St. Francis Xavier SS: Computer Studies ICS3UC

Colin C CPT Project Report

**Title page**

Grocery Mayhem!

Colin Chambachan

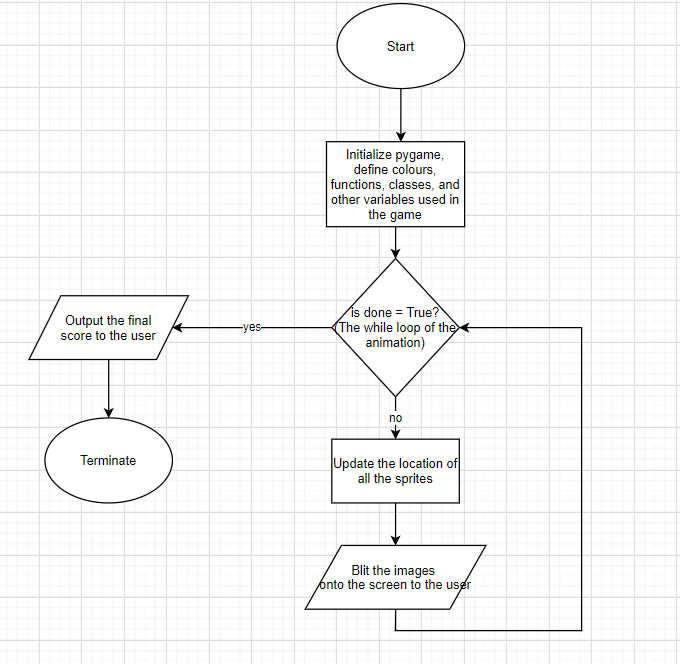
ICS3UC-03

June 19th, 2020



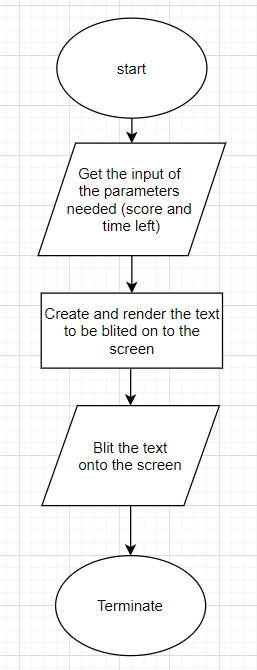
**Design Section**

Main Program:



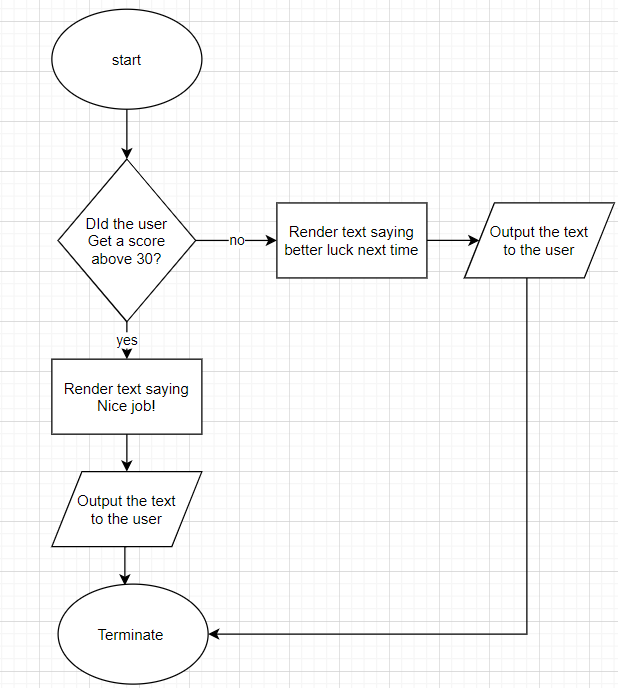
The main program of the game is designed to run like the image above. There were many features I wanted to include into the game, such as the ability for the character to jump, a feature so that the items fall at a faster rate with only 30 seconds remaining, and sound effects. For the program, there two subprograms that I wanted to use, one was used to update the scoreboard (user score, and time remaining), and the other was for the end game screen which displays the score to the user, which were mapped out with the following logic.

