Web Development – Mr. Turner

Project – One on One Fighter

Project Overview

Two fighters go in. Only one comes out. Depending on your interests, they can be any type of fighters. Gladiators, pokemon, superheroes and villains, the real housewives of New Jersey. It's up to you. A single player will take on the computer in a round by round combat until one of the combattants is knocked out.

On his or her turn, the fighter will choose to either attack, defend, or attempt a finishing move. The results of each of these moves will be based on the situation of the fight and stats of the fighter.

Display

The list below includes the essential elements of the page.

- There will be two fighter sections, one for the player and one for the computer.
    - Each fighter section will have a log of the moves made by the fighter.
    - Each fighter section will show the original and current values of each of the fighter's stats (Strength, Speed, Cunning, and Fatigue).
- The player's section will have a button each for Attack, Defend, and Finishing Move.

Functionality

Each fighter is defined by 4 numerical values:

- Strength - used for attacks
- Cunning - used for attack and defense
- Speed - used for defense
- Fatigue - used to determine how close a fighter is to being knocked out.

Each fighter begins with strength, cunning, and speed at values of 6, and a fatigue of 30.

Before the fight begins, the computer will generate two new fighters, altering the base values in the following way:

- It will choose 2 values to increase and 2 values to decrease.
- It will modify strength, speed, and cunning each by a 0 or a 1.
- It will modify fatigue by a random number between 0 and 6.

The modified values should be considered the original values for each fighter.

During the round, each fighter must select whether to attack or defend. Under certain circumstances, a fighter can also attempt a finishing move. The user will choose by clicking a

button. The computer will choose to attack or defend randomly ***unless*** it has the finishing move available. Then it will always choose the finishing move.

When a fighter attacks, its attack value is determined by the following:

Attack Value = (Strength + Speed + Cunning) / Random Integer from 1 to 3

This attack value is compared to the opponent's Defense Value. The Defense Value is determined by one of the following formulae

If Defending...

Defense Value = (Speed + Cunning)

If ***NOT*** Defending

Defense Value = Speed + Random Integer from 1 to 6.

If the attack value is greater than the defense value, the opponent's fatigue is reduced by the difference.

Since both fighters are choosing to attack or defend at the same time, the moves are simultaneous. The results of each attack must be calculated ***before*** *the final changes are made to the fighters.*

If a fighter defends and does *not* take damage on that turn, it will recover 1-6 random fatigue points. ***Fatigue may never rise above the original level, but may go below 0.***

If both players defend, then each gets a random number of fatigue points (1 - 6) back.

Update the display accordingly. All current and original values for both fighters must be visible on screen at all times.

In addition to Attack and Defend a player may attempt a **Finishing Move** under two conditions.

- If one fighter's fatigue is at least double the other fighter's fatigue, it may attempt a Finishing Move.
- If a fighter's fatigue is below zero, its opponent may attempt a Finishing Move.

A Finishing Move is a standard attack, *but Cunning is not calculated into the formula*. If it succeeds (would do more than 1 damage), the fighter performing the Finishing Move is the winner and the fight ends.

The Finishing Move option should only appear in the menu if a fighter is in a position to perform that action. Otherwise, the fighter will only be able to attack and defend. This may change multiple times during the fight.

Enhancements

- A fighter weakens as its fatigue decreases. Each time a fighter loses 5 fatigue, reduce its battle values by 1. If the fighter should regain fatigue, then the values should increase accordingly.

Necessary Programming Skills

- Comprehension of the specifications sheet.
    - This particular project has a lot of nuances that are often overlooked.
- Planning
    - Figure out the information you need to keep track of.
        - This information will become your global variables.
    - Plan out the individual tasks your program must perform.
        - Think through the steps for each task.
        - Think through the information your task needs (where does it come from?).
        - These will become your functions.
    - Plan out the user interface.
        - You can start with the barest interface, but you should have an idea what you want the final product to look like.
- Managing your variables
    - What's global, local, and passed through as parameters (hint - this program can use all three)?
- Assigning functions as tasks
    - Does your program sort out the different tasks into their own functions?
- Sequencing
    - Does your program sequence from the user interaction into the necessary functions?
    - Is there an efficiency to your code that flows from the design document?
        - There are several steps executed each round of the fight. Does your program manage those steps in a way that works efficiently in your code and provides a meaningful and understandable display for your user?
- An intuitive user experience
    - Is your display appropriate to the program (what's viewable and what scrolling has to be done)?
    - Is your display adaptable to other resolutions?
    - Is the interface intuitive?