ACTION SCENES AND EQUIPMENT

**Equipment**

Equipment is vital to the function of action scenes. Equipment may provide additional move options during an action scene. Items may also increase the number of times that one may be hit before losing an actions scene. For example, the Family Agent Armor that the protagonist starts off with can come with the move quicken, which increases the chance of success for the next movement related move. This piece of armor also comes with 1 hp, meaning that the protagonist can take one extra hit due to the armor before the player loses the action scene.

Each piece of equipment comes with the same set of stats that in the base class (introversion, extroversion, sensing, intuition, feeling, perceiving, judging, thinking). During an action scene, the stats of a piece of equipment can alter the outcome of a certain move (for more on moves, look at action scene info).

The protagonist, being a member of the Family, is obligated to wear the armor of Family Agents. The headgear, armor, and feet pieces for the protagonist are fixed throughout the game. However, as the game goes on, the player has the option of upgrading this armor and changing the moves associated with it. This can occur at the end of an arc or by talking to certain characters.

On the other hand, weapons are interchangeable and not upgradable with the exception of the palm rifle, which the protagonist receives in the intro mission. That is built into the Family Agent armor. Additionally, the palm rifle serves as a reliable and constantly available weapon, sort of like a default weapon in a shooter game.

**Action Scene Moves**

Action Scenes are won or lost by executing moves. Based on their equipment, weapons, party members, and previous arc rewards, the protagonist can enter the action scene equipped with a variety of moves for the player to choose from. When each move is used, there is a chance for that move to succeed or fail. The various variables of the action scene are tallied to determine how difficult it would be to accomplish the move. The stats of the protagonist which represent personality and skill, along with their equipment and weapons, will determine whether the protagonist is able to successfully execute the move despite the difficulty based on the conditions.

-most moves will increase or decrease the progress bar, though how much they increase or decrease the bar is dependent on the type of action scene (combat or chase)

-moves may come from equipped items

-moves may also come as a reward for completing certain arcs so that they result in a certain ending (example: resolving an arc in favor of a character in it who is a soldier might result in a reward that is a combative move)

-general method of a move

1. Calculate difficulty of the move

-uses an algorithm/mathematical function that is unique to the move

-takes into account the current conditions of the action scene

2. Calculate the skill of the protagonist

-uses an algorithm/mathematical function that is unique to the move

-takes into account the current stats of the protagonist including the bonuses given by equipped items

3. Generate random number

4. Subtract skill of protagonist from difficulty of the move and compare number generated from step 3 to the result or add randomly generated number to skill of protagonist and compare to difficulty of move.

5. If the result is greater, then the move was successful. Depending on how much greater one can define different levels of success. If the result was lesser, the move was unsuccessful. Again, depending on how much lesser, one can define different levels of failure.

6. Output text relating to the result determined in step 5. Depending on level of success or failure, the output text will be altered.

**Action Scenes**

-to start (same for combat or chase)

-set variables

-set conditions

-set environment conditions

-set actor conditions

-prepare player info

-determine hit points and moves

-prepare opponent and opponent info

-create opponent object via baseclass or use an already existing object for frequently used models

-give opponent appropriate items (this will be determined by story)...if using existing object then disregard this

-display blurb about win condition?

-in combat, hitting is primary, movement is secondary...hitting will result in larger changes to the progress bar than movement

-in chase, movement is primary, hitting is secondary...movement will result in larger changes to the progress bar than hitting

**Combat Action Scenes**

-for each turn (player)

-display moves for player to choose from

-wait on input from player (select whichever move)

-run move function (the following indented will be handled by the move function)

-calculate difficulty of the move...difficulty will be uniquely calculated for each move based on the current conditions of the action scene

-check if move is successful or how successful move is...some moves can be more or less successful while others are either successful or failed

-take appropriate action if move is successful...this often means altering the progress bar but it might mean something else like changing stats or for ex, Scan, if done successfully, would display info about the opponent

-output text to describe what happened (we could write this to be a bit random)...think pokemon, “it was not very effective...” but maybe a bit more dynamic to add some flavor by changing the text a bit based on how successful the move was or the current conditions of the scene or both

-output more text if need be, like a summary of what just happened?

-pass turn to opponent

-for each turn (opponent)

-using random number generator, determine move to use

-run move function

-calculate difficulty of the move...difficulty will be uniquely calculated for each move based on the current conditions of the action scene

-check if move is successful or how successful move is...some moves can be more or less successful while others are either successful or failed

-take appropriate action if move is successful...this often means altering the progress bar but might involve temporarily changing stats or something

-output text to describe what happened (we could write this to be a bit random)...think pokemon, “it was not very effective...” but maybe a bit more dynamic to add some flavor by changing the text a bit based on how successful the move was or the current conditions of the scene or both

- output more text if need be, like a summary of what just happened?

-pass turn to player

-win by hitting the opponent until they don’t have anymore hitpoints

-lose if opponent hits you for all your hitpoints

**Chase Action Scenes**

-distance between actors determines the win/lose condition

-moves play out very similarly to combat action scenes

-however, the result of a move will be handled a bit differently...the progress bar will still be altered, but now it will be based more on distance as opposed to hits

-if the actor who is chasing hits the actor who is running, the distance between them is decreased

-if the actor who is running hits the actor who is chasing, the distance between them is increased

-if running away, if distance is greater than certain value, player wins

-if chasing, if distance is less than certain value, player wins

-the value to win is determined by story and hardcoded