Scoreboard subprogram:



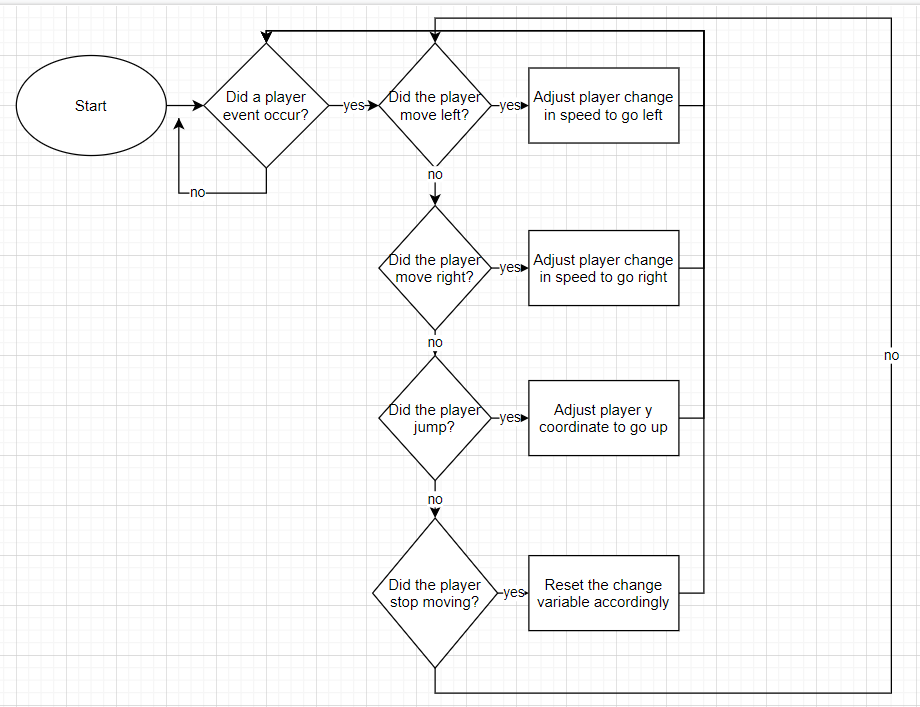
The time left parameter was inputted using a variable called secondsLeft, which basically started at 60 (the number of seconds remaining at the beginning of the game), and reduced it by one every time a second went by. This way the parameter could be inputted with ease and efficiency.

EndGameScreen Subprogram:



To give this game more of a flare, I wanted it to make the end game screen as part of a reaction to how well the user did, and found that the score of 30 was a good benchmark for this score.

