University of York

Department of Computer Science

SEPR - Assessment 4

Implementation Report

Team Craig

Thomas Burroughs

Huw Christianson

Joseph Frankish

Isaac Lowe

Beatrix Vincze

Suleman Zaki

(Any changes in our code for Assessment 4 have been marked using the comment - **//TEAM CRAIG: NEW CODE**.)

New Features

Zombie Mode

Associated Requirements: F11, F13 [1]

Additions to code:

New Relevant Classes:

- Human: Represents an instance of a human(non-zombie) character

Classes Changed:

- SteeringPresets: Addition of method 'getEvade'
- Level: Added attribute 'aliveHumans' which is list of the 'Human' object type.
- Level: Addition of code to resolve for attacks between player and humans in the 'update' method. Addition of if statements to remove game functionality not needed for this mode.

Explanation and justifications:

Zombie mode has the player playing as a zombie with the goal to hunt down all the humans present in a level. The player enters zombie mode for a certain level when they die whilst playing the level as normal player character.

The new 'Human' class is needed to implement the human character type. The code in this class handles motion and interactions related to the human character type. To signal whether or not the game was in zombie and thus should act differently, the attribute 'zombieMode' was added. As this attribute is read and changed by multiple classes in various packages within the program, and is a property of the game itself, it needed to be a static attribute in the 'Zepr' class. The human characters needed the AI capability to run away from the player. To do this the built in AI steering behaviour 'Evade' was used. This was implemented using the 'getEvade' method which is similar to methods used to implement other steering behaviours already present the game. The program already resolves interactions between players and zombies in the 'update' method of the 'Level' class.

The team decided to not allow the user to progress through the game if they complete a level as a zombie, as the user would have essentially completed the level with a free life and would prove less of a challenge. Completing the level as a zombie allows the user to experience the later part of the level, including the map, despite having died. This provides the user with another gaming experience, hopefully engaging them further, as well as preventing frustrations from building up from continuous deaths. To summarise, implementing zombie mode in this way not only satisfies the additional requirements, **F11, F13** made in Assessment 4 but also helps to improve the user's experience and motivation to continue through the levels, justifying its implementation.

Secret Item/Cure

Associated Requirements: F12 [1]

Additions to code:

New Relevant Classes:

- Cure: Represents an instance of the cure pickup.
- Human: Represents an instance of a human(non-zombie) character.

Classes Changed:

- Character: Addition of method 'getDistanceTo'
- Level: Added attribute 'aliveHumans' which is list of the 'Human' object type.
- Level: Addition of methods 'cure', 'getClosestZombie', 'getClosestTarget'
- Zombie: Changed parameter in 'update' method from 'player' to 'target' so that zombies can target humans as well.

Explanations and justifications:

The cure is a item the player can pick up which turns zombies within a certain radius into human character. As humans, these character run away from zombies and zombies hunt them if they are closer to them than the player. The cure has a chance of appearing in a level after each wave much like the powerups in the game.

The new class 'Cure' is needed to implement the cure item. It handles information about the item as well as the process of its activation. The class inherits from the 'PowerUp' class because the cure item is very similar to the power-up items already in the game. The list 'aliveHumans' is needed to store the humans present in a level so that they can be updated and can interact with other objects. This 'cure' method has to be in the 'Level' class as it requires access to the 'aliveZombies' and 'aliveHumans' lists. The 'getClosestZombie' method finds the closest zombie to a human character so knows it which way to run away. The 'getClosestTarget' finds the closest target, player or human character, so that a zombie can chase the appropriate one. Both these methods are in the 'Level'

class as they require access to previously mentioned lists. Both these methods use a distance value to determine their what they'll return. This distance value is calculated by the 'getDistanceTo' method. Zombies can chase different characters so the attribute 'target' and method 'setTarget' are needed to assign and store which character they are chasing.

The team decided to only allow the cure item to turn zombies into non-zombies within a restricted radius, as the removal of all zombies in a level or wave would significantly reduce the challenge. We believe that challenging levels are vital for engaging the players interests and allows the player to feel accomplished when completing each level. A further decision was made to prevent the user from using the cure when a zombie, this is justified as we did not want the user to be able to progress through the level if they have died. Our choice to implement the cure in this way was directly influenced by requirement **F12** as it explicitly states that 'the player should be able to pick it up if a human', due to this we believe our implementation is justified as it satisfies the associated requirement.

Points System (including **GUI addition**)

Associated Requirements: E2 [1]

Additions to code:

Classes changed:

- Zepr: Added static integer attribute 'totalPoints' to 'Zepr' class.
- Level: Added integer attribute 'levelPoints' to 'Level' class.
- Level: Made changes to 'update' method so that points are incremented and decremented appropriately.
- SelectLevelScreen: Changes made to display the points the player has.

Explanation and justifications:

The player earns points for killing zombies in a level. Once the level is completed the points they earned for it are displayed and added to the number of total points which are visible on the 'select level' screen.

The 'totalPoints' attribute holds the value of the total amount of points the player has earned across all levels. As this value is needed by various classes it is suitable for it to be a static attribute of the 'Zepr' class. The 'levelPoints' attribute holds the value of points earned in the level currently being played. The value is only relevant to the current level so the attribute can be private to the 'Level' class as it is only used by that class. It used in the 'update' method to increment it for each kill and also be added onto the value of 'totalPoints' at the end of the level.

This addition is justified as it ensures our product satisfies the product brief laid out by the stakeholder project and helps to further satisfy requirement **E2**. The team felt that the game we inherited did not provide the user with enough visual feedback at the end of levels. Our addition of a point system helps to keep the user engaged with the game as it provides an additional goal for the user to reach. Furthermore, it allows the user to set their own goals for personal achievement. For example, the point system may encourage users to play the game again once in order to beat their previous point score. Adding to the games replayability value, improving marketability.

Changes

Improved Zombie Attack

Associated Requirements: F9, F4 [1]

Additions to code:

Classes Changed:

- Zombie: Value assigned to 'hitRange' attribute changed in the constructor.
- Constant: Value for 'ZOMBIEHITCOOLDOWN' changed.
- Level: 'update' method changed to resolve attack better.

Explanation and justifications:

An improvement was needed for the zombie attack mechanic because it was very unintuitive. Zombies would not attack the player unless the player attempted to attack and would often not attack other human characters causing them to be stuck next to human character forever as they would not kill them. The source of the issue was found to be the value of the zombie 'hitRange'. It was not large enough to trigger an attack even though the zombie was as close as it could be to it's target. The value was increased slightly to fix the issue. To fix the problem of the zombies only attacking a player when the player attacked; the 'update' method was changed so that the attack trigger did not depend on the players attack ability. Even with these fixes, the zombie attacks were so slow that a zombie would be stuck attacking a human for almost an entire wave. To prevent this, the zombie cooldown value was reduced so that they can kill faster.

This is justified as unpredictable zombie attacks may build user's frustrations with the game, as it defies the user's intuition and understanding of the game's combat system. This also ensures that requirements **F9** and **F4** are suitably satisfied, by ensuring that all zombie types perform as described in the user manual.

References:

[1] Team Craig. *Updated Requirements - Assessment 4*. Available: https://teamcraigzombie.github.io/assets/downloads/UpdatedRequirements4.pdf