Because I am fairly interested in RPG-esque games, I decided to create a turn-based fighting game for my project. The game revolves around four heros: Rob, the valiant knight who's experience with fighting monsters much outweighs his age; Jeremy, theand stumbling amateur wizard who is surprisingly good with ice spells; Fisher, the heavily armored tank who is far from afraid of anything in his way; and Ava, the majestic huntress who snipes her prey from above. These four heroes find themselves together after a glitch in the Kingdom CodeBlocksia teleported them to the Great Fault of Segmentation. The King is in trouble and the only way to save him is by fighting through the hordes of ASCII characters.

The game starts off in a menu with 4 options: New Game, Help, Joke, and Quit. New Game begins a new adventure, Help explains the controls, Joke prepares the player with a knock-knock joke, and Quit will exit the program. After any of the choices are selected by entering a corresponding number, the function for each will run its code and return to the main menu by a while loop until the user selects the Quit option. New Game opens the character selection menu where the player can choose his or her favorite hero to play as. From this point, users make selections by using the ‘s’ and ‘w’ keys to move a cursor ‘>’ between options and then double tapping ‘enter’ to confirm a choice. The user will then proceed to the battle in which he or she chooses a unique attack based on the character they choose to fight the enemy. Each character has three attacks that do varying points of damage. Both the health statistic of the enemy and the player are displayed, as well as a virtual battlefield with the player's avatar shown and the ASCII it is fighting.. After the player attacks, the enemy uses its base attack to fight the player. When the enemy object is instantiated, it has a random chance to use the constructor that boosts the attack and health statistics for the enemy. The game ends when either the enemy or players health is depleted past zero. The game displays the high score for the player, which is determined by subtracting the damage done to the player from the damage done to the enemy. The high score is then written to an external file to log the most recent score from every playthrough of the game. The user is then returned to the main menu where he or she can choose to quit or to play again.

Unfortunately, I had problems with core dumping and segmentation faults when attempting to use pointers to designate the player's character at runtime. Although the menu functionality of choosing a character still works, you can only really play as Rob the Knight. I would have also implemented the option to use either an item or attack on the player's turn, have statistics like defense and strength effect total damage output, usage of stats boosting items, randomly generated ASCII character for fighting against, and a boss fight. I am satisfied with the idea I invented, and I will most likely further pursue the development of this game in a program such as Unity.