Game Development Assignment 1 - Ruth Dooley 19300753

I have transformed the space action shooter into my basic interpretation of the 3D simulation game "Overcooked!". The aim of the game is to fulfil as many orders as quickly as possible as a team before the time runs out. This becomes more and more difficult as the game progresses as different obstacles make completing orders more challenging. I will begin the report by discussing the controls needed for the game and follow on with the features that I have integrated into the game.

Controller Guide

Player movement is handled with the WASD letters on the keyboard.

Key-Stroke	Description
W/U	If the player is not facing the up position - player turns to face up. Else move up by one tile.
A/H	If the player is not facing the left - player turns to face left. Else move left by one tile.
S/J	If the player is not facing the down position - player turns to face down. Else move down by one tile.
D/K	If the player is not facing the right - player turns to face right. Else move right by one tile.

The F and L keys for player1 and player 2 respectively, controls the rest of the game mechanics:

- If the player is facing towards a lettuce, tomato or cucumber bin and they press space, they will "take out" the corresponding fruit/veg.
- If the player is facing a plate, lettuce, tomato or cucumber and the press space, the player can "hold" that item. This will take first precedence over taking something out of a lettuce, tomato or cucumber bin in the relevant cases.
- If the player is holding an object and they are facing a plate, counter space or fruit/veg bin, they can place the object they are holding. This takes precedence over picking anything up. This may create a stack if the player is dropping the object that they are holding onto another object. Note* 2 objects of the same type "plate", "lettuce", "tomato", "cucumber" will not stack and if the player tries to combine they will still be holding the duplicate object. I.e. if the Player tries to combine a stack of plate, lettuce, tomato with a tomato, the plate and lettuce will be placed on the tomato to create a stack of plate lettuce tomato and the player will still be holding the tomato.
- If the player is facing the bin and they are holding something, the objects that they are holding will be "binned" and the player will no longer be holding them. This is with the exception of the plate that cannot be binned.
- If the player is holding a stack of plate + a combination of other fruit/veg and the stack is part of the orders needed, the entire stack including plate will disappear and be "delivered". If the stack is not a valid order the player will still be holding it.

Features

Player Movement

As mentioned above the movement of the players are handled by WASD/UHJK buttons and the interactions they have with other objects is handled with F/L. The player is able to move in 100 pixel chunks at a time and this feeds into the grid based movement and collision system that I have created for the project. I also altered the original code so that the player would be facing the direction that they are moving as it is a birds eye view implementation. Every asset is tied to a cell of the grid from 000,000 to 12000,900 including the player and has a 100 x 100 pixels hit box. This is indicated by the tiles on the kitchen floor, each tile is a new box. I decided to implement this system so it would be easy for the user to see whether the player was in front of the asset that they were trying to interact with. I also like how the game has an arcade style, 8 bit feel with the jagged movements that transition to smoothness if a key in one direction is held down.

Multiplayer
The game works smoothly to integrate additional players.

Collisions
My project is able to handle the collisions between player & object, player & player, and object & object. For example, the Player is not able to move through the fruit/veg bins, counters, delivery area and bin, but the fruit/veg and plates can. I also altered the original code (Point3f class) so that the user cannot move outside the play area bounds (original code had bound canvas width 900px).

Player Moving Objects
An important part of the game is assembling orders and I was able to do this by allowing the player to stack the ingredients on top of the plate. The player is able to pick up an ingredient(s) that is stored in memory and can be moved around. I also manage that there is never more than one object of a single type on a stack so that the objects don't seemingly disappear/merge.

Orders
I generate the orders randomly for the player to fulfil and it can be any combination of lettuce, tomato, cucumber (7 options). This is displayed in the right hand panel of the screen and the user is easily able to determine the makeup of the recipe. The user is given 2 orders at all times and another order is added every 10 seconds if there are not already 6 orders total. This was done by rounding the current time in milliseconds to the nearest ten as the current time was only registered every 10 milliseconds. Each of these orders also have a limited time allotted to them and will expire when that time runs out (customisable each level). The list is ordered by soonest to expire and the art keeps up with the orders being added and removed smoothly.

