## Problem Definition

There are many purposes to this project. Firstly, learning how to program in an object-orientated fashion instead of in a procedural manner as there are many benefits to this change. In procedural programming, you must code the program in a very specific order for everything to run smoothly and because of this, expanding your program or developing larger projects is very difficult to code well. Some of the problems associated with this can be seen in larger companies where due to procedural programming's specific order, it is almost impossible for multiple people, let alone a development team to work on and not break each other's code. In Object-oriented programming, this problem is solved through the use of user-made classes which break up the code into much more manageable pieces and allow people to receive and return data without other classes needed to see how this data is processed. This means that development teams can work on separate classes and as long as their class takes and returns a set of agreed-upon arguments, everyone else's code will work. The Secondary purpose of this project is to fill the lack of good quality vintage RPG games currently and make something that is well coded, runs smoothly and most of all is very fun to play.

## GANTT Chart

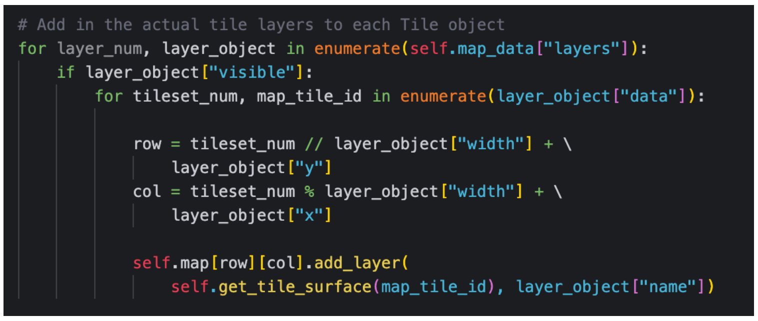
| | | Term 2 | | | | Holidays | | | Term 3 | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Wk6 | Wk7 | Wk8 | Wk9 | Wk1 | Wk2 | Wk3 | Wk1 | Wk2 | Wk3 | Wk4 | Wk5 |
| | Planning | 🟦 | 🟦 | | | | | | | | | | |
| | Map Making | | 🟦 | 🟦 | | | | | | | | | |
| | Map drawer | | | 🟦 | 🟦 | 🟦 | | | | | | | |
| | Player movement | | | | | 🟦 | 🟦 | 🟦 | 🟦 | | | | |
| | Enemy attributes | | | | | | | | 🟦 | 🟦 | 🟦 | | |
| | Enemy Movement | | | | | | | | 🟦 | 🟦 | 🟦 | 🟦 | |
| | Weapons | | | | | | | | | | 🟦 | 🟦 | |
| | Other Items | | | | | | | | | | | 🟦 | |
| | Bug Fixing | | | 🟦 | 🟦 | 🟦 | 🟦 | 🟦 | 🟦 | 🟦 | 🟦 | 🟦 | 🟦 |
| | Game Completion | | | | | | | | | | | | 🟦 |

## Data Dictionary

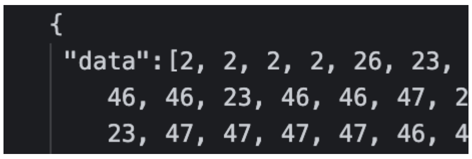
| **Data Item** | **Data Type** | **Format** | **Bytes for storage** | **Size for display** | **Description** | **Example** | **Validation** |
|---|---|---|---|---|---|---|---|
| current_h | Integer | NNNN | 2 | 4 | The height of the game window | 1440 | |
| DIRPATH | String | “LL…” | 43 | 43 | The pathname of the directory of the current file | /Users/sam.carroll/Desktop/Carroll Tile RPG/ | |
| running | Bool | X | 1 | 2 | A variable that is used to determine whether or not to run the main loop. | True | |
| map_data | Array of Records | | 100+ | 100+ | The array that holds all the information regarding tiles eg. tile ids, tile layers etc. | | |
| layer_num | Integer | N | 1 | 1 | Holds the index of each layer in a certain tile. | 3 | |
| name | String | “LL...” | 10 | 10 | The variable containing the name of the player in-game. | “Jerry” | |
| row | Integer | NN | 1 | 1 | The row of the tile object for use in displaying the tiles in the right place. | 14 | |
| sheet | Image | | 3 * image width * image height | 3 * image width * image height | Holds all the different tile images in one larger image to be sliced later. | | |
| tile_slice | Object | | 6 | 6 | Holds the rectangle size and position to “slice” the correct tile off the sprite sheet. | tile_slice = self.get_tile_rect(local_tile_id) | |
| game | Object | | 100+ | 61 | Holds all of the methods and functions in order to run the game. | Game<Map<BigOverworld, 64w, 64h>, Player<Danny, 100HP, LVL1>, 0m> | |

