Assignment 2 – Written Report
490380129

Project Description:

The aim of this assignment was the creation of a 2D playable game, implemented through the Java language and the processing.core package. My design of the game includes a player-controlled tank, barriers to protect against incoming projectiles, three different types of computer-controlled "invader" enemies as well as a high score and score HUD. In addition, as part of my extension task, I have also implemented a pause screen as well as an additional health HUD which dynamically displays the current health of the player.

Prior to beginning the assignment, I thoroughly read through the brief provided whilst also consulting the Ed forums. This allowed me to gather further requirements not explicitly stated in the brief such as halting the movement of the tank when both the left and right arrow keys were pressed. I then began prototyping on paper, mapping the prospective location of objects and a general class hierarchy on what would be needed in the game.

Milestone Submission:

My initial implementation included a parent *Object* class which contained inheritable draw and getter methods and a constructor. I also created the *Projectile*, *Invader*, *Tank* and *BarrierParts* classes to represent the respective objects in the game. These classes were subtypes of *Object* and inherited from the *Object* class as they shared overlapping properties and methods. This allowed the object-oriented programming paradigm of inheritance to be applied whilst also reducing the amount of code needed, making it a logical decision. I also included in my milestone submission a *Barrier* class, which did not inherit from *Object*. The intention of this was to act as a class which would condense each of the *BarrierParts* into one *Barrier*, physically mirroring the way each barrier in the game was constructed out of 7 "barrier parts".

The *App* class was also heavily utilised, with game logic regarding the motion of the player being placed in the *draw()* function of the app. In the *draw()* function I also utilised a *render()* function which iterated through *Object* instances in the *App* and called their respective polymorphic *draw()* functions, allowing for a more streamlined process to separate game logic from the drawing portion of the game. The *setup()* function was also used to instantiate and store *Objects* in separate lists based upon their type.

Some issues faced during the milestone included the mapping of movement for the player. Often, the tank would seem to "lag" between inputs, especially when rapidly swapping in between the left and right arrow keys. Upon debugging, I realised this was due to the built-in *keyPressed()* and *keyReleased()* methods, with the continuous swapping changing the value of the *keyCode* causing jerky movement for the tank. Consequently, I implemented a *Map* which would have key-value pairs of *keyCodes* and booleans. I then overrode the *keyPressed()* and *keyReleased()* method to swap the value of the *keyCode* in the map. This allowed for more fluid motion as well as easier handling of edge cases when both the left and right keys were pressed to halt motion.

Final Submission:

My final submission differed greatly from the milestone submission and allowed me to further utilise object-orientated paradigms to efficiently reuse code. In my final submission I instead developed the *Projectile, BarrierParts, Invader, Tank, PowerInvader, ArmouredInvader, Screen, GameObject* and *App* classes. The *Projectile, Barrierparts, Invader* and *Tank* class inherited from *GameObject*, similar to in my milestone submission whilst the *PowerInvader* and *ArmouredInvader* class inherited from the *Invader* class. As the behaviour of the *PowerInvader* and the *ArmouedInvader* was identical to the regular *Invader* with only minute differences in the health, points and damage, it was logical to have these extend from *Invader* as they were a subtype and demonstrated a "is-a" relationship. The *Screen* class was treated as a separate class which did not inherit from the *GameObject* class and encompassed the "game over" and "next level" screens. This was due to the lack of certain properties exhibited by screens such as health and movement. Hence, it made greater logical sense to separate these into their own classes.

From my milestone, I continued to use the idea of a map to control not only movement, but also firing as well as my extension pause screen. I furthermore continued to use the inheritance of common properties from a parent *Object* class which I instead renamed to *GameObject* to avoid clashes with the java in-built *Object* keyword.

Differences instead included the removal of the *Barrier* class following consultation with my tutor due to the lack of necessity of treating each barrier part as a whole – collision occurs separately for each and thus, we should treat each part as a separate entity regardless of position. I also shifted away from the usage of large amounts of logic in my *App* class as advised by my tutor. I instead began implementing a greater amount of methods within each class and then calling these methods from the main *App* function whilst also passing relevant data through these functions which would need to be used. This allowed for more modular code whilst also adhering to programming design principles, with variables assigned the appropriate field such as *private* in order to avoid potential access from unauthorised methods. A major change which I also revoked during the development of my final submission was the use of a *Variables* interface which was implemented by both *GameObject* and *App.* This was used to allow different classes to access certain variables. such as the value for the left boundary, mimicking the use of global variables. However, as I neared the end of my project, I felt this did not adhere to good object-orientated principles and instead removed this interface, passing the values through methods and refactoring code to avoid the use of excessive global and static variables.

Reflection:

Upon reflection on the development of my project, I would instead continuously implement commenting and testing as I developed new features, rather than at the end. This would allow for an easier development process due to documented methods which I could refer back to as well as earlier detection and removal of bugs. In the future, I will also ensure I start assignments earlier due to the large amount of time this assignment took, something I had underestimated. This led to sub-optimal work as well as an increased level of stress, reducing both my mental health as well as the quality of the end product.

Potential improvements include a greater amount of test cases in order to ensure that my code functions correctly as expected. In addition, a further shift from code away from the main *App* class towards a greater usage of methods in classes would aid not only the performance of my game, but also the readability and testing of my code. Reflecting on changes made throughout the course of my project, I believe the changes I made were beneficial. The further refinement and integration of inheritance through the *GameObject* and *Invader* class allowed for easier testing as well as efficient reuse of code whilst the removal of unnecessary classes, such as the *Barrier* class, reduced the amount of code whilst also increasing the logical readability of my program.

Extension:

My extenstention task includes the addition of pause screen (accessible by pressing the 'p' key) as well as the addition of a health HUD which is automatically visible throughout the game. After gaining approval via the Ed forums, I first implemented the pause screen by altering the *keyPressed()* and *keyReleased()* functions and by altering the mappings in the stored *HashMap*. The health HUD was implemented using the *text()* function as well as rendering the *Health* object, an object representing the players health. This dynamically updates based upon the players current health, allowing for easy visualisation of remaining health.

References:

- Marie, N. (2015). *Heart 16*16*. [online] OpenGameArt.org. Available at: https://opengameart.org/content/heart-1616 [Accessed 10 Nov. 2019].