Timer
Each level is on a timed basis. I have handled this timer and it is displayed in the bottom right corner. The level syncs perfectly with the timer and the level ends when the timer hits 0 seconds.

Scoring System
The user generates coins with the orders that they fulfil. Each of the orders will give the player 5 coins per fruit / veg as well as a time bonus i.e the time that they had remaining to fill that order. If an order expires the score will be deducted. This is handled by the delivery area which I have created an animation sprite for. Rejection for orders that are not listed in the order list is also handled.

Plate regeneration
When the user submits an order the plate is taken with the order (as in the actual "Overcooked!"). This means that the plate will need to be reintroduced to the kitchen. I do this by creating plate spawn points when setting up the level. The plate will regenerate when there are less plates in the level than what was declared in the level setup, excluding number of plates "held" by the players.

Sound and Art
I have created all of the art for this project by myself. I used AI such as Dalle2 and Midjourney but they were not giving me the results that I was looking for. I wanted all of the pixels to be the same size in the actual game and the AI was not outputting satisfactory consistency; some pixel art when scaled for my project would look quite realistic because the pixels were so small. Combined with having to remove backgrounds and sizing issues I decided to create my own art and animations using https://www.piskelapp.com/p/create/sprite which also gave me more control over the colour palette used in the game and therefore theme.

I used royalty free stock music & sound effects that I felt fit with the 8 bit theme. Sources:
Overall music: https://www.fesliyanstudios.com/royalty-free-music/downloads-c/8-bit-music/6

Failure, coin, crumple, error, failure and victory sound effects:
https://pixabay.com/sound-effects/search/failure/?manual_search=1&order=None
I was able to make a music player method so that music and sound effects can be handled effectively by just calling the method.

Easily Create New Levels

As a result of the method I used to create the assets and their collisions, it makes it very easy to create new levels. To do so you would need to edit the 4 in this line in the game loop function to the number of levels you want in your game + 1.

```
if (levelNumberSelected == 4){
```

Additionally to actually create the new level, add a new case in the switch statement in the Model.gameDesignSetup() function and add all of the assets you want to include. The players, fruit/veg bins, counter, hole and bin can be added like so:

```
LettuceBinList.add(new GameObject("res/lettuceBin.png", widthAndHeight,
widthAndHeight, Point3f.setPointInit(100,100, "lettuceBin")));
```

As there is only delivery drop off spot it is handled as so:

```
deliveryDropOff = new
GameObject("res/deliveryZone.png",widthAndHeight,widthAndHeight,
Point3f.setPointInit(1000,100, "dropoff"));
```

And add the max number of plates and that number of plate spawn points.

You will also need to edit the following variables:

```
timerStart = 120_000; //The amount of time that level has in milliseconds
orderTimeBeforeExpiry = 25_000; //The amount of time each order has before
expiry
deliveryList = 7; //The choice of orders that can be chosen from
```

