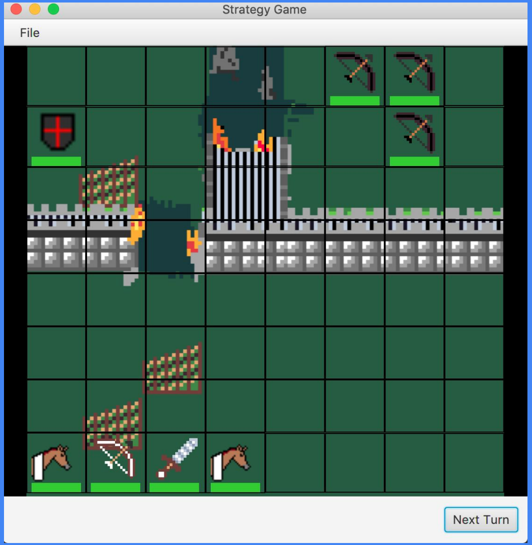
Knight’s Quest



Created by Drake Sitaraman, Ember Chan, Saul Weintraub, and Amber Converse

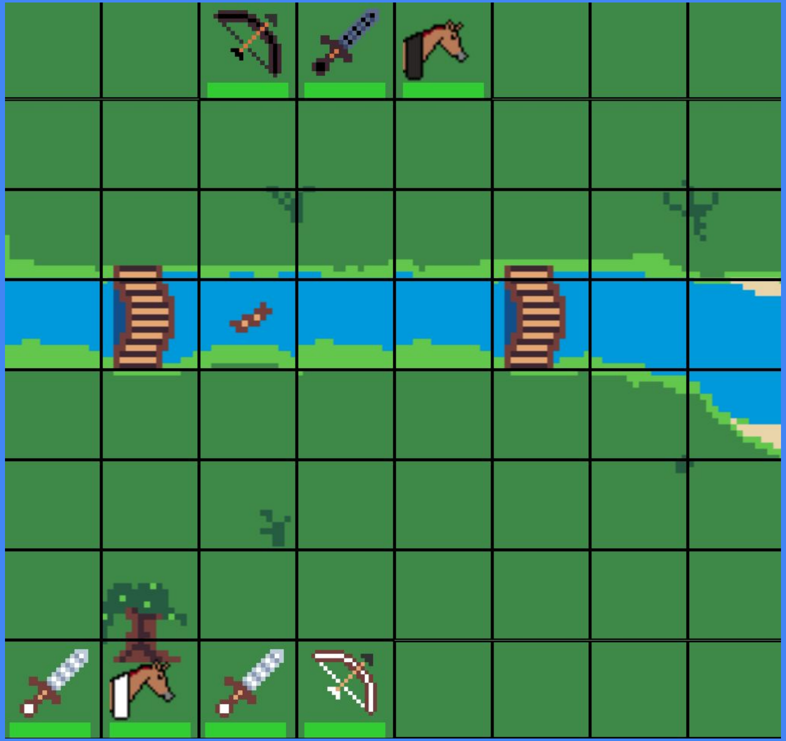
Main features

Main game

- Features two teams, the human and computer teams. Both can have the same pieces. The goal is to kill all of the enemies pieces.
- Each piece can either attack, defend, or rest as an “action”. Movement is independent of the action. Each piece can move its maximum move range every turn. Pieces will grey out when they are out of movement.
    - Attacking allows a piece to do damage to an enemy piece.
    - Defending allows a piece to reduce the amount of damage it takes if attacked.
    - Resting allows a piece to regain health each turn. However, **damage taken is increased by 50%** if a player is attacked while resting.
- New levels unlock when the user beats a level for the first time

