Design

Overview

2D Stickman Defence is an RTS/TD game in which the player must defend their castle and destroy enemy forces. This is done by choosing from a selection of stickmen fighters (with various strengths and weaknesses) and placing them on the map, ready to fight enemy stickmen. There is one path from each player to all other players, and the stickmen are placed on one path, originating from the player. The stickmen will then follow the path to the enemy castle, and engage in battle with enemy stickmen they encounter on the way. Once they reach the enemy castle, they start attacking it, until it is destroyed. Once a castle is destroyed, that enemy is eliminated, and all its stickmen also disappear. The last castle standing wins!

When launching the game, the player is asked how many enemies they would like to play against, followed by the enemies' difficulty. This is something I changed from my original pitch, where I said that the game would automatically start with 1 enemy, and as each level increased, an enemy would be added, up to 3 enemies. After that, the enemies would just get smarter. The reason I decided to change this and let the player pick how many enemies they wanted to play against was to give more flexibility to the user. In this way, people who want to play against 3 enemies the whole time don't have to play 2 games with fewer enemies before they get to play in the way they prefer. Similarly, players who are new to the game or struggling to make progress can focus on only playing against 1 enemy in order to practice, before tackling the game with 3 enemies. Once they have made their choices, the game begins. In order to play, the user can select which path to spawn a stickman on using the up/down arrow keys (the selected map is highlighted in blue), and can select a stickman by clicking on the stickman's "card" from the list of cards on the left hand side. They have a number of coins, which *only* increases with time, and the price of each stickman, along with the stickman's stats (strength, armour, dexterity, range) are displayed on the cards. Once a stickman is selected, the user's coins decrease by the price. The stickman is then spawned onto the specific path, and makes its way to the castle at the other end of the path. The stickmen themselves vary in stats – we have a small/fast/cheap option, a few regular options that get more expensive as they get stronger, and we have a magic stickman which is very expensive with a large ranged fireball attack, but with very low health, and which doesn't move (just defends the castle). Finally, we also have a defence option which increases the castles defence, and is visualised by a black tower.

For engaging in battle, the stickmen have a list of priorities. They first fight any enemy stickmen that are owned by the castle they are going towards, if there are any within their range on their path. If not, they then look to see if the enemy castle they are going towards is in range, in which case they attack it. If it isn't, they then look for other enemy stickmen to attack, if they are within range. Finally, if none of these are in range, the stickman begins to move in the direction of the castle at the end of the path. The reason I chose the list in this way is so that stickmen are focusing on the specific enemy castle they were assigned before attacking enemy stickmen, which creates an interesting dynamic as sometimes, stickmen of different teams team-up to attack a specific castle, before attacking each other when the castle is destroyed. However, to give a player whose castle is being attacked a better chance at defending, if they were to spawn a stickman whilst being attacked, the enemy stickmen would turn their attention to that player's stickman (and therefore not attack the castle) until the stickman is dead, in which case they would go back to attacking the castle. This makes more sense, as it gives some breathing time to a defending player under attack if they spawn a stickman, whose job it is in fact to defend the tower. Finally, if a stickman's target castle is destroyed, and there are no enemy stickmen in sight, they then randomly choose another enemy castle to attack, and start making their way towards it. If there are no more enemy castles, the player has won the game.

To run the game, simply compile and run the "Game.java" file. Make sure you have the processing library included too, which wasn't included in the source code because it took too much space to include in the zip file and wouldn't upload to MMS!

Implementation

In terms of the code, which is contained in the “src” directory, I decided to include 4 packages – GameObjects, Players, Upgrades and Weapons.

