Software Project Plan, Tower Defence

ELEC-A7151 - Object oriented programming with C++

Nico Renlund 715874

Oskar Holmberg 706595

Sandra Schröder 713274

Victor Mesterton 652885

Martin Sederholm 716080

1. Scope of the work

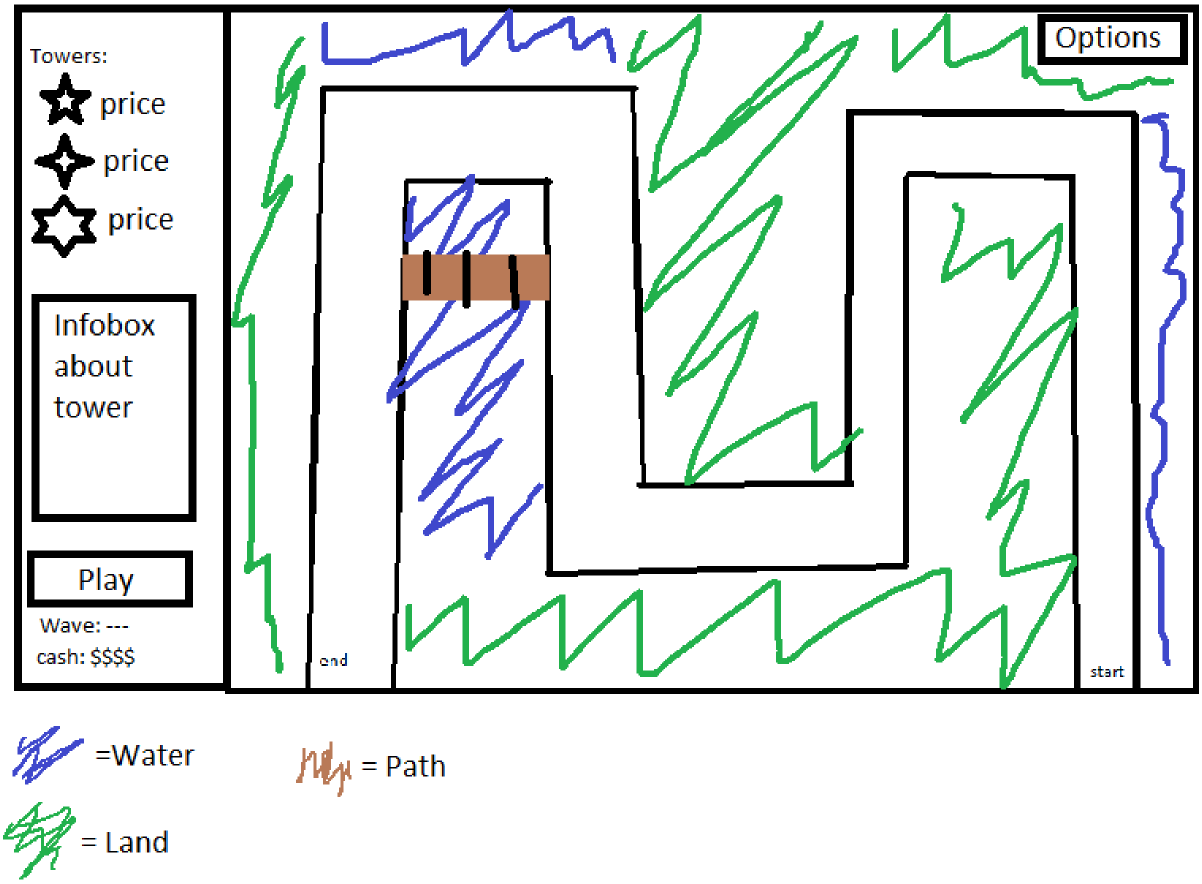
Our project is to develop a tower defence game. The objective of the game is to stop the enemies that are walking along a path to reach the finish. The path will be defended by “towers”, which will have different abilities/attack methods to kill the enemies. The enemies have hit points and different abilities, such as walking, flying or defence against certain forms of attack. If an enemy gets through will the player lose a life.

At the beginning of the game the player will have a small amount of in-game currency that they will use to buy the defence “towers”. The player will receive extra in-game currency from killing enemies and from surviving a wave. The cash will be used to buy more “towers” and to possibly upgrade existing towers to gain additional power and abilities.

The first round will be easy with just a few weak enemies. As the game progresses will each level become more difficult. There will start to spawn more enemies per round or more advanced enemies with more hit points and special abilities.