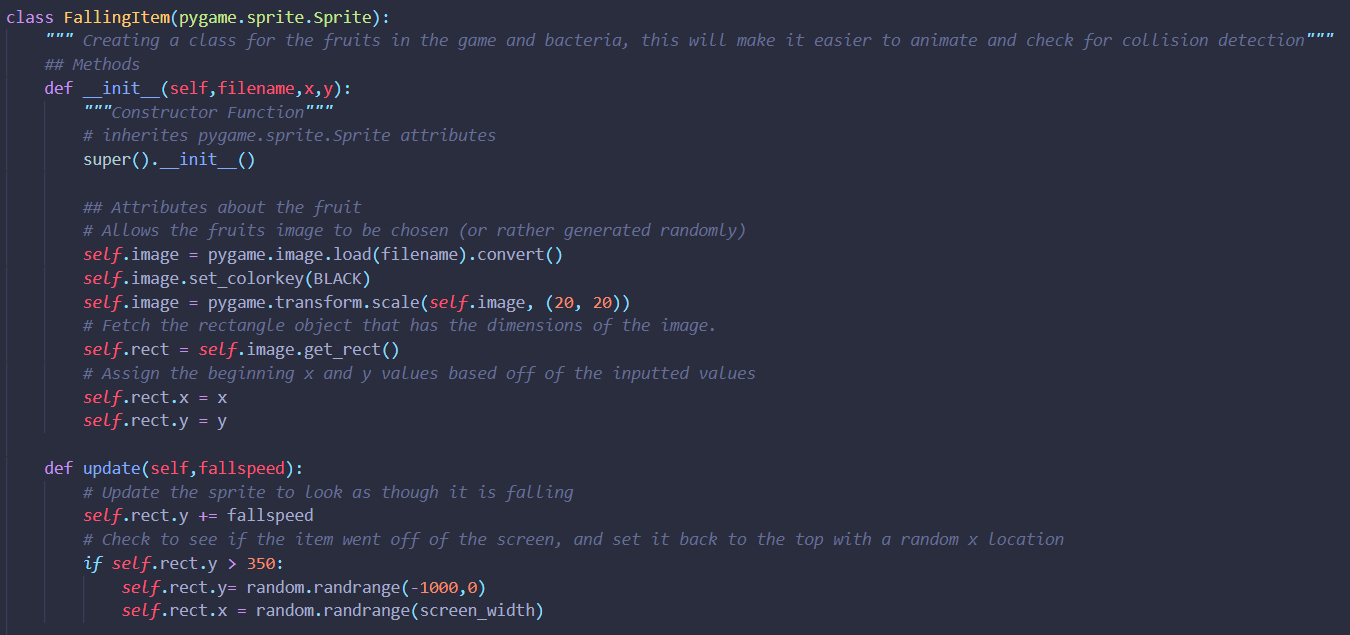
The premise of the player movement worked as follows:

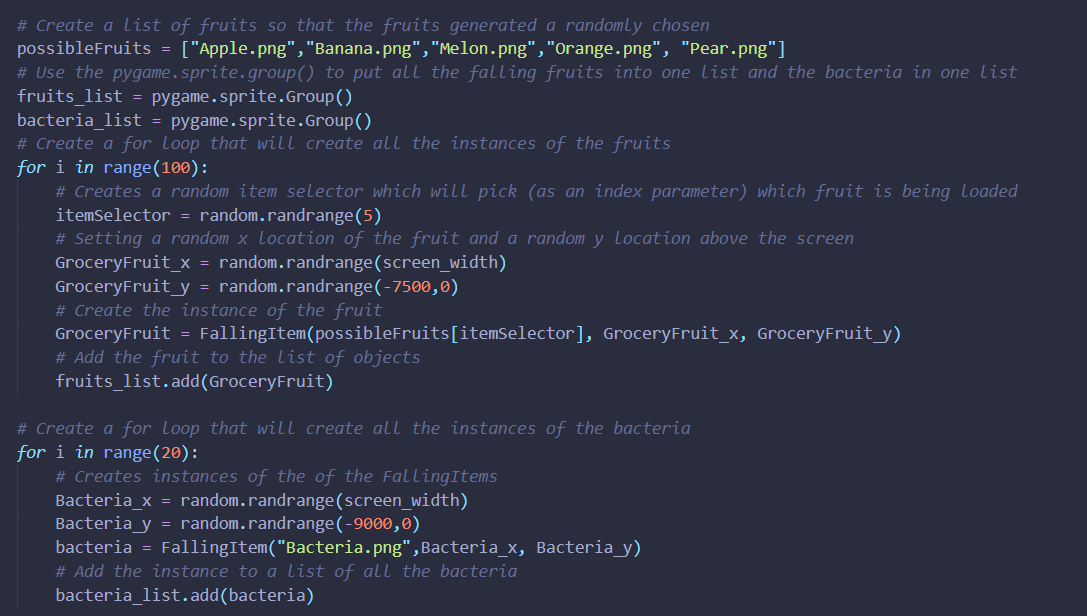


The idea is that it continually checks if an event occurs, and only stop checking once the game is done

**Implementation Section**

The first part of the main code is the creating the class that creates the main character of the game. The code used was taught in Lab 13 of Program Arcade games, where I was able to import an image that I wanted to use as the main character of my game. I created attributes such as self.rect.x and self.rect.y so that I could continually update its location. The self.direction\_facing was an attribute that I came up with so that the computer could figure which version of the image was to be used (the original, which faced left, or the flipped, which faced right). I used an update() method which would automatically adjust the location of the player, creating a more efficient code in the main program loop.





