Game Project

Taylor Smyth

H00470749

Ts4002@hw.ac.uk

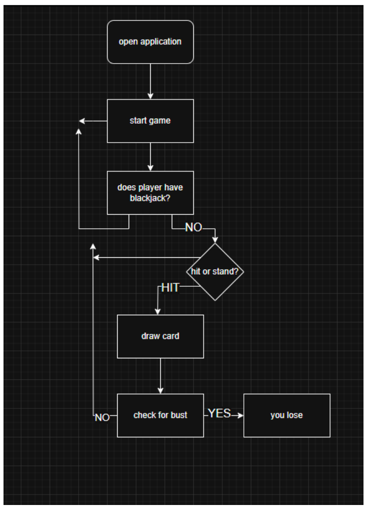
Introduction:

For this game report, I made Blackjack in C. My purpose for choosing Blackjack was that I wanted to learn and understand structs and pointers more, as they have been a hard topic for me to grasp. I can confidently say I have now learned how to use them effectively with a lot of revision and research. This game of Blackjack has a dealer you can play against, and was the simplest "AI" opponent I could think of. I also considered adding a betting feature, but I didn't feel comfortable including that.

Rules of the game:

The dealer deals cards in alternating fashion, dealing their first card face down but revealing it after the player has decided to end their turn. This is all implemented in my game.

The player can choose to hit or stand. Hit draws a random card from the deck; stand ends the player's turn. The dealer draws until they have at least 17.



Code overview:

The use of struct was for easier usage of data and better readability. This was really only useful in the Hand and Card structures.

initialize Deck creates arrays of cards for the deck to be composed of. This allows for real-world odds—it's exactly like an actual game of Blackjack.

calculateHandValue is used to calculate the total hand value and deal with aces when they make the total go over 21.

displayHand is used with a Boolean to hide the first card or not, depending on whose turn it is in the game.

clearScreen is used so that the terminal stays free of clutter, allowing for better gameplay.

Debugging

I had a big issue at the end with compiling the game—I had forgotten to initialise the calculateHandValue variable. This caused me a lot of time and research to fix, but it was resolved by simply initialising it at the start of the code.

One problem that has persisted on Mac devices is that the screen clear doesn't function at all. I decided not to research this as I have no experience with any Mac coding.

Making the core game maths logic (main function) took a lot of just writing things out on paper and trial and error. This easily took up half of my time making the game.

I had many issues with pointers, which were all human error and my fault. I had a tough time learning to read pointers properly.

Conclusion:

The game works as intended and is very fast at processing—even the screen clear is rapid. I believe this is because C is a very lightweight programming language. A betting system is something I considered but decided not to include, as it seemed immoral to make a gambling game for a project.

My conclusion is that the game works on Windows computers due to the libraries I use, and this can be easily resolved if needed.