The GameObjects package simply contains aspects of the game used repeatedly, such as buttons, paths, upgrade cards (the cards the user can click on to buy a stickman), the healthbar and the “Game State”, which is used by the AI. Aside from the GameState, these are POJOs, which contain a draw method to be drawn on the screen, some simple getters and setters, and some extra useful methods used by the game (such as reducing health on healthbar, clicking a button, purchasing an upgrade from a card, etc.). The healthbar itself was created using the tutorial and help at the following link: https://www.openprocessing.org/sketch/120612

The Players package contains the class Player, which contains all the methods needed for a player to play the game. The draw method is responsible for drawing the castle, otherwise the class contains the player’s upgrades (stickmen/defences), their enemies, their health, etc. The methods it contains include getters and setters used by the Game, such as getting the location/health/upgrades/enemies of a player, as well as deducting a player’s coins and more helpful methods. The Enemy class extends the Player class, and adds functionality for the AI. The main method used by the Game class is the getMove method, which gets that player’s best move, given a game state. Initially, the Game gives “null” as the game state, in which case the method uses the getInitialGameState method to generate the current game state. A game state includes the players attributes (health, attack, dexterity), which includes the attributes of the player’s stickmen, and also all of the player’s enemies’ attributes. It then has an “evaluate” method, which gives a score on the current game state for the given player. getBestMove then evaluates the gameState after each of its possible moves, and picks the move which results in the best game state. This is a recursive method, which stops recursing when “max_steps” is reached (depends on the AI difficulty), meaning that if max_steps is 10, the AI will evaluate each possible game state 10 moves ahead, considering every possible combination of moves. This value is 2 on “Easy”, 5 on “Medium” and 10 on “Hard”, which does make a difference in the gameplay. For example, the Easy AI that only looks two moves ahead tends to choose volume of stickmen over their power (more weaker stickmen), which tends to hurt the player in the long run. Finally, the other methods in this class help with getting the values to create the game state after a given move, such as getting the total enemy’s attributes after a move, as well as the player’s, which requires seeing which enemy stickmen are attacking the player’s castle (and calculating the damage), and therefore deciding which move will result in the highest player health, attack and dexterity, and lowest enemy health, attack and dexterity. In addition, the “Move” class represents these moves, and simply contains an upgrade card that can be purchased, and the target that that upgrade would be targetting. If a move is null, it means that the player has decided to do nothing, and wait (in order to get more coins, to save up for more upgrade cards).

The Upgrades package contains all the possible upgrades a player can purchase, which are stickmen and defences. The Stickman class is abstract, and contains the methods for drawing a stickman, controlling their movement with their weapon, and useful methods used by the Game class. A stickman has different states they can be in – either they are chasing some target (another stickman, or an enemy castle), or they are fighting a target. Once the target is dead, they go back to chasing, in order of the priorities mentioned above. The classes which extend this class are the different possible stickmen that can be purchased – StickmanLevel1 (a smaller, weaker but faster stickman with a wooden dagger), StickmanLevel2 (a stickman with a wooden sword), StickmanLevel3 (a stickman with a metal sword) and StickmanLevel4 (a stickman with a gold sword). These use the different weapons in the Weapons package, and have different strengths, weaknesses and costs. A different kind of stickman, StickmanLevel5, is the magic one which attacks with a ranged fireball. In this class, we have added methods to make the fireball move without the whole stickman. In addition, we have the FortifiedTower class, which is the defence option to the user, and adds armour to their castle. This is a simple class which simply draws a new image, replacing the old one of the castle, and has more health.

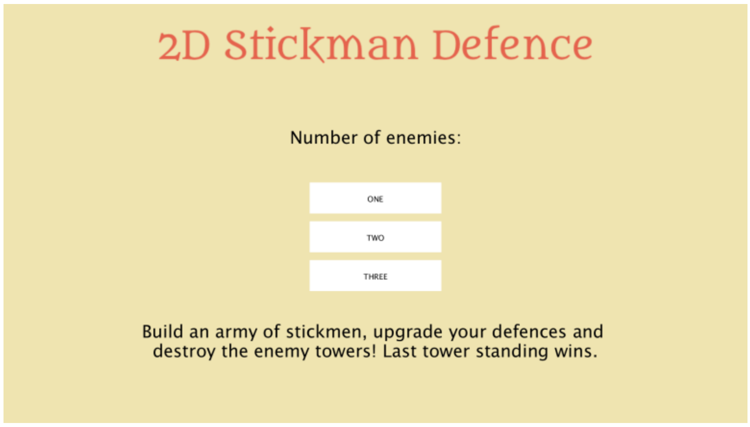
The Weapons package contains all the weapons stickmen can carry. The Weapon abstract class contains common methods all weapons use, such as getters and setters for their attributes. Each weapon is then in charge of implenting its draw method and its movement, which depends on whether the stickman is chasing or fighting. The weapons included are WoodenDagger, WoodenSword, MetalSword and GoldSword, with different looks and attributes.