The game is operated using the mouse. The various defensive towers are placed on the gameboard by dragging them with the mouse and dropping them on an available tile. Furthermore, the mouse will be used to activate the next wave by pressing the play button.

The visuals of the game:
The game will be similar to the picture presented below. There will be a main window, with the path, towers and enemies. Then there will be a secondary menu window on the left-hand side. The secondary window will display the available towers, a small box with information, a play button and an information table, which will display the current wave and the number of in-game currency and remaining lives.



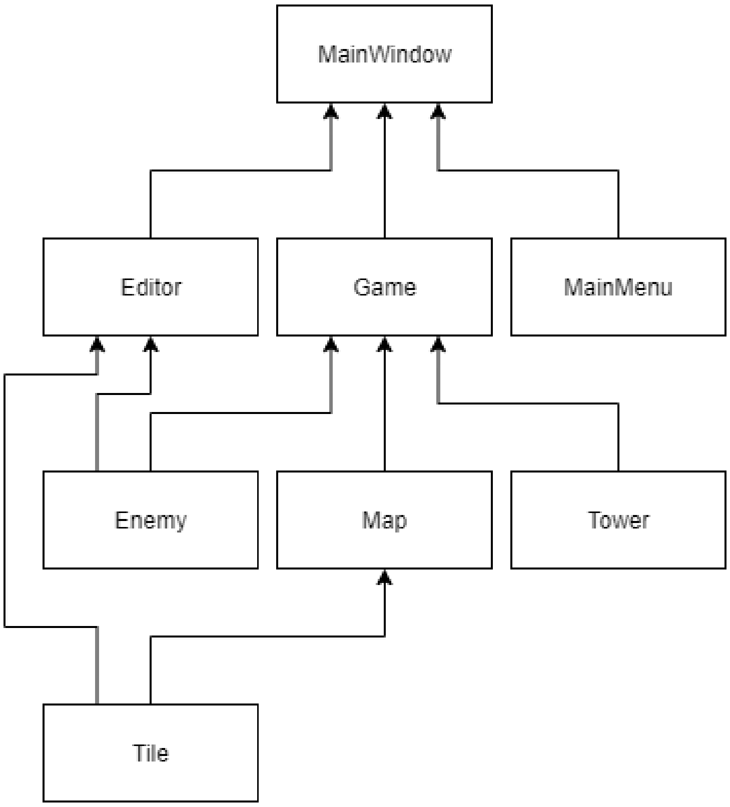
Features:

1. Sound effects (or background sound)
2. At least five different kinds of towers
   a. A basic tower, shoots enemies within its range
   b. A water tower, that can be placed it water
   c. A utility tower, that can generate money over time or improve towers that are nearby
   d. A tower that can only shoot flying enemies (kind-c)
   e. A tower that can attack multiple enemies at the same time
   f. A tower that can shoot all kinds of enemies (also flying)
3. At least four different kinds of enemies
   a. An enemy that's killed immediately when it's hit
   b. An enemy that takes multiple hits to kill
   c. An enemy that splits into multiple kind-a after killed
   d. An enemy that flies in the air

4. The path will not have branches
5. The player will only be able to set towers before the wave is ran
6. The map will be read from a file (non-hardcoded)
7. The towers will be upgradable
8. Towers can not be damaged by enemies
9. The game can only be played by one person at the time
10. The player will have a certain amount of lives and the game will be lost when he runs out of lives. Thus, the game will not be lost when the first enemy gets through.

2. Structure of the software

Architectural decisions:



Class division

The picture above explains our structure of the program. The Tower and enemy classes will have sub-classes for each variety of Towers and enemies.

3. Planned use of external libraries

After thorough consideration of different external libraries, we have come to the final conclusion that SFML will be the most suitable for our needs.

4. Division of work and responsibilities between the group

- Sandra makes and designs enemies
- Nico makes and designs towers
- Oskar will try to make the map and the map editor
- Victor will make the MainMenu
- Martin will be responsible for the sound
- We will all work on the game itself as well as the final touches and optimizing

5. Planned schedule

- Week 1:
    - Continue planning
    - Creating the towers
    - Creating the enemies
- Week 2:
    - Creating different kinds of maps
    - MainMenu
    - Editor
- Week 3 & Week 4
    - Making the game
- Week 5
    - Getting it all together
    - MainWindow
- Week 6
    - Final touches
    - Possibly adding extra features
    - Optimizing rewards, speed etc.