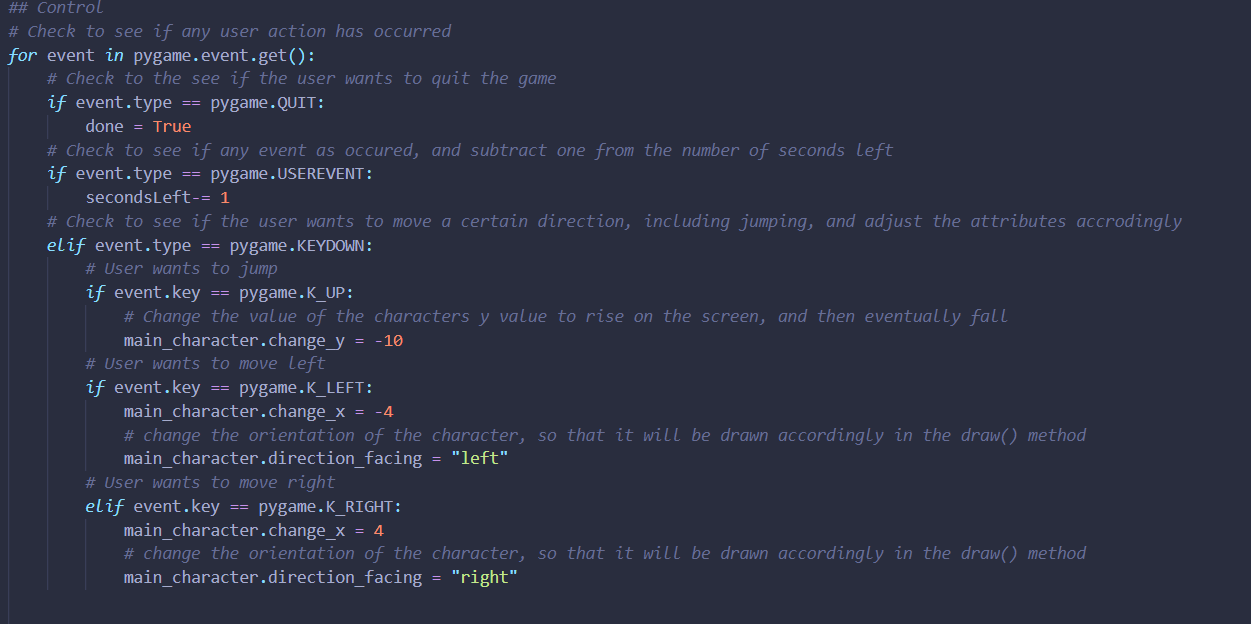
.

The falling items class was pretty simple for me to create, I gave it the standard attributes of a sprite and an update() method that updated the location of the images. I also made it take a parameter of its fall speed, this was to future proof it so that I could change the fall speed at anytime (which I did do when there were 30 seconds remaining)

For this class, I created two objects, one was the falling fruits and the other was the bacteria the player had to avoid. At first, I was stumped on how to have a variety of fruits, but I then realised I could create a list of the file names of the images, and then use a random integer to reference them, thus creating an instance with one of those images being used. The bacteria was straightforward because there was only one image I wanted to use

Lastly, I assigned the images a random y coordinate that was off of the screen, this was so that I didn’t have to create the images during the main code, but they were set by the time the game started, and all I had to do was make them fall onto the frame of the screen. I also made the bacteria have a larger range to spawn them because I wanted the fruits to be shown more, and bacteria be a curveball they had to avoid.