Finally, the Game class is in charge of running the game. It lets the user pick their game options, sets up the game according to these options, and controls the drawing of the whole game by using the above objects, as well as enforcing the rules and controlling the flow. It also handles user input, such as mouse click (to select items) and key presses (to select a path to spawn stickmen on).

The Stickman class and the Weapons package was initially based on the same classes as in my second practical for this class, hence the similarities. However, apart from some of the movement (which was extended) and the look of the shapes of these objects, most of the code and functionality was changed and edited to fit the needs of this game.

Testing

Menu:

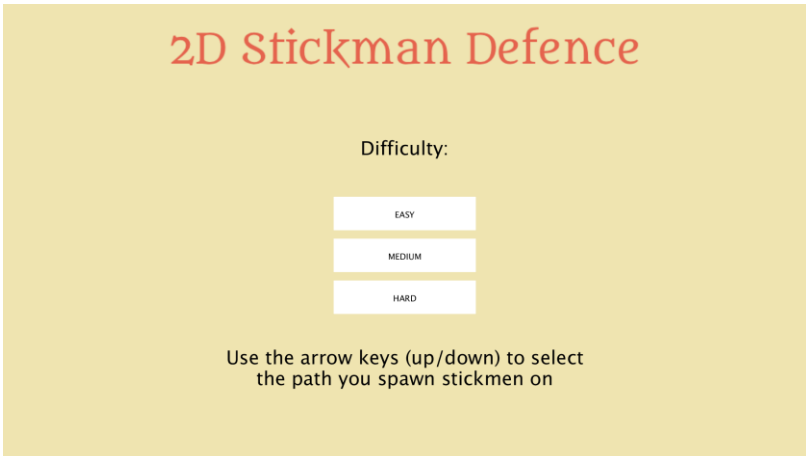


2D Stickman Defence

Number of enemies:

Build an army of stickmen, upgrade your defences and destroy the enemy towers! Last tower standing wins.

Choosing difficulty:

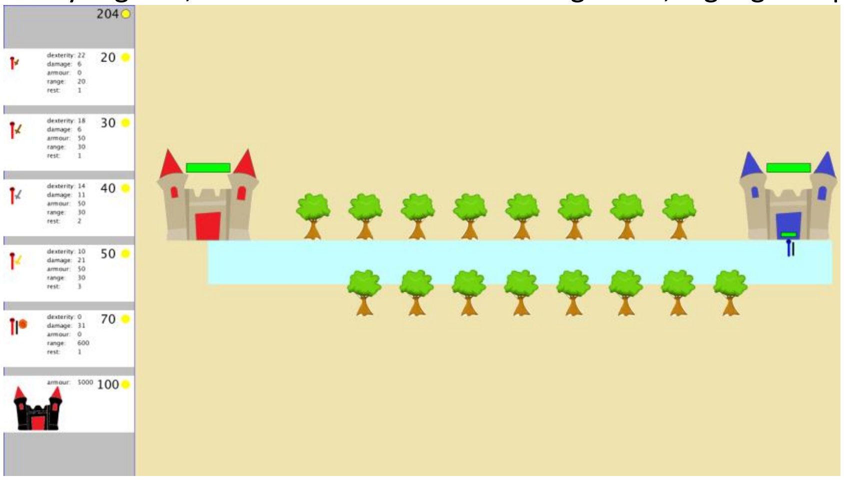


2D Stickman Defence

Difficulty:

Use the arrow keys (up/down) to select the path you spawn stickmen on

2 Player game, menu on the left for choosing items, highlighted path where stickmen spawn:



204

dexterity: 22 damage: 6 armour: 0 range: 20 rest: 1 — 20

dexterity: 18 damage: 6 armour: 50 range: 30 rest: 1 — 30

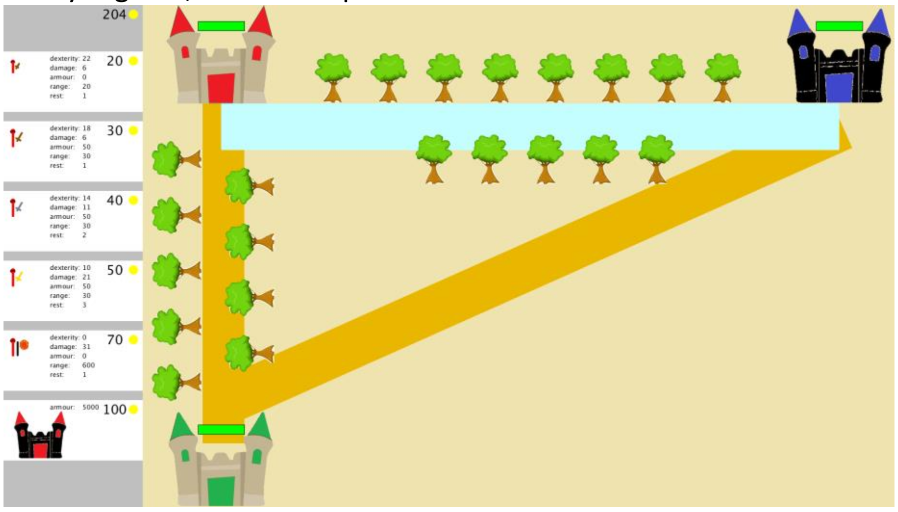
dexterity: 14 damage: 11 armour: 50 range: 30 rest: 2 — 40

