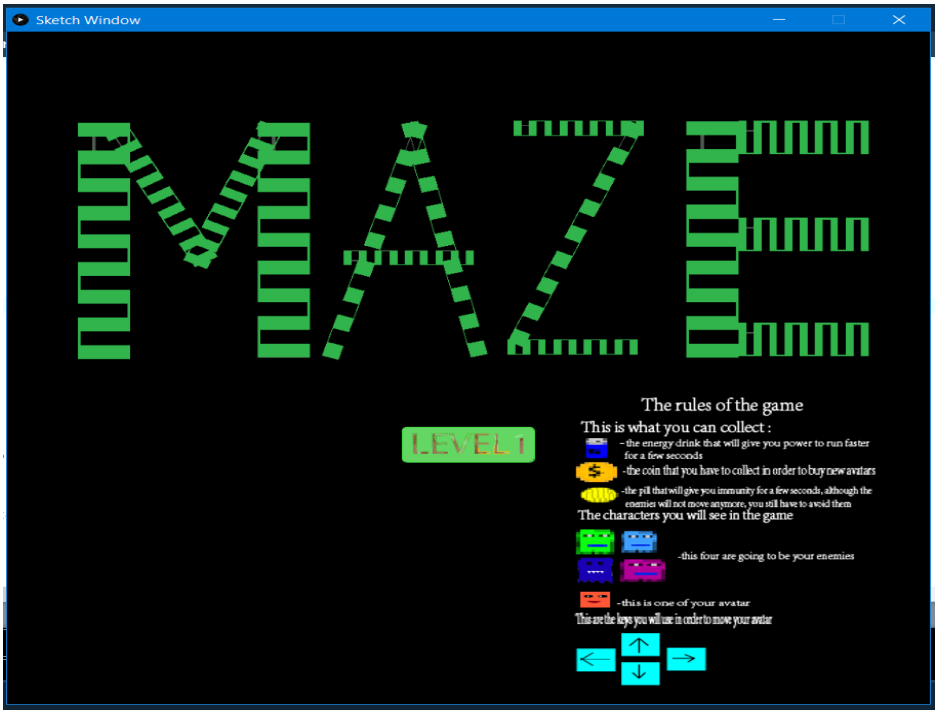
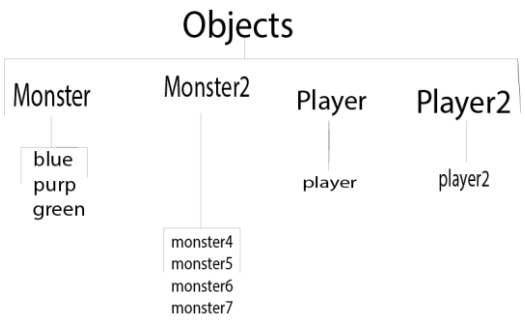
The top down maze I have developed is called Maze, as seen on the first frame of the game. It also contains a brief user manual. The avatar is controlled by the user using the keys: up, down, right, left. There are different objects that the user can collect, the blue can gives the avatar the power to move faster for a few seconds, the coins are helping the user to buy new avatars from the market, the pill gives the avatar immunity from the monsters, they will stay still for a few seconds, while the avatar can move around and collect special objects. Although the pill gives immunity to the avatar, it comes with a limitation, if the avatar runs into the monsters, they will lose a life. The user reaches the end of the game when they arrive into the empty space of the grid, the space that is empty from the beginning of the game.



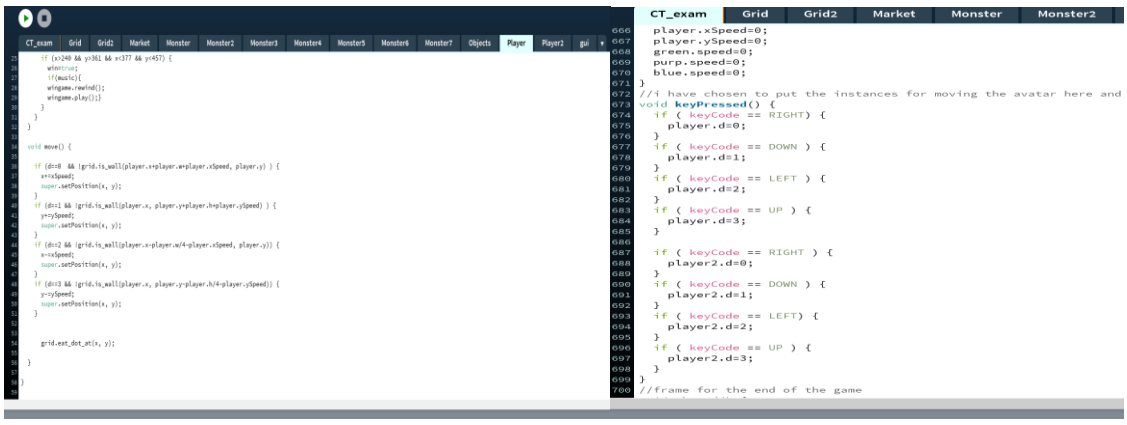
Some of the major features I have implemented are: the UI elements, such as: the mute button for the sound effects and the ability of the user to lose lives, the ability of the monsters to chase the avatar through the maze, without hitting the walls, the market place, where the user can buy different avatars depending on how much coins they have gained in the first level.