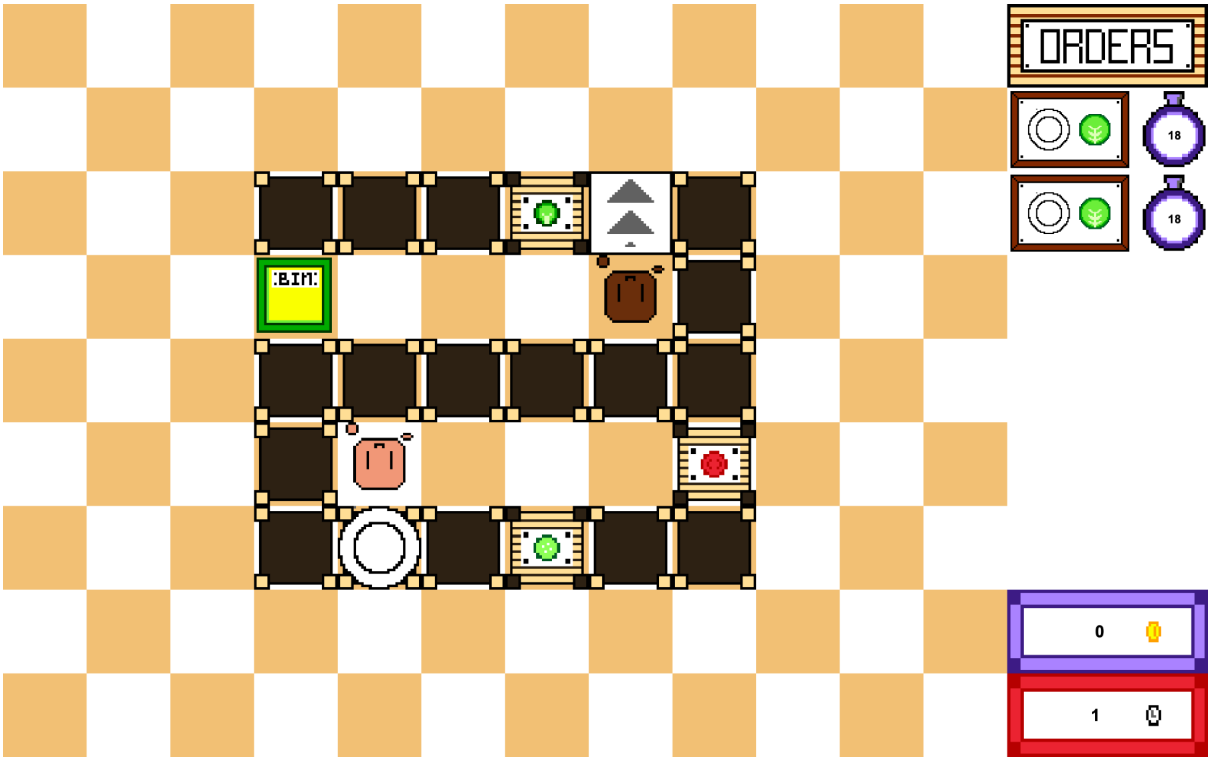
With this setup I could easily create 100's of more levels with the assets that I have already created without having to touch any other source code.

Levels Replay

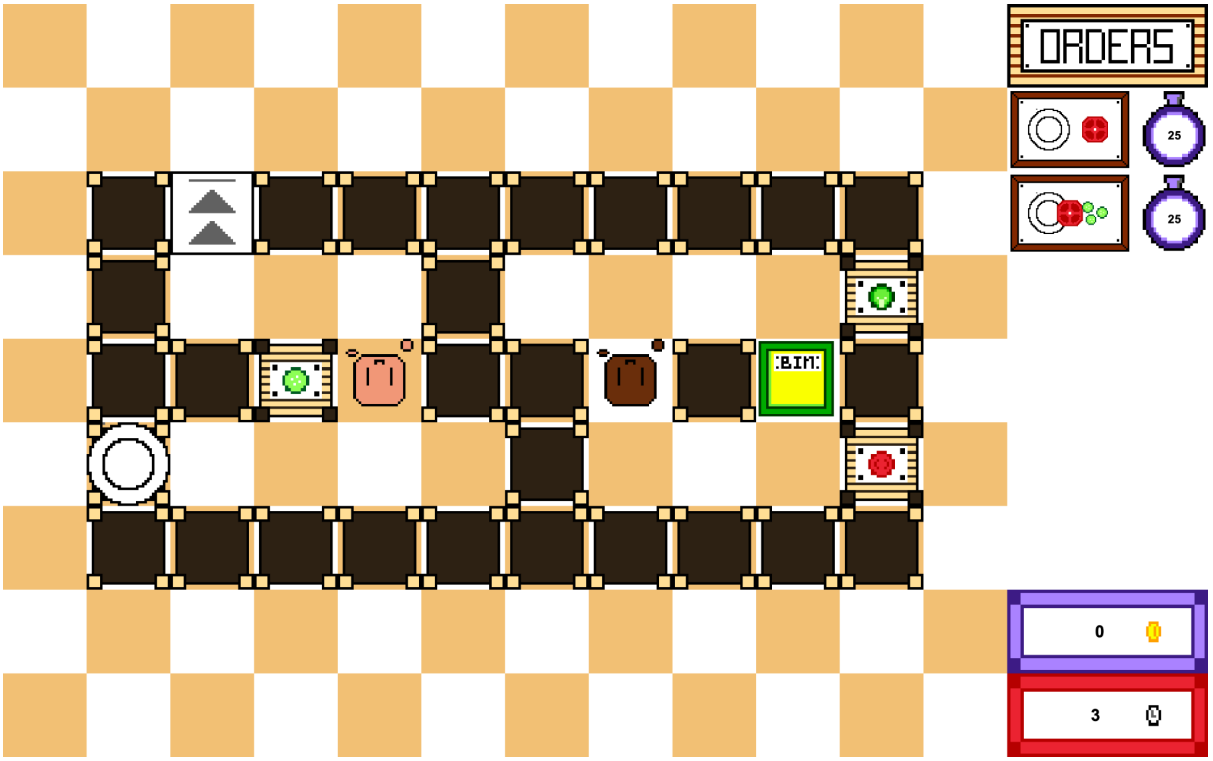
The players need to achieve at least a one star rating in order to move onto the next level. Otherwise the current level will need to be repeated