dexterity: 10 damage: 21 armour: 50 range: 30 rest: 3 — 50

dexterity: 0 damage: 31 armour: 0 range: 600 rest: 1 — 70

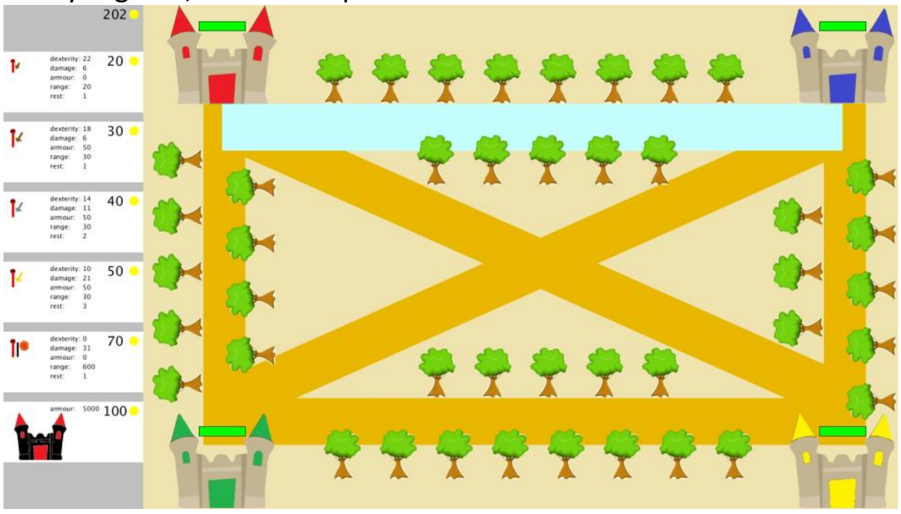
armour: 5000 — 100

3 Player game, with more paths



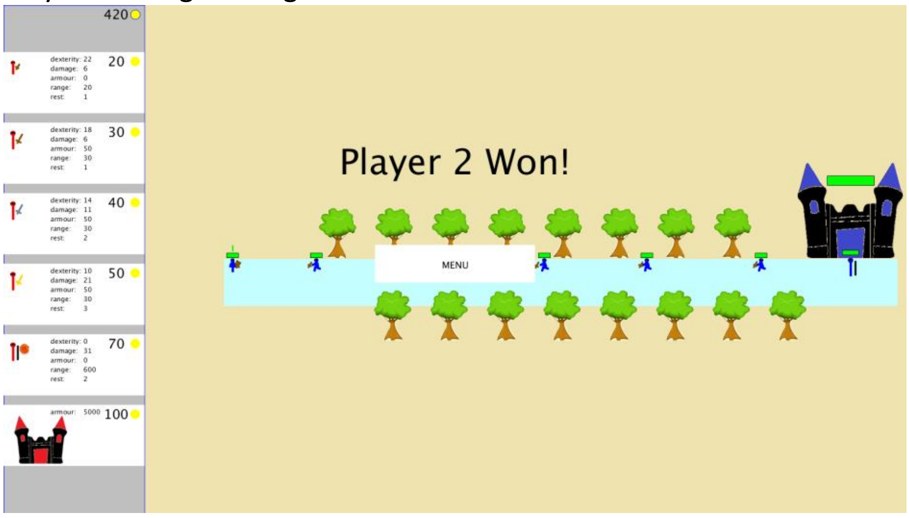
```
204
dexterity: 22   20
damage:    6
armour:    0
range:     20
rest:      1

dexterity: 18   30
damage:    6
armour:    50
range:     30
rest:      1

dexterity: 14   40
damage:    11
armour:    50
range:     30
rest:      2

dexterity: 10   50
damage:    21
armour:    50
range:     30
rest:      3

dexterity: 0    70
damage:    31
armour:    0
range:     600
rest:      1

armour:  5000  100
```

4 Player game, with more paths



```
202
dexterity: 22   20
damage:    6
armour:    0
range:     20
rest:      1

dexterity: 18   30
damage:    6
armour:    50
range:     30
rest:      1

dexterity: 14   40
damage:    11
armour:    50
range:     30
rest:      2

dexterity: 10   50
damage:    21
armour:    50
range:     30
rest:      3

dexterity: 0    70
damage:    31
armour:    0
range:     600
rest:      1

armour:  5000  100
```