In terms of the description of the code, I faced a few challenges in making efficient use of the classes. I wanted each monster to have its own avatar as shown in the user manual visible on the first frame of the game, however I was not able to pass the picture of each monster using the constructor. Therefore, at the beginning of the game I had 13 classes, however I decided to give up on that idea because I wanted to showcase a more elegant approach and to use the classes more efficiently.



Given the fact that I have used collision detection in order to make the player lose a life each time it hits a monster, I had to connect the 4 classes, therefore I have made the parent superclass Objects. The classes Monster, Monster2, Player, Player2 inherit from the class Objects the next variables: x, y, rectWidth, rectHeight and the next methods setPosition, setRectangle and hit. The reason I have two classes for monsters and players is because the class Player is guided by the class Grid and the class Monster is guided by the class Player, while the class Player2 is guided by the class Grid2 and the class Monster is guided by the class Player2.
The diagram below shows the representation of all the classes used in developing the game.

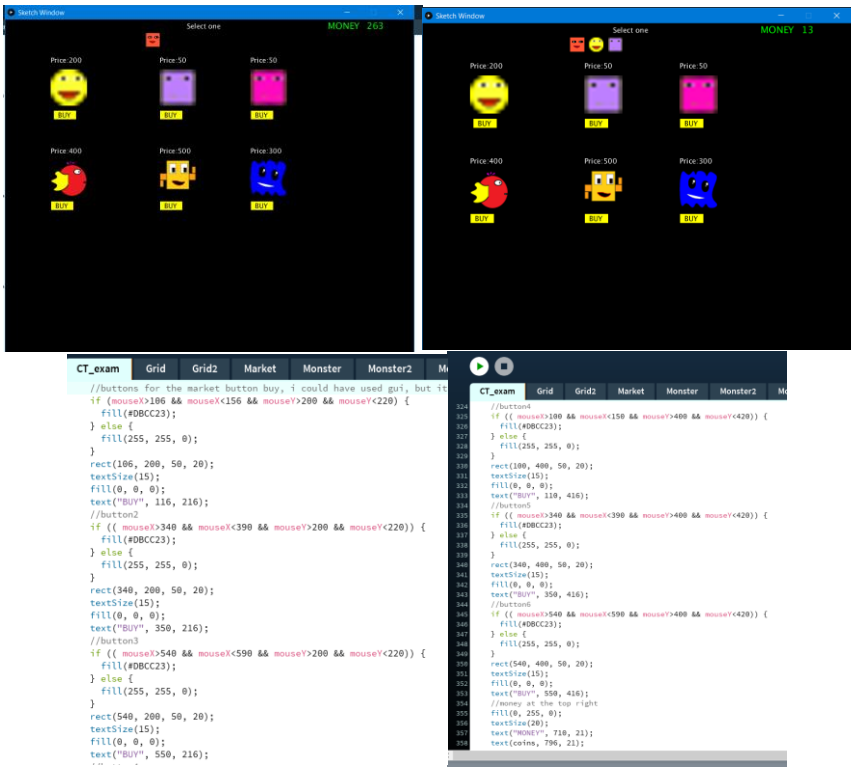
Another challenge I have encountered was the fact that I wanted to make the players unable to move through the blocks of the grid. Therefore, I had to calculate the exact next position if I wanted to make the player move towards left, right, up or down. In terms of the movement of the player, I have used a method which was developed by TfGuy44 (2015). The method presented on the forum involves declaring a variable “d” for example, inside the class Player and Player2, and then assigning a value for “d” inside “CT_exam” in the keyPressed function. The value that is assigned for the variable “d” depends on what key is pressed. I have used this method because it lets the avatar to have a continuous movement, without keeping the key pressed, because the move function is called in the draw function.