Snapshots of the Game

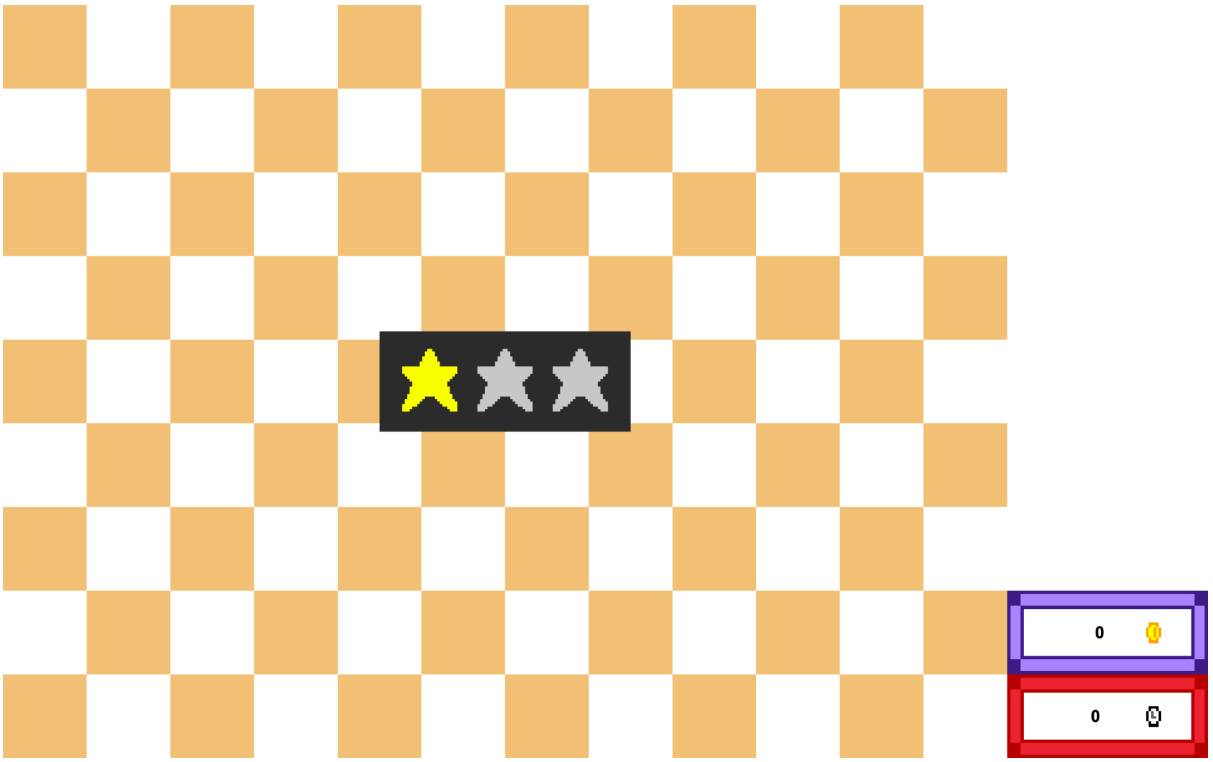
Level 1:



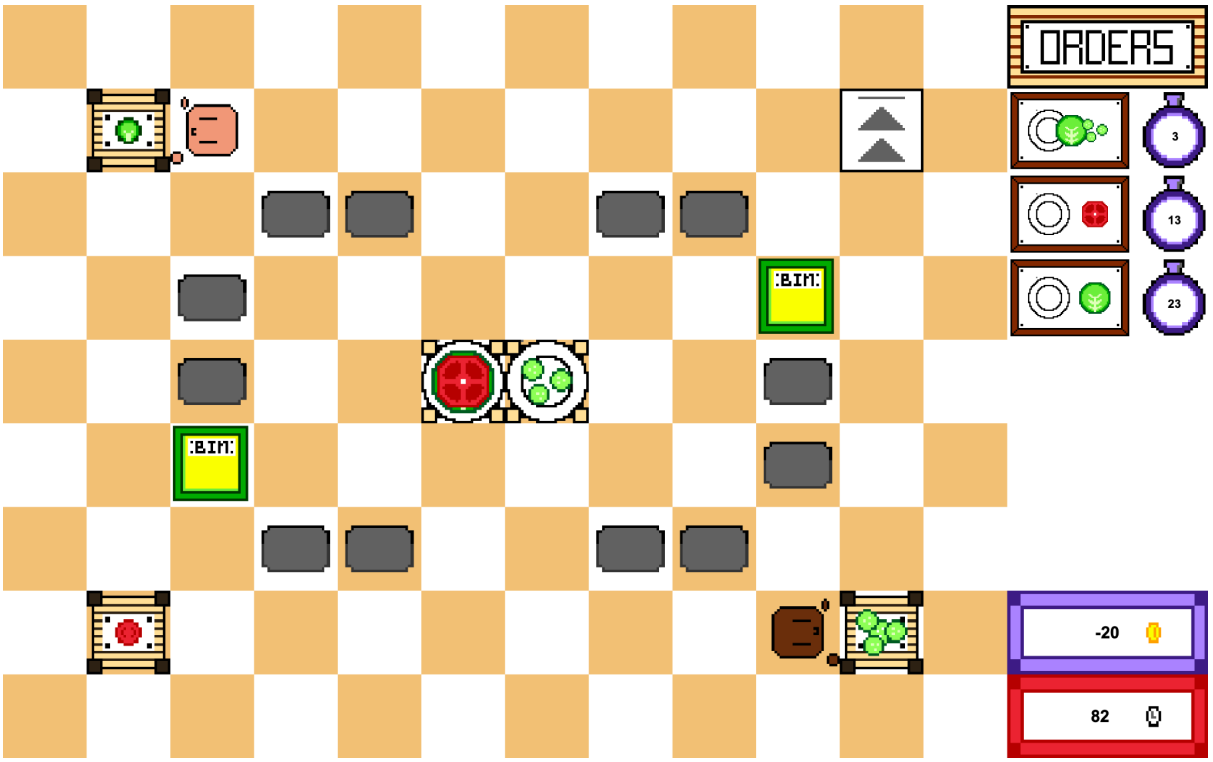
Level 2:



Level Success: (Changed the success to be 0 coins in this case)



Level 3: (Displaying stacked ingredients, 3 orders, negative scoring system)



*Note in the video submission I removed the sound because there was speaking during it and also it is 2x speed in order to display the full 3 levels and have it <3 minutes for the requirements of the assignment.