# Development Journal

1. While trying to load all the different layers for each tile such as base terrain, mountains, forest and manmade, I noticed that only one of the tiles was getting displayed on the screen. I went and put breakpoints in my code in my load_map function before run&debugging my code. While skipping through my layer appending for loops one iteration at a time I noticed that in the local variables tab, the tile row and column were not changing as they should in order to populate each tile. I figured out that In the process of declaring the placeholders tileset_num and map_tile_id, to the index and id of each tile using the enumerate() function, I mixed them around and was looking for each tile in the .json using their id, instead of using their index. Because the first four tiles in the file were the same, this resulted in their map_tile_id being identical (2) and adding the layers of each of the first four tiles to the tile with the index 2. To fix this all I had to do was flip the order of these two placeholders so that the index was actually assigned to the correct placeholder, in this case, tileset_num.

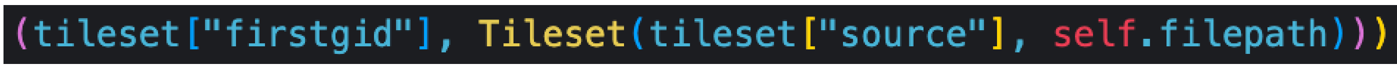


```python
# Add in the actual tile layers to each Tile object
for layer_num, layer_object in enumerate(self.map_data["layers"]):
    if layer_object["visible"]:
        for tileset_num, map_tile_id in enumerate(layer_object["data"]):

            row = tileset_num // layer_object["width"] + \
                layer_object["y"]
            col = tileset_num % layer_object["width"] + \
                layer_object["x"]

            self.map[row][col].add_layer(
                self.get_tile_surface(map_tile_id), layer_object["name"])
```

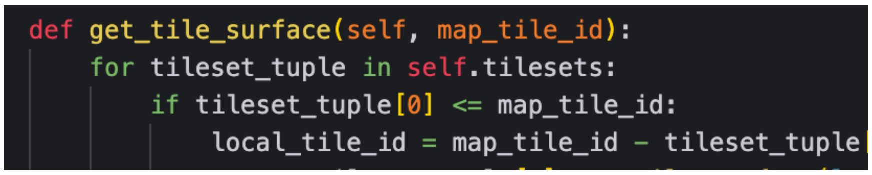


```
{
 "data":[2, 2, 2, 2, 26, 23,
    46, 46, 23, 46, 46, 47, 2
    23, 47, 47, 47, 47, 46, 4
```

2. I had a problem in my code where all my tiles were being loaded correctly except for one of them and when I blitted that map to the window that specific tile would be blank anywhere it appeared in my map file “Overworld_1.txt”. I looked through my code and couldn't find the issue so I looked at my sprite sheet and subsequent map file and found that the image that wasnt working was the very first image (top left corner of sprite sheet) that was being loaded in my code. In my map file, tiles are given different tile_ids in order to specify their position on the sprite sheet. Also, because each tileset needs a blank/empty image, they leave the first id in the list (the index 0) for a blank tile and will specify the index of the first non-blank image, given in the key “firstgid”. In my logic in the second screenshot, I checked to see if the tile I was about to re-id, for use in blitting, was blank or not with the statement “if tileset_tuple[0] < map_tile_id”. This was just a simple error that would make the condition return false for the first tile in the tileset, causing it to be drawn as blank later on. I fixed this by adding a less than or equal to sign (<=) so that it would return True for this first tile and therefore draw the actual image.
(First image below is the format for the tileset_tuple placeholder in the second image.)