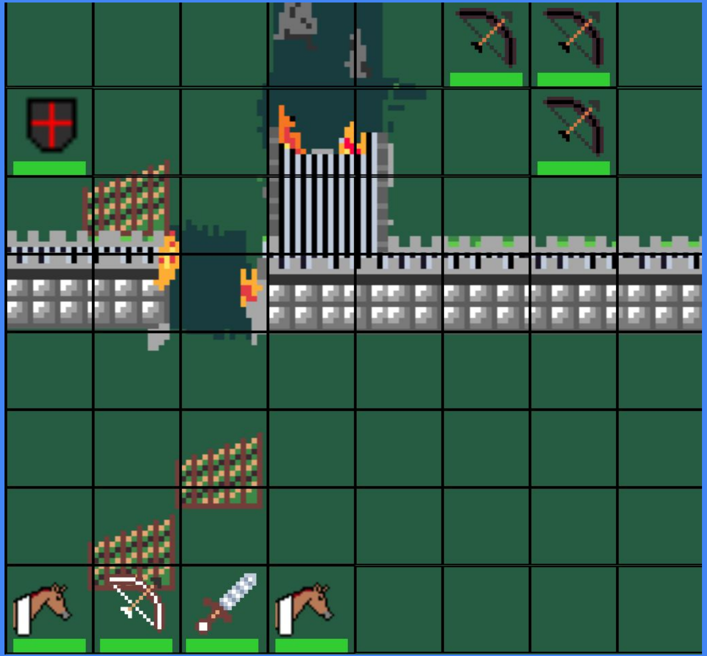
Level 1

A simple level with a river in the middle so there are only two ways to get to the enemy.



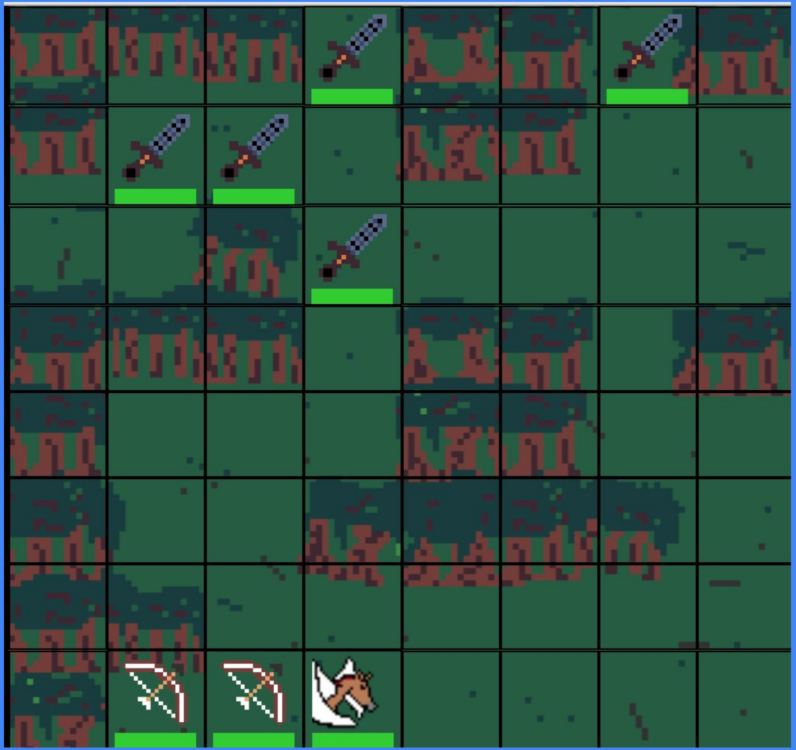
Level 2

A level where the enemy is holed up in a ruined castle, allowing the user to corner them.