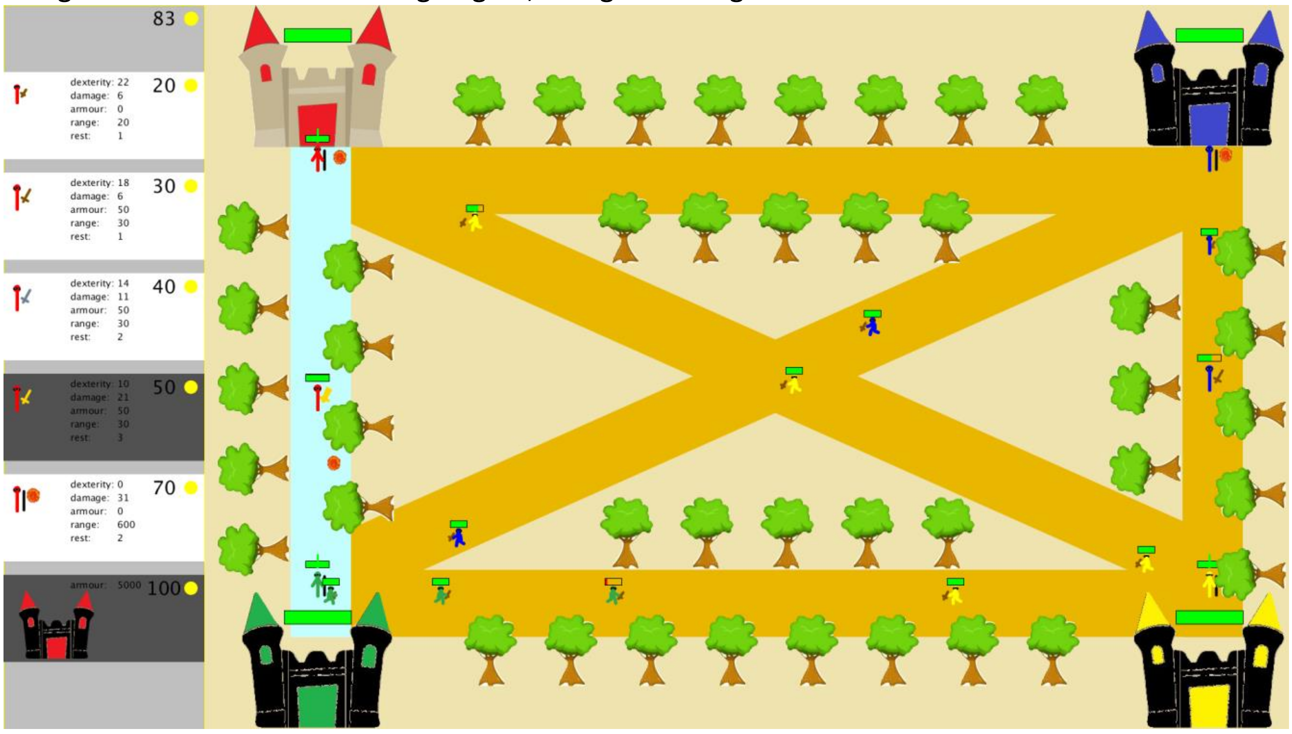
Player winning message:



```
420
dexterity: 22   20
damage:    6
armour:    0
range:     20
rest:      1

dexterity: 18   30
damage:    6
armour:    50
range:     30
rest:      1

dexterity: 14   40
damage:    11
armour:    50
range:     30
rest:      2

dexterity: 10   50
damage:    21
armour:    50
range:     30
rest:      3

dexterity: 0    70
damage:    31
armour:    0
range:     600
rest:      2

armour:  5000  100
```

Player 2 Won!

Full game with different battles going on, along with ranged fireballs and fortified towers:



83

dexterity: 22
damage: 6
armour: 0
range: 20
rest: 1
20

dexterity: 18
damage: 6
armour: 50
range: 30
rest: 1
30

dexterity: 14
damage: 11
armour: 50
range: 30
rest: 2
40

dexterity: 10
damage: 21
armour: 50
range: 30
rest: 3
50

dexterity: 0
damage: 31
armour: 0
range: 600
rest: 2
70

armour: 5000
100

Context

Real-Time Strategy Games

A real-time strategy game is one in which a player is in charge of defending their base and destroying their opponent's base by building an army given limited resources. In general, users are able to construct units on the map that give them a range of benefits (more resources, more defence, more attack), as well as choosing from a selection of soldiers/fighters that will fight for them, defend the base and fight off enemy forces, with the aim of destroying the enemy base. These fighters, once chosen and placed on the map, are generally not controlled by the player any longer and instead perform their duties on their own, whether that is collecting resources or attacking the enemy. The player is simply in charge of building the army and managing the resources, which all happens in real-time, as opposed to a turn based game.
"Utopia" was one of the first RTS games released in 1981, and it introduced the real-time element, but also included turn-based elements. It is widely considered the root of the genre, while there are other similar RTS games that were released at that time, such as Cytron Masters (1982) and BYTE (1982), which involve founding elements of RTS such as managing resources and constructing units. More recently, some of the most successful games/franchises in this genre are "Warcraft: Orcs & Humans", "Age of Empires" and "StarCraft". The first two are set in Medieval times, where Warcraft includes some fiction and Age of Empires is more realistic and historical, and StarCrraft is an RTS game set in space. These 3 games were hugely successful and are considered the benchmark RTS games released in the 90s, and are still played today. Age of Empires, for example, has released 6 more games since the original release. Additionally, the three games are similar in terms of what a player can do and the gameplay, but differ hugely in
feel and storylines. In particular, the AI in Age of Empires is renowned for being one of the first in RTS to be completely fair, meaning that the AI was not "cheating", given any additional powers or knew anything that the human did not – it played by the exact same rules as a human.

