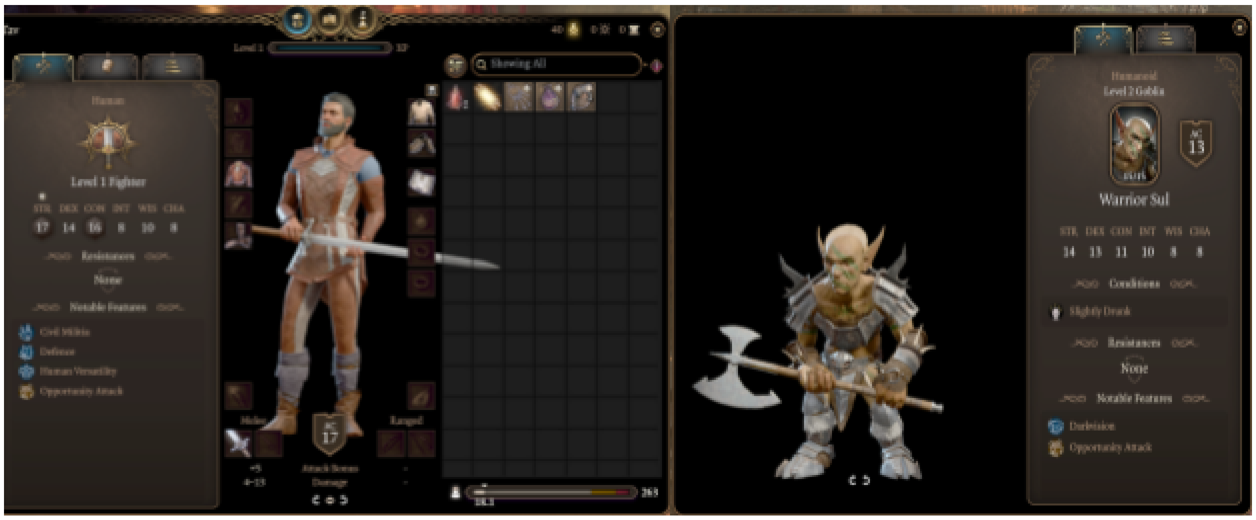
============================[Topic]================================

The topic I have chosen for my individual project is combat design. Specifically combat design in Baldur's Gate 3. Baldur's Gate 3, created by Larian Studios, is a role-playing video game based on the tabletop role-playing system of Dungeons & Dragons 5e. I chose this topic because I had been playing the game a ton and when looking for a topic, MJ suggested I do the combat in BG3. Working on this project has taught me the complexity of turn based combat when translating it into code. This is due to the various conditions that need to be met. I started playing BG3 about 3 months into its release. Currently I have 237.2 hours playing BG3 according to my Steam library.

============================[Logic]===============================

Unlike its previous iterations, BG3 is turn based since it is based upon D&D 5th Edition. In this link or on my webpage matthewsareaofwork.com I have placed the program that I wrote. It is based on how BG3's combat works in a simpler scenario. In the game, the player usually controls up to four characters while fighting various enemies. For the simplicity of the program, I have decided to make it a 1v1, a human fighter vs a goblin fighter without movement.

Here are the stat blocks from the game that I used to base the stats and actions of my program:



STR	DEX	CON	INT	WIS	CHA
17	14	16	8	10	8

STR	DEX	CON	INT	WIS	CHA
14	13	11	10	8	8

Note: Please pull up the code that I linked in order to see what I will be referring to. In addition, at the start of a new section of the code, I have put a comment titling the section including the lines of code they occupy. I have also included comments on the program for a more in depth explanation

—

Lines: 1-42

D&D is a dice based game, in order to create dice rolls I have used the "import random" function in order to use random numbers. Following that I have placed the character's (Tav's) stat block and used a formula that D&D 5th edition uses to calculate bonuses granted by the character's stats and labeled them (stat)Bonus. The other variables are meant to give and take away certain resources/conditions the character has.

—

Lines: 44-87

This is exactly the same as Tav's section except the variables have been renamed to fit the goblin.

—

Lines: 90-302

These lines contain several functions that, when called, perform the code contained. These functions are the attack actions for Tav. These functions will roll to check if it hits the goblin, deals damage,

—

Lines: 304-327

These lines contain two functions that, when called, perform the code contained. These functions perform potion drinking. The potions given at lvl 1 are 2d4+2, the function heals the drinker and is programmed to not go over the maximum hp a character has.

—

Lines: 329-460

These functions act the same way as Tav's attack functions, except the goblin doesn't have access to pommel. All of the variables are different but act the same way as before.

—

Lines: 463-483

These lines of code display messages showing that initiative is being rolled. Initiative was actually already rolled in the stat blocks at the beginning. I used an if else statement in order to determine who goes first based on who had the higher initiative. The appropriate variable is then set to 1 and that activates that character's turn. The combat actually starts on the last line of this section with the while loop, as long as one of the HPs are above 0 the code keeps looping.

—

Lines: 485-565

These lines of code are what help Tav's turn function. When the TavTurn variable is set to 1 and goblin is at 0, Tav's turn starts. The first part will display a message that shows the turn has started. The second part is to check if Tav possesses the bleeding condition, if so it will deal the appropriate damage to Tav. The third part is what determines what actions Tav takes. The code will ask the user to input a number between 1-4 (Attack, Pommel, Drink, END). Once a number is selected, you will select another number in order to make sure you want to perform the action. Once a number is inputted, the appropriate function is called. If you performed an action/bonus action already in the turn and try to do it again, it will deny you permission to perform the action/bonus action. If you used an action/bonus action that has a limited amount of uses (Pommel, Lacerate, Potions), the moment you use it, the number of uses will be reduced. It will also not let you use it anymore if there are no more uses.

—

Lines: 567-573

These lines of code encompass the turn switch once Tav's turn is over. It will restore Tav's action/bonus action and set TavTurn to 0 and Goblin to 1, fulfilling the condition for the Goblin's turn to start.

—

Lines: 577-612

These lines of code encompass the workings of the goblin's turn. Like Tav's, it will first determine if the goblin is bleeding and/or dazed. The dazed condition is given when Tav hit's the goblin and it fails its constitution saving throw. If the goblin fails, the dex bonus it has to its armor class is removed. This lasts for two rounds. Once the conditions are checked, The goblin will determine if it has an action, if so, it will determine if it has a use of its lacerate ability (1 time use). If so, it will automatically use that as its attack. If not, it will use its standard battleaxe attack. Then it will determine if it has a bonus action and it has less than 9 HP. If so, the goblin will drink a potion. Once actions are done, its turn ends.

—

Lines: 614-621

These lines of code act the same as Tav's turn switch once their turn ends. Restores action/bonus action and then switches goblin to 0 and TavTurn to 1.

—

Lines: 625-629

Once either Tav or the goblin is defeated (HP = 0), a message will be displayed depending on which character died.

=========================[Wrap Up]=============================

The things I learned during this project were somewhat expected and some not.. I knew that it was going to be somewhat long since a turn based game usually has multiple conditions that need to be met and performed. I didn’t know however, how frustrating it would be to name each variable to make them somewhat different from each other. While working on this project, I have learned that coding is incredibly rewarding once you manage to figure out the systems and make them work. I originally thought the difficulty I would have would be figuring out how to create the functions that I used, but really it was naming the variables.