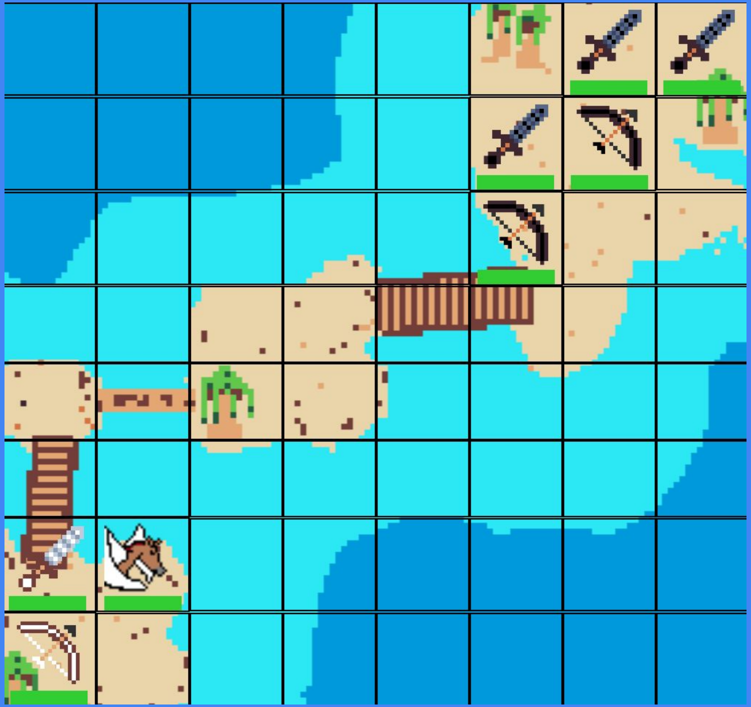
Level 3

A forest level where the user is outnumbered. Similar to level 3, the user must use their more mobile units to their advantage to defeat their opponent.



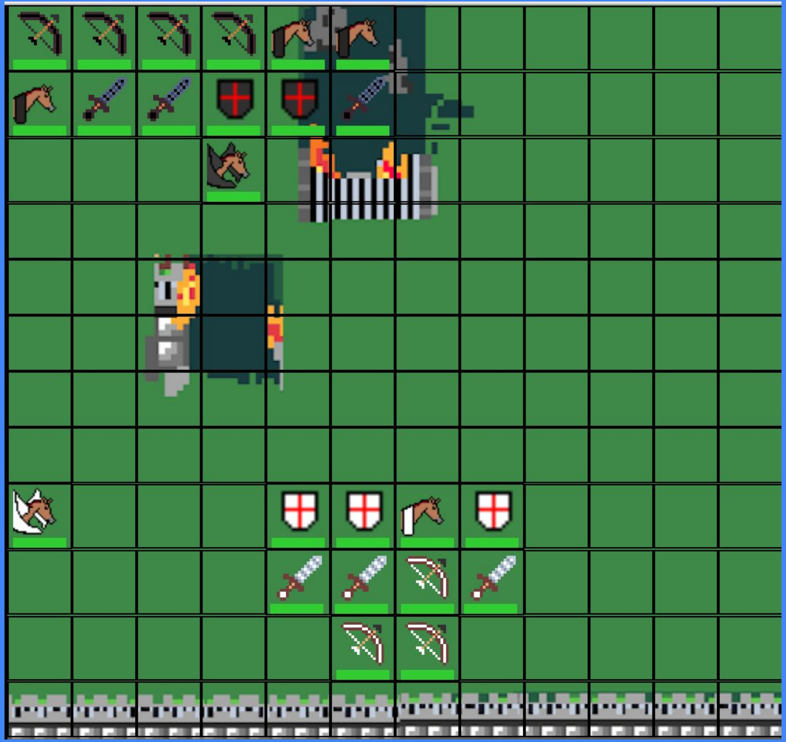
Level 4

A very cramped island level where the user is outnumbered. The user must use the Pegasus, a flying unit, to their advantage.



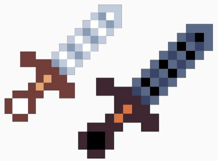
Level 5

The final level, a large scale battle where the user must defeat an invading army and defend their castle while outnumbered.



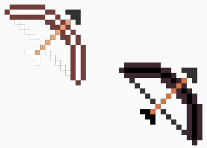
Pieces

Knight ***a slow but strong attacker***



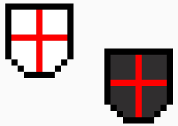
Damage: 20-30
Defense: 10-20
Movement: 1
Attack Range: 1

Archer ***a fast long-range unit, but with very low defense***



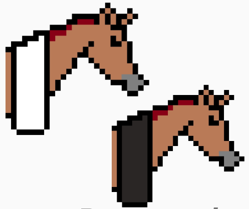
Damage: 10-15
Defense: 5-10
Movement: 2
Attack Range: 2

Armored Knight ***a slow and low-attack version of the knight, but with very high defense***



