Senior Project Proposal
Hunter Stevens

Problem Description

Learning is key to progressing through life, but so many people think learning is boring and slog through whatever task is it hand or want to ‘learn’ as quickly as possible and have an AI do everything for them. Many people think this way about software engineering/coding. Thanks to the popularity of the field there are many who get pushed down this career path but don’t understand one of the core principals of what software engineering is: solving problems.

Proposed Solution

I will make a browser-based, programming, duel game to get people to engage with programming in a fun way that gets them thinking like a programmer. The game takes place in a 16x16 grid with two players. The players’ starting location is random every match. Each player must defeat the other by making a program before-hand from a list of predefined assembly-like instructions. Each player will have to think about the problem space and come up with the best strategy to defeat the other without having access to critical information before-hand, like their own location, or the location of the enemy.

The user will access the game through a browser; the player will have a list of 10 instructions to choose from. Each instruction has a description of what it does and any parameters the instruction takes. The player can build a program 20 lines long, a line being one of the 10 base instructions. Each instruction will allow the player to manipulate their character on the screen. Various instructions for attacking, defending, and moving will be included in the game. A few logical instructions will be included to enable players to make loops as well as conditional branches within their programs.

Once a program is built and submitted the server will run the program against the enemies’. The server will randomly choose a player to go first and start executing one instruction at a time from each players’ program. The match will be finished in milliseconds, but the results will be displayed in a slow playback manner to allow the player to see whats happening as the game progresses.

In this way, players will gain experience thinking like a programmer, by solving problems with the tools at hand. The simplified quasi-programming language in the form of the game instructions will hopefully allow for people with little to no experience in programming to think like a programmer and approach problems in much the same way a programmer does.

Technical Overview

I will be programming the frontend user-interface/website in HTML, CSS, and Javascript. I started learning Javascript in SE-3200 and have used it a decent amount since then. To make the site look better I will use the CSS framework bulma which I have been using for about a month. If I need a lot of Javascript on the frontend I plan to use Vue, just the simpler CDN version we used in SE-4200

For the backend/server I will be using go and any data storage will be handled with MongoDB. I have a little experience with MongoDB from SE-4200, but go will be brand new for me.

Milestone List

Dates are when these milestones should be completed by.

Week 1-2 01/17/25

Project proposal and presentation

Week 3-4 01/31/25

Go – Ten base instructions finished.

1. Fireball (AoE)
2. Lightning Bolt (Line Attack)
3. Move
4. Teleport (Random Location)
5. Shield (Protection From Damage, Specify what type fire/lightning/acid)
6. Divination (Find where other player is)
7. Time Jump (Jump asm)
8. Conditional Jump
9. Acid Puddle (Lingering Damage on the ground)
10. Wait
11. Loop (Start/EndLoop)
12. Invisibility
13. Recharge (recharge mana)

Able to run a (singular) 'battle program' in a simple, static game environment

Week 5-6 02/14/25

Go – Able to simulate an actual battle, two programs at once, ie one instruction from player 1 is executed then one instruction from player 2 is executed.

Get a simple server running that can receive a request. Example, server running that will receive data from the frontend (battle program) and send that data back to the frontend with a msg appended to it

Javascript – Simple text interface with text box that will send a battle program to the server and display the returned and modified program on the page

Week 7-8 02/28/25

Go – Server able to run many battles back to back. Connect the server to MongoDB so that data persistence is achieved. Enable the creation of users so that programs will be tied to each user.

Week 9-10 03/14/25

Javascript – Get simple (visually) interface working; able to create a user, make a program, and have that program saved to that user (able to retrieve that program later from the DB)

Display the results of a user's battle (Arena and list of actions that occurred in order from beginning to end

Week 11-12 03/28/25

Javascript – Make the battle replay graphical. Add the ability to see multiple battles by clicking through a list of replays.

Week 13-14 04/11/25

Javascript – Polish frontend visually
Backend – ?

Week 15-16 04/25/25

Project Over/Final Presentation

Validation Plan

I will let users try and build a program and play the game as soon as possible. If they can't figure it out or understand I might try to make a better tutorial or simplify or change the instruction set.

I want users to be able to read a tutorial of the game overview and start experimenting with the various instructions the game provides and see what their character does in game and refine their instruction set.

If a brand new user starts playing the game and can make a program that does something within a few minutes and gets enough excitement from that to modify their program or make another, I consider that a success.