across nine not_budr obstacles to impede your likelihood of success along with the fatest game play that hot_budr snake offers. Keep your eyes peeled as the velocity will increase with every budr you come across.

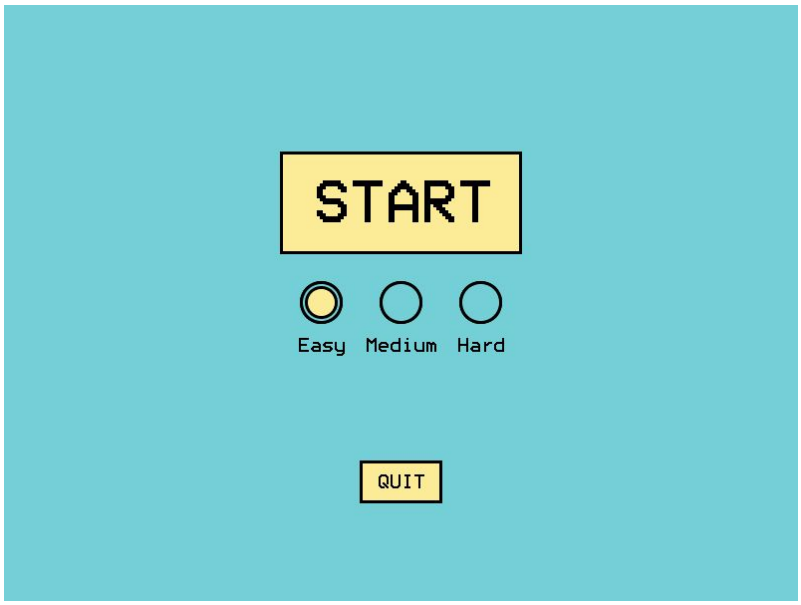
In order to make hot_budr_snake properly complex and increase accuracy, collision detection will be done with edge detection and image processing. Instead of checking the locations of obstacles and butters in comparison to the snake head, we will check neighboring pixel color in order to get a more accurate hit box, since our images are not precise shapes.

Work Distribution

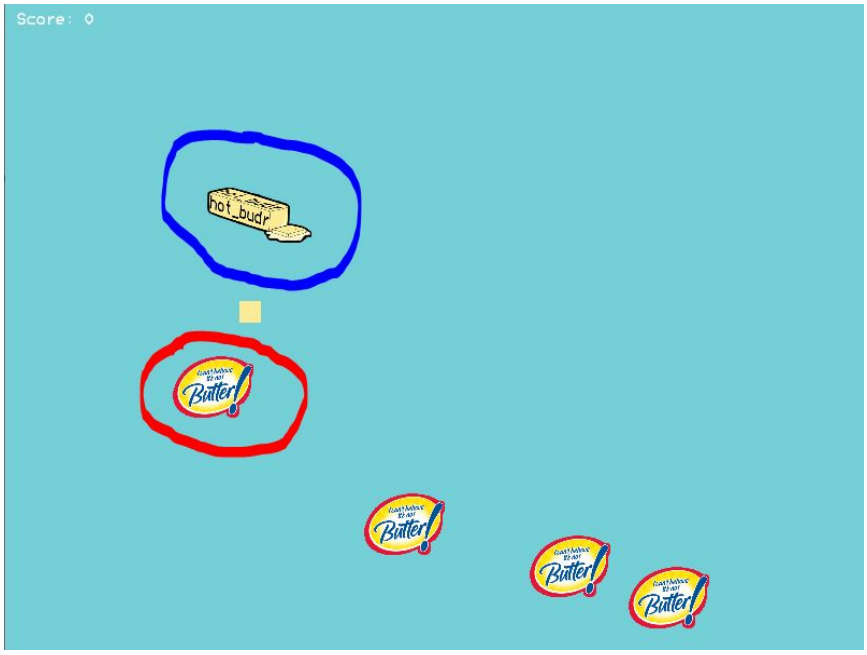
Group Member	**Contribution**
Kathryn Curley	Documentation, beginner snake class, speed difficulty, game state / score keeping
Daniel Gilbert	Documentation, UI and color scheme, snake class, edge detection and collisions with budr, snake and obstacles, testing
Rebecca Loredo	Documentation, sound effects, UI (How to Play), high score implementation, testing
Daniel Shatz	Documentation, images, obstacle class, budr class, testing

