Alien Vengeance: A Python Programming Project

Final Report

Submitted to

The Faculty of Operation Catapult XC

Rose-Hulman Institute of Technology

Terre Haute, Indiana

By

Group 38

Tim Borgen The Fulton School  
 Rockwall, Texas  
 Darrington Altenbern Ensworth High School  
 Nashville, Tennessee  
 Cale Stallard East Richland High School  
 Olney, Illinois

July 3, 1998

For the budding programmer, the Python programming language is a unique tool for building coding skills as well as developing useful and practical programs. Offering a wide variety of simple functions and methods (a set of code that can be called to execute a specific function) that can be used as building blocks to create fascinating applications for a user, Python was a suitable choice of programming language to utilize for the creation of a new Python program that would be useful to a user. The next major step was determining the function of the program to be created. A wide variety of topics were brainstormed, ranging from a client-server application such as a website, to a multitude of video game themes. We quickly sifted out a video game program from this vast list of possibilities. Eventually, by weighing out the pros and cons of these topics, using a plethora of criteria including difficulty, practicality, and entertainment value to a user, we arrived upon the idea for a retro style video game that hailed from past Mario games. In our decision process, we rated our choices most heavily on a delicate balance between entertainment value and difficulty. Entertainment value was important because a user wouldn’t waste time with a dull and boring game. However, the limit of our entertainment value was set be how difficult the coding would be for the program, as none of us possessed any prior computer programming knowledge in the vast and mysterious plains of Python programming. The Mario style structure, with a linear level within which a character jumps up and down platforms and slaying monsters, was a perfect compromise between these two requirements. Basing the game’s style upon a previously established, classic game cemented the idea in the practicality field, as users established connection with the greatness and fun playability of retro Mario games.

The game itself took a large amount of concept development time, as the game turned out greatly more complex than we previously imagined, but still manageable. The most important decision in the design process came in choosing a protagonist for the game, the character the user would control throughout the game. As none of us possessed any artistic talent, we turned to the internet to supply us with a set of images useful in programmed games, most commonly referred to as sprites. A sprite is simply a small image useful to programmers because most sprite groups have most of the images for a moving sequence such as jumping or running. We were challenged in finding an interesting looking character while also being able to obtain all the images necessary for a smooth movement, and we settled upon a small alien creature (see Figure 1). The sprite sheets we found for this character contained every animation that would be necessary, including a variety of more we would never need. Because the sprites we found also included images of the alien shooting, we agreed that the character would kill enemies by shooting them, rather than jumping on them like the old Mario games. The second important part in our sprite hunt was finding images of the enemies that the hero would encounter throughout the game. Fortunately for us, we had many more enemy sprites to choose from, and we eventually decided upon a small bug, a flying dragon, and a “boss” warrior creature. We also searched for a sprite of blocks that made up tiles in the old Mario games, but it turned out to be incredibly difficult and time-consuming to tile these in to every platform, while setting up collisions for each block. As a result we determined that our platforms would be mono-colored rectangles, which were much easier to program. The controls were very easily to assign: The left and right arrow keys would move the character in the respective direction, the up arrow key would allow the character to jump, and the spacebar would shoot bullets to kill enemies. With the sprites and controls out of the way, we set upon the daunting task of programming our dream into reality.

Our programming journey can be divided into three major steps: learning Python and pygame (a graphics library consisting of a variety of useful methods), creating classes for each object in the game, and making it all flow together into a functioning game. The first step took up the majority of the first week of classes as we experienced a crash course in the intricacies of the Python programming language. We learned a multitude of useful programming methods, including Classes, Definitions, “If” statements and drawing shapes and images in pygame. Classes allowed us to create a set group of actions for each enemy and object including the player in the game in order to allow us to easily create a variety of each object without having to type in lines and lines of code each time and allowed us to group them into a group of objects for easier reference. Each class had to contain many definitions for each different movement the character had to perform, including moving left and right, jumping and being able to shoot in the case of the player, and most importantly being able to collide with other objects in the game. Colliding was the most significant challenge we faced throughout the development and coding of our game. The second step in game design was