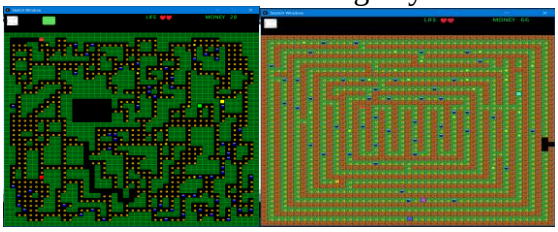
In terms of the movement of the monsters, I have developed a way in which they can have autonomy, however they are dependent on the movement of the avatar, as seen below. Chasing the avatar was not complicated, however the method I used for the monsters to not be able to go through the blocks was a bit difficult, I had to test multiple times if the calculation made was successful. The reason I used super.setPosition(x,y) every time an object was changing its coordinates was to update the class Objects in order to detect collision.

```
void move(){
  if(player.x<x && !grid.is_wall(x+rectWidth/4, y) ){
    x+=speed;
    super.setPosition(x, y);
  }
  else if(player.x<x && !grid.is_wall(x-rectWidth, y) && !grid.is_wall(x+rectWidth, y) ){
    x+=speed;
    super.setPosition(x, y);
  }
  if(player.x>x && !grid.is_wall(x+rectWidth, y) ){
    x+=speed;
    super.setPosition(x, y);
  }
  if(player.y<y && !grid.is_wall(x,y-rectHeight/2+4) && !grid.is_wall(x+2,y-rectHeight/2+4)){
    y-=speed;
    super.setPosition(x, y);
  }
  if(player.y>y && !grid.is_wall(x,y+rectHeight)){
    y+=speed;
    super.setPosition(x, y);
    }
  }
}
```

In terms of the assets I used in this game, I have made every picture and I used Illustrator. However, regarding the sound effects, for the coins I have used a piece of work made by ProjectsU012 (2016), for the moment the user collects the energy drink I have used a sound made by sabbyt2 (2017), for the intro of the game I have used the sound made by knarmahfox(2010), for losing a life I have used the sound made by noirenex (2012) and for the win effect I used the sound made by EVRetro (2019).
Regarding the existing bugs, limitations and possible improvements, my code has existing bugs in terms of the code developed for the market. The bug consists in the fact that even though the user has purchased an avatar, the buy button for that avatar will still be visible.
This problem might be because I have not used G4P in order to make the buy buttons, thus I cannot set the visibility to be true or false. Besides this bug, after the user exits the market, the button back is visible until the end of the game, it is no longer working, but it is still visible.



Another existing bug is in regards with the process of losing lives, the problem is that for example if the user loses a life during the first level, the second level will start with the user having only two lives left, I have tried changing the code by assigning the value 3 to the variable nPics inside the if statement for level 2, which stands for the length of the array for the life counter, however this solution was problematic because the user was not losing anymore the lives.



Students should note that marks are provisional until ratified by the external examiner.

In terms of the limitations, if the user presses the pause button, which is visible only during the first level, the immunity pill is no longer efficient and it works only once.
In terms of future improvements, I would like to add portals, which can be opened only after collecting a specific key, a bonus level, which would let the user collect coins for a limited amount of time and also let the user save the avatars they have previously purchased, as well as the coins gained.

Word count: 1066

Reference List

knarmahfox, 2010. Game Traxx Vol. 1 » Overworld 1-1.wav. [download] Available at: <https://freesound.org/people/knarmahfox/sounds/99700/> . [Accessed 1 March 2021].

ProjectsU012, 2016. 8-bit Video Game Sounds » Coins 1. [download] Available at:< https://freesound.org/people/ProjectsU012/sounds/341695/?page=1#comment> . [Accessed 28 February 2021].

EVRetro, 2019. Video Game Noises » "Win" Video Game Sound. [download] Available at: <https://freesound.org/people/EVRetro/sounds/495005/> . [Accessed 7 March 2021].

sabbyt2, 2017. Sounds » gulp.wav.[download] Available at: <https://freesound.org/people/sabbyt2/sounds/400374/ >.[Accessed 1 March 2021].

noirenex, 2012. Life Lost Game Over. [download] Available at: <https://freesound.org/people/noirenex/sounds/159408/>. [Accessed 1 March 2021].

TfGuy44, 2015. Pac man Game(Not completed) Asking for assistance. [online] Processing Forum. Available at:< https://forum.processing.org/two/discussion/11156/pac-man-game-not-completed-asking-for-assistance#latest >. [Accessed 23 February 2021].