**Testing Section**

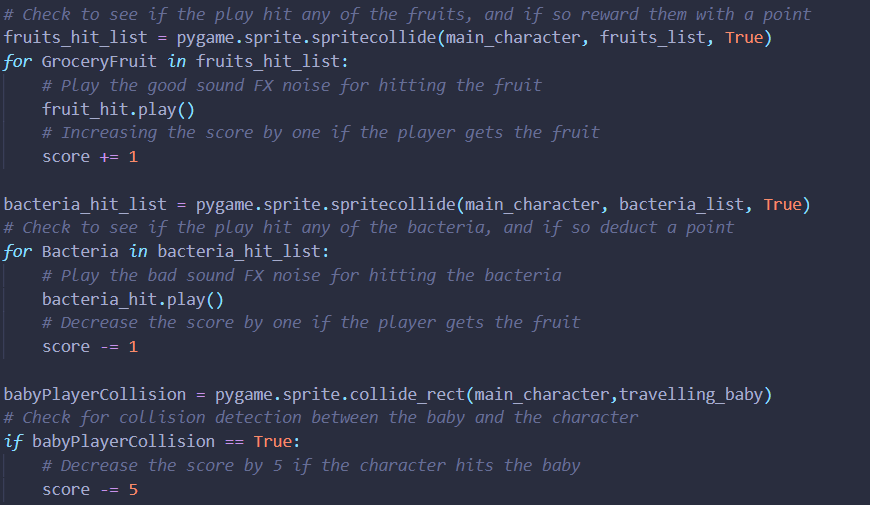


This image is the event-checking of the game, the code checks to see if the user wants to move in any direction, and if so adjusts the main characters speed and also the parameter which adjust whether it should be blit’d flipped or not. The event tracker also checks for if a second passed by, in which case it decreases the number of seconds left (the timer) by 1 (or rather 1 second).

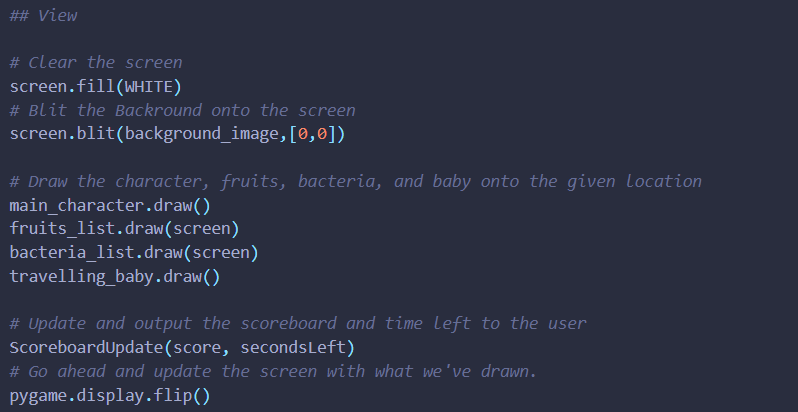
A screenshot of a cell phone

Description automatically generated

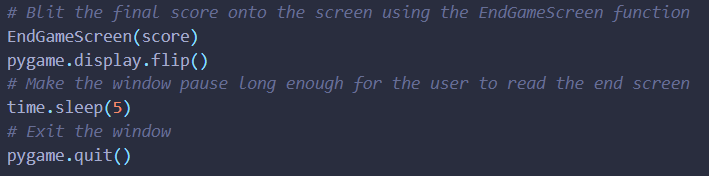
This image displays all of the updating and most of the control events of the game. The code checks a) if the timer ran out b) if the user travelled off of the screen c) whether there are less than 30 seconds left, and adjusts the fall speed and d) if the travelling baby has wandered off of the screen. The code also calls all of the update methods with the needed parameters so they can then be blit’d onto the screen.



This code checks for collision between any of the sprites, and adjusts the player score accordingly.



The code responsible for putting the all the wanted images onto the screen.



Once the game is over, the EndGameScreen function is called and displayed onto the screen, and then the window sleeps for 5 seconds so the user can read the text, right before the game ends and the window closes.

