Project Final:

Team:

Brandon Gomez (brlgomez)

Ricardo Murillo (rmurill2)

Antony Robbins (androbbi)

Project Title:

Random Boss Generator that then gets evaluated based on the tools of the player.

Theme:

Both AI as Design and AI as Boss Generator

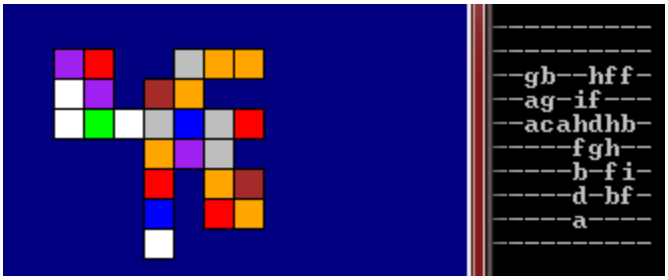
Overview:

For our project we created a program that generates a boss and evaluates it. The boss is creating by first collecting a random assortment of points on a grid, forming a skeleton for the boss. Each of these points is calculated using the previously calculated points, in order to assure that the boss is connected. From this skeleton, the points are populated with random squares that represent pieces of the boss. Each of these pieces have different properties points that are calculated by the evaluation later on. These properties are meant to represent things such as weaknesses or strengths, with the smaller number being strength and the bigger number being weakness. At the moment the only properties at present are damage multipliers, so the evaluation finds the average damage multiplier for the entire boss, and uses this to evaluate the difficulty

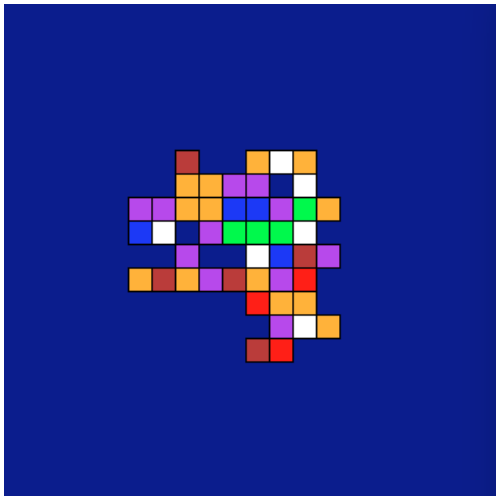
The intention of this project was to create a boss that is built using pieces and fragments that are weak or strong to the player's current toolsets. For example, in response to the player's bow and arrow, a boss might have a single portion weak to arrows and other parts that are immune and attempt to cover the weak point. As the player accumulates different tools and objects, the bosses grow in complexity and become more challenging, but still possible to defeat.

A bit of summary for the pictures below, each color represents a boss part with different weakness values. These values are shown in the last page as a reference. Each letter represents one specific color.

Memento:

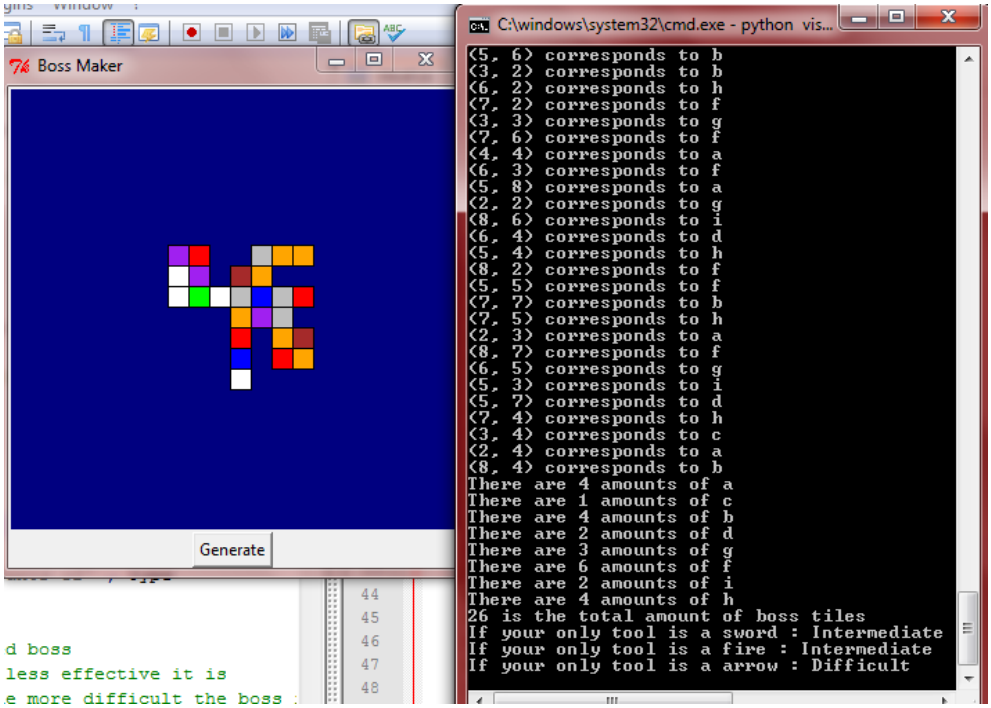



```
Library = {
  'Initial': {
    'width': 30,
    'length': 20,
    'body':{}
    },
  'Types': ["a", "b", "c", "d","f","g","h"],
  'Props': {
    "a" : {
      'sword': 2,
      'arrow': 1,
      'fire': 0
      },
    "b" : {
      'sword': 0,
      'arrow': 0,
      'fire': 3
      },
    "c" : {
      'sword': 1,
      'arrow': 1,
      'fire': 1
      },
    "d" : {
      'sword': 2,
      'arrow': 4,
      'fire': 1
      },
    "e" : {
      'sword': 2,
      'arrow': 1,
      'fire': 0
      },
    "f" : {
      'sword': 0,
      'arrow': 0,
      'fire': 3
      },
    "g" : {
      'sword': 1,
      'arrow': 1,
      'fire': 1
      },
    "h" : {
      'sword': 1,
      'arrow': 4,
      'fire': 1
      }
   },
  'Inventory': {
    "sword",
    "arrow",
    "fire"
   }
}
```



```
0% - 33% Beatable: Difficult, 34%-66% Beatable: Intermediate, 67% - 100% Beatable: Easy
sword { score} 31 {out of} 92 : 33.69565217391305 %
fire { score} 77 {out of} 138 : 55.79710144927537 %
arrow { score} 42 {out of} 184 : 22.82608695652174 %
```

A boss with stats indicating how much a tool affects it.



We also have analysis on the lower level design of the boss. Which part goes where, how many of each part there is, and how many parts there are total. For example, this information can assist an artist design bosses with pixel art with coordinate information.

Novelty:

Unlike the shooter boss example we had in class, which was based off of the user's methods and strategies, our boss designer is more of a 2D Zelda styled game where the bosses will be based on the player's tools or inventory. In the Zelda game, A link to the Past, the player can enter any dungeon in any order thus creating very generic bosses that can be beaten mostly by in-dungeon tools and the sword, with our tool a boss can analyze the tools to make a boss that utilizes all the players tools.

Value:

There are many possible uses for this type of design tool, these uses include: A way to prevent sequence breaking by having bosses generate dynamically, a more challenging yet satisfying boss experience that will always have a win condition, and as a design tool intended to create example bosses for programmers and designers. Can create other dynamic bosses or enemies for other games like Space Invaders, where player will have appropriate tools or attacks for every enemy on screen.

Technology:

The project was done exclusively through python, with the visual elements done using the TKinter library. This proved to make the program more visually appealing then using the block of letters we started out with.

Breakdown:

We planned on working through this project together, but due to the difficulty in finding times where we could all be together to work on things we split the work into three parts. Brandon focused on altering how the program generated bosses and implemented a flexible way of altering the boss like making it skinner, wider, bigger etc. Ricardo focused on working with the TKinter library and visualizing the boss's design, and Antony worked on the project's evaluation and analysis functions. The intention was also to create a game around these, but we weren't able to finish it before the presentation.