In terms of my game, the similarities are that the player has similar objectives – build an army, manage resources, defend your base (which is a castle) and defeat the enemy castles. It is also set in medieval times, hence the portrayal of castles for the bases, and swords/bow & arrows as the weapons the minions have. Similarly to these games, the player is in charge of choosing the minions in their army, and the minions then try and destroy enemy castles, whilst engaging in battle with any minions they encounter on the way. The minions are also not controlled by the user once deployed, which is a similar feature to these games.

Tower Defence Games

A tower defence game is one in which the player has to defend their base from waves of enemy forces, as opposed to RTS where the player can also attack the enemy base. They can do this by strategically placing objects on the "path of attack" - the path on which enemies come from - that are in charge of killing any enemies that come their way, in order to protect the base. These objects can be fighters, canons, weapons, etc., each of which provide different strengths and weaknesses.

This genre also originated around the 80s, with the first of these being "Space Invaders", where the player had to defend their territory against waves of enemies by strategically using shields and weapons. Other TD games like this from that time are Defender (1981) and Choplifter (1982), introducing the basic foundations of the TD genre. More recent, successful TD games are Plants vs Zombies (2009) and Bloons (2007). In Plants vs Zombies, the player is in charge of defending their garden (the base), using plants that can shoot or provide resources (weapons), against waves of zombies (the enemies). Similarly, in Bloons, the player has a certain amount of health, which is reduced each time a balloon goes past the end of the path of attack. To stop these waves of balloons, they can place weapons along the path, which are in the form of monkeys with darts, to pop the balloons. The AI involved in these games is much simpler than RTS, because the enemies must simply follow a specific path towards the user's base, and the user's weapons must simply attack the closest enemy, given their strengths and weaknesses.

The way my game is similar to these is that the player must defend their base, and there are only specific paths where the enemies can come from. This way, the player must strategically decide where to place their weapons (the minions) in order to stop these enemies. Similarly, the minions simply follow the path they are instructed to follow by the player, and engage in battle with enemy minions on the way, or attack the enemy bases once they get there. These minions have different strengths and weaknesses (and costs), so the player can strategically pick which minion goes where and when – another common feature of TD. Finally, the user's minions can't be controlled or instructed once they have been deployed, they are just controlled by an AI, which is a similar feature to most TD games.

Differences & Original elements

I consider my game to be a combination of the RTS and TD genres, leaning slightly more towards the RTS genre. This is because it is more a feature of RTS that the player also has to attack the enemy bases to win, as opposed to just defending the base as in TD. However, the fact that the minions are purely controlled by AI and follow a specific path, engaging in battle on the way, is more of a feature in TD. The most original elements of my game, that isn't seen commonly in either TD or RTS games, is the fact that the game is a "free for all" mode with multiple enemies (controlled by AI), which the player must fight against. This adds a whole new dynamic to the game, as the player is required to fight on multiple fronts simultaneously, and adds a new side to the strategies chosen by the player, as they can decide if they want to evenly spread their forces, or concentrate on eliminating enemies one by one. Meanwhile, the enemies are also fighting against each other as well as the player, so interesting dynamics and situations can arise.