**Testing Section**

In the game, there were many cases that had to be tested. One of these being the falling items within the game (the fruit and the bacteria), which had to be tested so that they were falling properly and the collision detection was functioning properly. When I was testing my code, I was unhappy for the rate at which the fruits and bacteria were originally falling at, they were too clumped up and it became more of a game of going towards the largest clump rather than actually maneuvering, a game which takes a considerable more amount of skill. After testing and adjusting the code, the game was able to evolve closer to what I had originally envisioned, ultimately improving the overall product.

**Analysis Section**

Looking back on the project, I am really happy to see the final outcome. When I first came up with the idea of the game, I was worried that bit off more than I could chew given the circumstances and the work I was responsible for doing in other classes’ as well. I was also intimidated because it was such a big project, that I almost didn’t know where to even start. However, I was really impressed to accomplish a task that inspires me to work on more larger scale projects.

One thing that I think that went really well in the project was my ability to problem solve, I ran into multiples problems, sometimes they were logic errors, sometimes they were syntax and runtime errors, and sometimes there were maximum recursion errors. Yet, ultimately, I was able to overcome them, and the project was able to move forward.

Another thing that I found that went well was the ability to implement new code into the running master code. Even though sometimes I would feel the code would get a little “messy,” I think I was able to structure most of the code in a cohesive and logical order that made the code both easier to look at and easier to adjust as I pleased.

An area that needed more time was probably the jumping mechanics. If I was given the opportunity, I would look to create the jumping effect using attributes like velocity, gravity and such rather than the approach I took. I would also look towards the text that I used; making the score and timer look more aesthetically pleasing, which would add to the general feel of the game, perhaps using a font that made it more cartoon-ish.

One thing that I am really proud of is the general art style of the game. I think that I chose character images, fruit and bacteria images, and a background image that just works so well together that they almost seem that they came from the same place. It was really the look of the game that I had initially envisioned, and why I’m also grateful that I was able to use sprites.

Another thing that I was really proud of was my work ethic throughout the project. As I want to go into Software Engineering, I was really glad that working on this project didn’t feel like *work*, it was a fun experience and I was actually excited to be working on my project. I hope that as I explore my career into the field I continue to have the same passion and excitement as I did during this project.

One aspect of the game I would’ve liked to improve was the learning curve of the game. Right now, it seems as though you’re thrown into the game without any instructions, and you just have to learn what things to do. Had I been given the opportunity, I would try to add a screen that gave instruction before the game actually began.

Further, an aspect that I would improve would be the unique content as the game progress, I had initially hoped to have a “power up” within the game that the play could get skills such as a high jump, or super speed, but because of the timeline I was unable to accomplish that.

Once again, I’m really glad with the turnout of my project, and I’m really proud of what I was able to accomplish. I hope that I can continue to work on fun and exciting projects all the while learning new things like I did with this project.

**References**

The countdown timer method:

<https://stackoverflow.com/questions/30720665/countdown-timer-in-pygame>

Main Character of the game:

<https://www.shutterstock.com/image-vector/friendly-supermarket-worker-holds-box-vegetable-470423108>

Fruits used in game:

<https://opengameart.org/content/fruits-0>

Background image of game:

<https://www.shutterstock.com/image-vector/big-shop-super-market-shopping-mall-568171612>

Travelling Baby:

<https://www.dreamstime.com/royalty-free-stock-photo-shopping-baby-image23973645>

Bacteria Image:

<https://opengameart.org/content/virus-various-colors>

Collision of two sprites:

<https://www.pygame.org/docs/ref/sprite.html#pygame.sprite.spritecollide>

Overarching game music:

<https://freesound.org/people/djgriffin/sounds/506771/>

Ambient Grocery Noises:

<https://freesound.org/people/ecfike/sounds/475703/>

Game won music:

<https://freesound.org/people/EVRetro/sounds/495005/>

Game lost music:

<https://freesound.org/people/deleted_user_877451/sounds/76376/>

Fruit Pickup sound:

<https://freesound.org/people/Phenala/sounds/478647/>