Final Project Summary: Tony Gu and Daniel Trinh

This Project consisted of our recreation of the popular platform shooter game Contra by Konami. Though this game was quite difficult to recreate due to the amount of features that this game included. Among the main features we first had to implement was the moving background. Due to the fact that our version of Contra contained two different Maps that were very large compared to our figure window. We had to move the background as our character moved. To do that, we put a velocity on our background that would come into play when the character reached a certain position in our figure window. We accordingly adjusted the character, so that he could not move past the middle of the figure window (in order to keep the background movement speed in check with the character’s speed). That leads me to our next major task, which was to create a moveable character. In order to use original images from the game Contra, we had to upload jpegs of the original character and his various movements. We photoshoped all of the images and made their backgrounds transparent before we did this of course. In order to make him move we had to take many variables into account. For example, we had to take the character’s orientation into account (whether he was pointing forward or backward). We created a Keyboard Callback Call based on the wasd movement controls, j to jump, k to shoot, and b for bomb syntax. The code for this was intensely dense. We had to account for the guy when he moved forward, backward, into prone position, and into jumping position. When the guy was in jump mode, we had a different set of movement abilities for him. For example, when in the jump case, the guy could move around. In addition, when he jumped we made his appearance exactly like the original Contra in that he would appear to roll around in the air until he returned to the ground. We cycled through many different images to do this. Our next task was to make him appear to move his legs as he moved across the screen. This required cycling past different pngs of his various movements, which gave the semblance of him moving. The next large task was to create enemies that would try to kill the character in the game. In order to do this, we uploaded 11 different enemy pngs and plotted them off screen. Then we spawned them using the idea of the original Contra spawning system. We set a timer for each kind of enemy (Gunner, left approaching enemy, right approaching enemy) and when this timer reached zero, each enemy would respectively have a chance to spawn. We used a rand seed to do this. Every enemy had a different odds of spawning. We updated the enemies position every time the while loop ran through. If the enemy left the screen, the enemy would be reset back to its original initial plotted position. This way, the enemy could be reaccessed again by the spawn function. The enemies approaching from the left had lower spawn frequency. The gunner had the lowest spawn frequency because we made him fire a bullet (image of a bullet) every time he spawned and moved to certain part of the screen. After he fires the bullet and does not die, he move back off the screen like the original game. In order to defeat these enemies we create a shoot function for our character. We plotted all player bullets on screen and brought them up whenever the player clicked the shoot gun button. The bullets “recycle back off the screen” like the enemies. Finally, we had to create a collision function in order to have these enemies die. We used 2d collision state if statements to check if each bullet had hit an enemy. If the enemy was hit, it would return to its original off screen position. We did this for all enemy types. We did this as well for the enemies colliding with the player, which would result in the player dying. Lastly, we did this with the gunner’s bullet and the player as well. Every time the player died, the player would lose a life a respawn in his according position on the figure window. Note that we made heavy use of axes position, in order to make the collision states simple. Some other features to discuss would include the platforms. In order to make the character able to jump on other platforms, we took into account each platform’s position relative to the background and its width and length. Then whenever they collided with a platform we would make them enter a “platform state.” This would essentially force the character to land on that platform. To do this all, we created a gravity state, which was always on unless the character was on a platform. In order to find all the platform positions, we used the guess and check method for each of our two maps. Another miscellaneous feature was our menu. This was a UI and contained a customized title image (thanks to paint and Photoshop). We had 5 different options on our menu: Start game, exit, options, credits and help. Each had their own customized features. Notably options has a slider that allows you to change bomb and lives amount. For sounds, our menu had a title sequence song (Highway to the Danger Zone). In addition, we had sounds for deaths, enemies killed, and bombs. There are a few unusual additions we had to the game including the wall that blocks your path in level 2. This wall must be shot several times before you can move along the rest of the map. This was our emulation of the same feature in the original Contra. We intended to create an explosion to signal this wall being brought down, but we ran out of time. In addition, in level one, the enemies spawn in different parts of the map. At one point, they will spawn at one y position and at another one, they will spawn at another y position and at another point, they won’t spawn anymore. We did this by assigning different enemy spawn points for different points of the background which we used by incorporating the current position of the background into the if statements. When the character reaches the end of the first level, the next level automatically loads and clears all old variables except lives, bombs, and score. Also, we have a highscore feature, every time an enemy is killed a score is tallied. The points earned per enemy is based on the enemy. The gunner Is worth more, At the end, if the highscore is beaten, it is saved and a highscore screen is displayed. This is like the gameover screen. The Bomb button “b” kills all enemies on the map (Resets all enemies’ positions). This covers almost all of the features of the game. We may have missed one or two, but that is because this game was a massive task. We hope you enjoy this game!