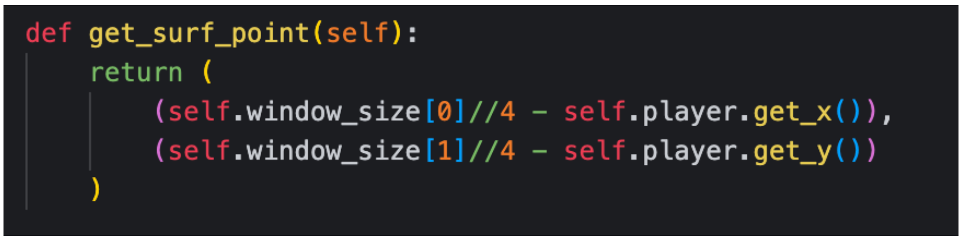
```python
(tileset["firstgid"], Tileset(tileset["source"], self.filepath)))
```



```python
def get_tile_surface(self, map_tile_id):
    for tileset_tuple in self.tilesets:
        if tileset_tuple[0] <= map_tile_id:
            local_tile_id = map_tile_id - tileset_tuple
```

3. Another problem that I had was that when I moved the player the map would move faster than the player. In I blit the map onto the main pygame surface, and then after making the map itself a surface, blit the player onto it. This is so that I can move the map at the same speed in the opposite direction to the player so that to the person playing the game, he appears to be walking in place in the centre of the screen. When I tried to execute this though, I found that the map and player were moving at different speeds and therefore the player would not stay in the centre of the screen. To fix

this I looked through my code and found that because I am upscaling the map, I needed to make the player move faster relative to the map in order to stay in place in the centre of the map. (Screenshot below depicts the calculations)



```
def get_surf_point(self):
    return (
        (self.window_size[0]//4 - self.player.get_x()),
        (self.window_size[1]//4 - self.player.get_y())
    )
```

## Reflection

Throughout this task I believe that I have made a quite nice RPG game that is fun to play and works well. I was struggling quite a bit at the start of the project as the template we were given was coded in an object-oriented fashion and I had never had experience with this programming methodology before. In all of my previous tasks, I have used the procedural programming methodology and although it can work well for smaller and less complicated games, it starts to break down when used at scale. For example, it would be great to make a small calculator app or game but would be very bad for a group project of significant size. On the other hand, through my learning of the Object-oriented methodology, I have been enabled to make a much larger program using classes and instances of the classes for all the objects in my game. I have found that although some of the principles of OOP such as abstraction and encapsulation are hard to always follow coming from Procedural programming where you can access a variable anywhere. I enjoyed making the Player-tracking enemy move function (auto_move) and trying to make him take the quickest vector towards, the player. I also enjoyed creating the maps using the Tiled Software because I had set up tilesets and terrain layers, all you had to do was pick a terrain and start painting and the software would match all the edges to the surrounding tiles seamlessly. Some things that I could have done better would have been to add animated sprites for the characters (player and monsters). I also think that I could have made my attack system a little bit more complicated with additions for projectiles and different evasive/ offensive strategies. In Summary, I believe that I started this project later than I should have because of this lack of knowledge on the topic of OOP. This ultimately led to me having to do a lot of coding in the days leading up to the due date. I think in the future I need to not be so phased when i don't understand how something works or is coded and ask for help earlier and be more proactive in terms of getting an understanding of what I need to code and how I can code it. For example, It would have benefitted me greatly if I had started going through the object-oriented programming tutorials right after we were given the task so i would be able to start chipping away at it earlier.