Preston Smith

CISP 1020

DR. Brown

PLAN

Board- The board will be a class that has a drawable texture, an overlayed array of particular positions so that the player can be notified

of their location on the tile( based on a fraction of the screen EX: area 1/4, 2/4,4/4,3/4 etc etc).

Player- Inherits sf::Sprite class for texturing , movement, and getting position. Health in binary-

starting at 100 ( will be expanded through power ups). Damage will be applied through bitwise operations.

Player will hold a reference to a base texture, and a base position.

The player class is being built is a way as to be modular and inherited-

by other classes ( such as the enemy class that will be implemented later);

Physics- Physics engine will be implemented on all player objects, and any moving drawable.The engine will handle : gravity, velocities in x and y. collisions.

RigidBody - Rigidbody will be applied to all objects that need to be solid or need to handle collisions. Collisions will be detected base on the touching of rigid bodies, and effects of the rigid bodies touching will be handled by the physics engine class.

Networking - Networking will be handled later with a raspberry pi 3 with http requests, running on a small local ip with a Nodejs api.

AI- ai will be handled with an AI controller class . The ai will have different difficulties. Hardest ai will do everything flawlessly

in combat. every player will have some random attributes, such as dodge%, accuracy, etc.

Medium difficulty will make some mistakes, but for the most part will make the smartest decisions.

Easy will be near pure random abilities and combat.

Animation- animation will be handled with an animation engine. animations will be stored in a doubly or circularly linked list, that way

the list can be iterated through for easy animation.

Gameplay- Gameplay will be a mario like, objective based platformer, with random power ups, chests, items, and small combat. Every round

will have many small fights vs weak players, and a boss battle. There will also be random encounter battles, and opportunities to battle

for power ups, that if lost will also make you lose stats + a random power up you already have. Gold will be gained per world, that way

you can make in game transactions for items necessary for the game : powerups, food, potions, weapons.

PVP: pvp will be added to the gameplay for local machine pvp, and online pvp. PVP in the game will entail turn by turn combat, use of power ups. An energy system for using abilities. There will also be critical hit chance, critical hit modifiers, accuracy modifiers, evasion modifiers. There will also be damage stats, defense stats, energy stats.

Customization: Characters will be customizable in minimally. The color can be adjusted for clothes, some attacks(possibly), and possibly some game structures/buildings. ( this part of my plan is very rough, but hopefully doable.

DATA STRUCTURE USES

Doubly Linked List && Circularly Linked List : these two data structures will be used for linking animations to the next, this way animations can be does easily and simply, however possibly inefficient.

STACK: A stack will be used to order tilemaps, this will be dynamically filled , thus every round will be different than the last.

QUEUE : players will have the option during combat to create a “macro” for what attacks they will use in the following turns ( one attack per turn). This allows for a more cinematic gameplay versus a turn by turn combat that leads to constantly pausing fighting for you to input an attack.

MAP: I will use maps for being able to link ID codes to items and powers ups for easier lookups, queries, drop tables. It will also mean I will not have to use some kind of branch statement to link a number value to a string or object, I can just search a map .

OTHER DATA STRUCTURES: I have no idea what the other structures do, or really how to use them.

GOAL FOR THE NEXT TWO WEEKS

My goals for the next coming weeks are simple. I want to finally implement some basic textures for allies and enemies. I also want to add to week 3’s challenge of drag and drop tilesets. And something I plan to start the next coming weeks is creating a parser and syntax for saving the tiles sets generated in edit mode, so that they may me created and loaded from a simple file. This will make level generation faster, but possible lower the speed at which levels load.