Evaluation

Testing

For my evaluation, I tested my games a large number of times myself, and also had 3 of my friends play the game. I decided to not tell them anything about the game or how to play beforehand, so that I could watch them figure it out themselves, to see how intuitive my game was, and what aspects needed to be more obvious. After watching them play and answering their questions, I asked them a few questions about the game and noted their responses. The questions I asked included:

- Was there anything you didn't understand about the game, or was there anything that didn't make sense to you?
- Was the game too easy or too hard?
- How many enemies did you prefer playing against, and why?
- What did you think about the enemy players? Were they smart or did they make decisions you wouldn't have done?
- Did you enjoy the pace of the game? Was it too fast or too slow?
- Did you think any of the stickmen were overpowered?

Feedback

In general, the response I received from my friends was a positive one, with some constructive feedback. The game itself was quite intuitive and easy to figure out from the get-go, which I observed myself from the beginning and concluded since there was nothing my friends didn't understand how to do or didn't make sense to them. The instructions were clear, the rules were simple, and at the beginning, since the AI started playing straight away, they were immediately thrown into the game and had to act. The first time they played they were mostly just trying out the different stickmen/upgrades, until eventually coming up with some strategies to try and win.

From my first friend, after some time playing, I recognised and received feedback that the game's pace was a bit too high for an RTS game, meaning it would only take a few minutes for the game to end and the player's strategies couldn't develop well. Therefore, I decided to add more health to the player castles, reduce the stickmen's speeds and add a few more stickmen to the game, which I believe solved this problem. Now, after testing, some games can act a long time, and the tempo seems slower, meaning there is more time allowed for players developing strategies, and working together to take down enemies.

From the second friend, the main piece of feedback was that they found that the magic stickman was too overpowered – their attacks were too strong, too quick, and no other stickman could defeat them. They therefore felt that the game would go on forever. To fix this issue, I largely reduced the magic stickman's damage and health, and also added a rest period to their attack, meaning that they can only hit another stickman once every 3 times they attack them. Although this is still a very strong stickman to have to defend your castle, this definitely nerfed their powers down to not make them too overpowered.

Finally, the third friend had less feedback to give since I had resolved the issues stated above, however they did mention that sometimes the game could feel like it would never end, and so maybe a certain timed aspect could be introduce, such as things falling from the sky that could damage random stickmen and players, or anything of the sort. Unfortunately, I didn't have time to implement something like this, although it would definitely have added an extra interesting dynamic to players' strategies.

Conclusion

Overall, I am extremely happy with my game and its development. I have had a lot of fun playing it, and everyone I tested it on played it again and again, even after winning once, because they enjoyed it and wanted to try out different strategies. Although there are still some issues with some of the gameplay (some of the movement of the stickmen can be strange in some situations), I am happy with the level of difficulty that this game presented. The physics and the AI decisions of the stickmen was particularly challenging, as was the enemy AI to pick between moves and looking ahead to evaluate which moves lead to better states. I am especially proud of this AI, as all enemies don't make the exact same moves at all times (which would be boring), opt for different strategies, attack each other as frequently as the player, and don't just act randomly. This makes for a fun game in my opinion in which the player has to really be fighting on different fronts at the same time, and needs to carefully select when they start attacking different enemy players (maybe trying to team up with another enemy?). If I had had more time, I definitely would've loved to add more stickmen, and more defences to the castles (such as canons). Additionally, I would've introduced external factors to speed up gameplay when it takes too long, such as things falling from the sky randomly to damage the stickmen/castles.

Resources

https://en.wikipedia.org/wiki/Tower_defense
https://en.wikipedia.org/wiki/Plants_vs._Zombies
https://en.wikipedia.org/wiki/Bloons
https://en.wikipedia.org/wiki/StarCraft
https://www.youtube.com/watch?v=8ahIzcJS7n4
https://en.wikipedia.org/wiki/Warcraft:_Orcs_%26_Humans
https://en.wikipedia.org/wiki/Real-time_strategy
https://en.wikipedia.org/wiki/Age_of_Empires
https://www.ageofempires.com/
https://en.wikipedia.org/wiki/List_of_real-time_strategy_video_games
https://www.openprocessing.org/sketch/120612