Damage: 10-15
Defense: 25-35
Movement: 1
Attack Range: 1

Horseman ***a very mobile unit***



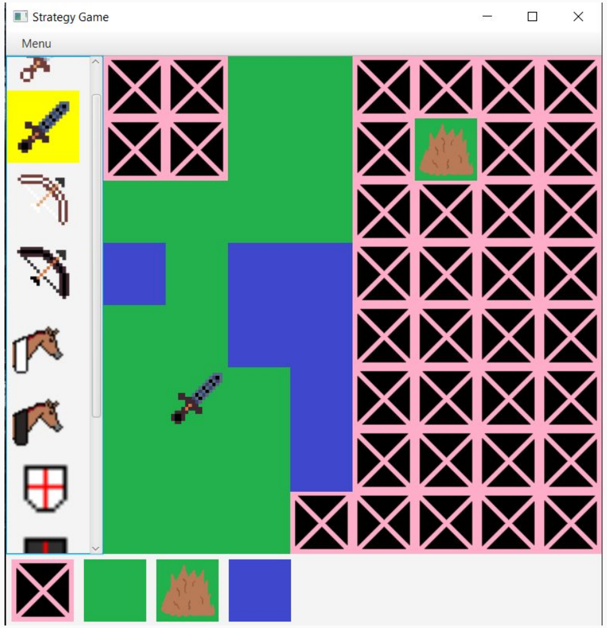
Damage: 10-15
Defense: 10-20
Movement: 3
Attack Range: 1

Pegasus ***the only flying unit in the game, can enter any tile***

Damage: 10-15
Defense: 0-10
Movement: 3
Attack Range: 1

Wow Factor: Level Editor

- Create, save and play your own custom levels!
- Custom levels have all tiles look identical instead of having a custom background.



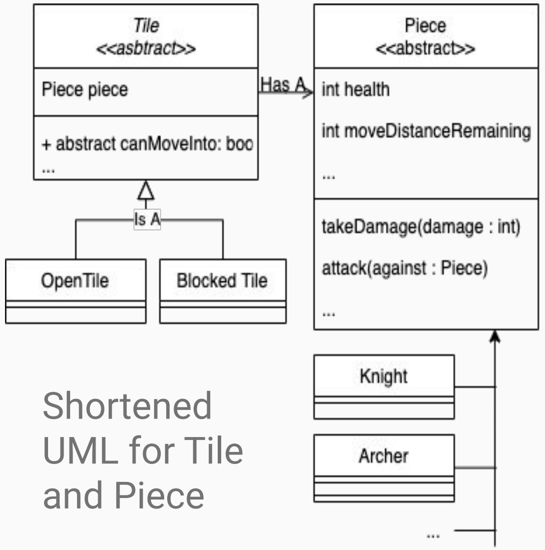
Design Choices

MVC Architecture

- The view is split into three parts, the main view (containing menus and encapsulating games and level editors), the game view, and the level editor view.
- The controller supplies the view with the necessary information to create a map for the user to see and receives commands based on user input to communicate to the model.
- The model stores the game as a StrategyGameState

The Strategy Game State

- Stores the game board as 2-D array of Tile objects.
- Tile is an abstract class which all Tile types extend, including OpenTile, BlockedTile, and BlockedSeeThroughTile.
- Every Tile has a Piece object, whether null (no piece on that tile) or an actual object.
    - The Piece object keeps track of its own stats based on calls from the controller (e.g. givenPiece.takeDamage(10) will subtract the 10 damage from the Piece's health, but also take into account any modifiers specific to that piece (e.g. it is currently defending))
- Every piece (Knight, Archer, etc.) extends Piece.
- Because Piece is an abstract class, all that needs to happen to create a new piece is to create a class which extends Piece and implements the unique piece's constructor (to determine its initial stats) and the abstract method resetTurn() (to determine its stats at turn start)



Shortened UML for Tile and Piece

Computer Player

- One of the key features of this game is the ability to play against a computer player.
- The ComputerPlayer class is kept separate from the Controller to improve modularity. It HasA Model, which it uses to get information about and update the board.
- The goal for the computer player is to get to the nearest human player as fast as possible.

Computer Player (cont.)

- While the board is stored as a 2D array of tiles, it can also be thought of as an undirected graph, where each node connects to its 8 cardinal directions.
- Because of this, the computer utilizes a Breadth-First Search algorithm to move, which guarantees the shortest path to the nearest human player.
- The computer moves as far as it possibly can, with the number of moves it has remaining.
  - If it is within attack-range of a human, it attacks.
  - Otherwise it defends
- This process is repeated for all computer pieces in a random order.
- With this algorithm, we have a fully-functioning, somewhat intelligent computer player!

Before Computer Move

After Computer Move