**Task 5**

**Group: Isaias Leos, David Amparan, and Alex Vasquez**

Add Super Strength Functionality to the Pistol Weapon Functionality Your task is to add the ability to toggle the super strength pistol cheat functionality. With this cheat enabled, a shot from the pistol will deal maximum damage (255 or 0xFF) to any character or object in the game. This is the most difficult task in this assignment. You will have to trace through assembly (it is recommended that you use Cheat Engine’s memory view window to do this, but you may use any Page 5/6 tool of your choice). Hint: You will have to conduct several trial/error function call-to-nop conversions and then backtrack to find the exact location you need to change. \* Navigate to the folder containing your solution file (.sln) and then delete any .sdf files (>20MB) and then zip the folder that contains your .sln (including all subfolders) and copy it to your local drive and name it deliverable5.zip. Also include your tutorial-style write-up for Task 5 (including a few test cases that helped you verify the functionality worked) in a file named deliverable5.doc \*

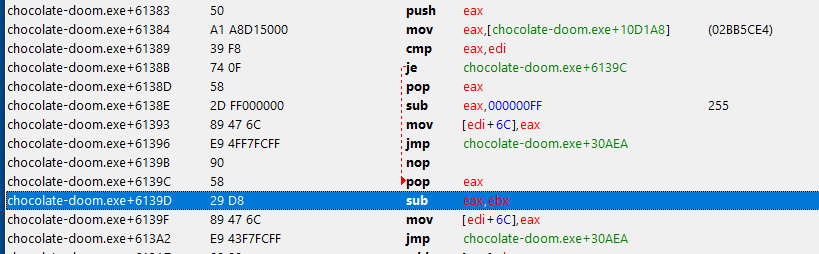
1. How we found address: we used address in task 4, we noticed here that the damage/health calculation were being done.
2. At first, we did not know how to add damage to gun, rather only where it needed to be done.
3. We used the following video (<https://youtu.be/H9_0exi5tCU> )
4. From this we were able to determine that the best route was to inject our own code, as we would only be responsible to write a little bit of assembly.
5. We also know that from the previous cheat we have some space that is not used.
6. We want to locate some area in the game that just has 00 00 bytes edi+6C is either your health or the damage you take since we know that, we want to make code that check if you’re getting hit and specifies the damage
7. When we create the jump we used auto assemble, then we create the following code, and it replaces
8. Before: Found where damage was being done and given. We found this out from the previous task. We decided to use this address because in the previous task we know this is where health and damage was being calculated. We then used auto assemble (AOB Injection Attack given by cheat engine). (LINK VIDEO HERE). The jump is where is goes to do what we injected.
9. In the picture below we determined that the highlighted portion was where the jump was taking place. We could inject the code here without breaking the game.



1. We use the auto assemble feature (also provided in the video) to generate the following jump. The jmp will point us to the address where we will inject our own code.

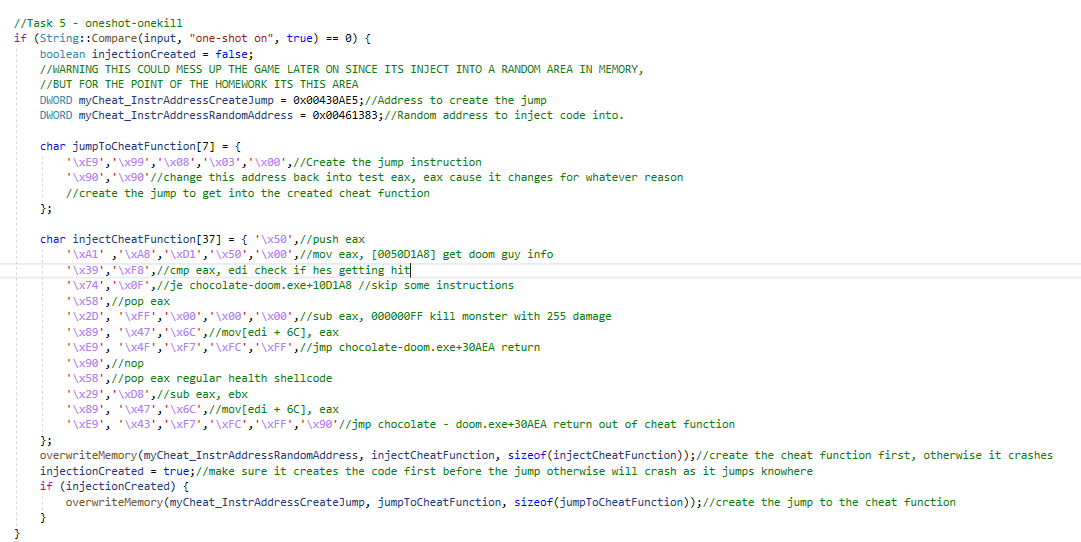


1. We create our own assembly and Cheat engine generates the code that we instructed it to create, as shown below.



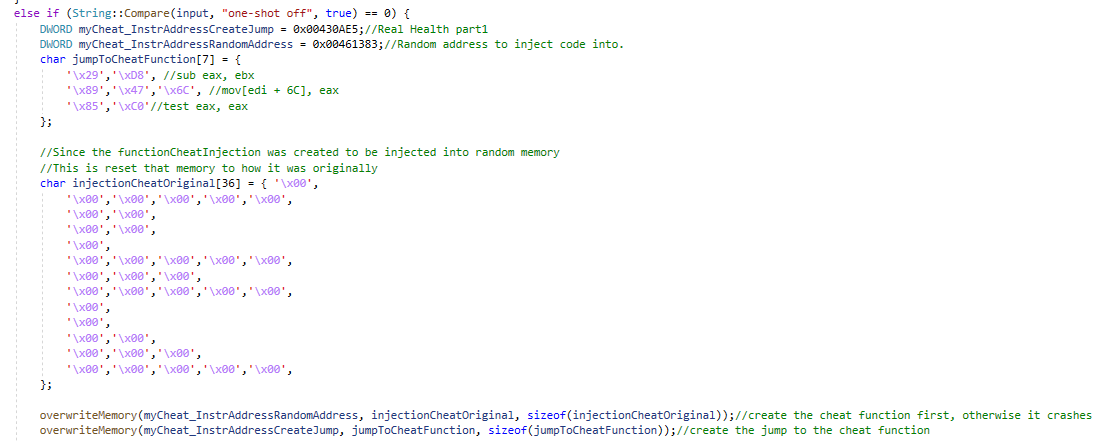
1. We now create the code to inject into the game.
2. We get the two addresses, one where the jump is created and the other a random address in memory pointing to 00 00 bytes.

Note: The following code provides a step by step description of what is happening.





1. Reset everything back to normal. So, where the jump is created is then returned back to normal. (Code below)



1. In injectionCheatOrginal this was what was already there.