User Manual

1. Select a game mode from the menu interface and press start.

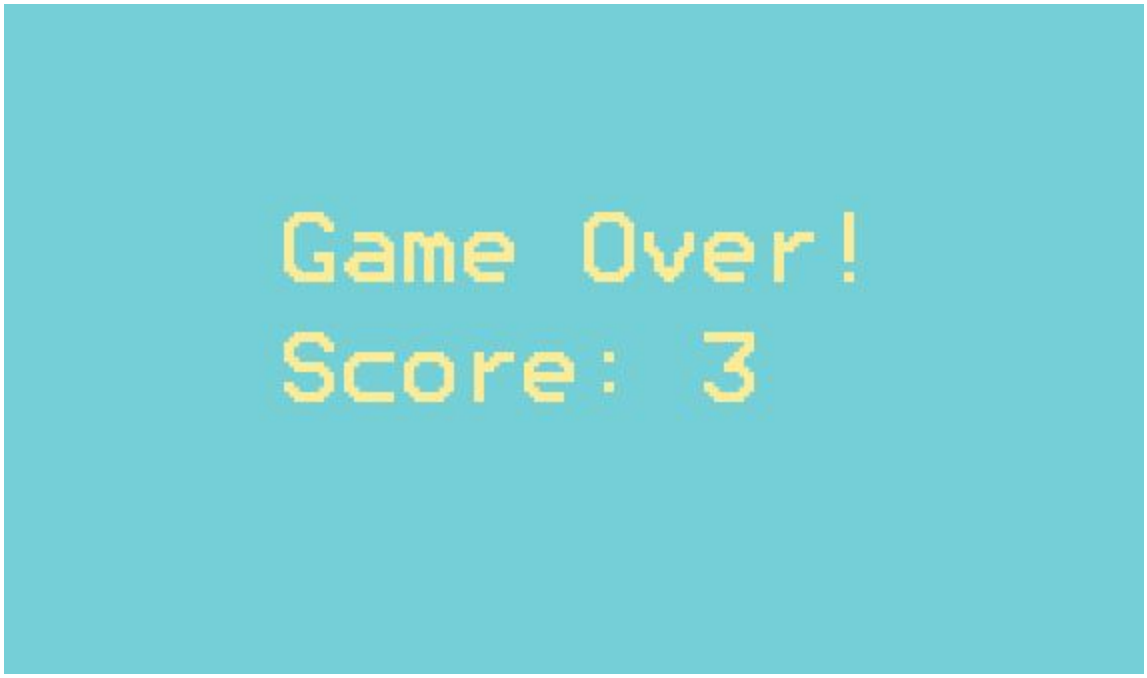


2. Use the arrow keys to control your snake and navigate the playing field.
3. Avoid colliding with your snake and obstacles (I cant believe its not butter) while collecting hot_budrs to achieve a high score.



- obstacle to avoid circled in red
- target circled in blue

4. Your score is kept in the top left of the screen and goes up by 1 every time the player collects a hot_budr.
5. When your score hits certain difficulty thresholds, the speed of your snake will increase, increasing the difficulty of gameplay.
6. If you hit an obstacle, run into a wall or collide with yourself, the end screen displays your score.



Difficulties Encountered

The greatest difficulties we encountered during this project was with the snake class. The snake class was particularly difficult because we needed to make the snake grow whenever it ate a budr and we needed its movement to flow like a snake. In order to achieve this we used a queue, an ArrayList containing PVectors, to remember each x and y coordinate of its body. We created our own queue class and its respective functions to do this. Moreover detecting when the snake eats or dies was also somewhat difficult since we were using image processing and edge detection of pixels to determine what object collides with something. We needed to check every surrounding pixel of the snake head and match it with the body (which was made a slightly different yellow for this purpose), an obstacle, or a budr.