Proposal

Description:

The program I will be creating is the game “Black Jack”. To begin, in “Black Jack” the player and the dealer are dealt two cards from a deck of cards. Each card has a numerical value, for all suits, cards with numbers 2-10 have a numerical value from 2-10 that corresponds to that card. For example, a one of hearts has a value of 1. In my version, an ace (of any suit), will have a value of 11. However, if time permits the player will have the option to use an ace as a 1 or 11. Moreover, face cards (jack, queen, and king of any suit) will have a value of 10. Therefore, the sum of the first two cards a player has is their total (in my program the two cards and total will be printed). As mentioned earlier, the dealer will also get two cards, however, only one will be revealed to the player (this will be printed). The player can then choose to hit (get another card) or stand (keep the cards they have) (this will be an option and the player can hit until they choose to stand or they bust). The sum of all the cards a player has cannot be greater than 21, if it is, they bust and lose. The dealer must hit if the sum of their own first two cards is less than 17, and the dealer must stand if the sum of their own first two cards is greater than 17. Therefore, if the player does not bust and they have a greater total with their cards then the dealer, the player wins, and vice versa. However, if the player busts (even if the dealer busts), the player loses. Furthermore, if the dealer busts and the player does not, the player wins. Finally, if the player’s total is equal to the dealer’s total (without either of them busting), then there is a draw. In my program the player will be the user and the dealer will be the computer. The player will be able to play as many times as they wish and there will be a menu option for the rules, starting a new game, playing again, and exiting the program. The program shuffles the deck of cards before each game, and then the program follows the “Black Jack” game that I described above. Furthermore, the program will also keep track of the wins, ties, and losses, if time permits, this will be replaced with the user’s money that they can bet (which will be given at the start of the game).

Details:

My program will try to imitate the experience of playing “Black Jack” as much as is feasible using the programming I have learned in this course/some new things. First and foremost, there will be a menu in my program where the user will be able to choose an option to read the rules, start a new game, play again, or exit the program. This menu will be displayed with a dialog pop-up box and the user’s input will be used to perform each task in the menu by parsing the string and using decision constructs (“if, else if” statements). Furthermore, the deck of cards will need to be shuffled before dealing the cards in each game, therefore, I will need a method for this. This method will contain a “for” loop, an enclosed “for” loop, and “if else” statements that will randomize the order of the cards and ensure that no card is repeated twice (I will also have to use a Boolean operator). Moving forth, I need to declare global arrays for the card names and card values so they can be used alongside the shuffled deck to assign card names and values to the player’s cards and the dealer’s cards. A method will also need to be created for the user’s choice of hitting or standing. The return value from this method will be used in another method that will use decision structures (“if, else if” statements) and repetition structures (“for” loop) to perform a series of actions based on whether the player decides to hit or stand. Moreover, a method that decides whether the dealer must hit or stand with the according consequences will be created with a similar format. Furthermore, another method will be created that will use comparison structures to compare the player’s total (sum of player’s cards) to the dealer’s total (sum of dealer’s cards). Comparison structures will also need to be used to see if the player or dealer busts. Finally, the program will then use this information to track the wins, losses, and ties. If the player decides to start a new game, this process needs to be repeated (done with a repetition structure, most likely a “do, while”) , however, the wins, losses, and ties need to be reset. However, if the player decides to play again, this process needs to be repeated (done with a repetition structure, most likely a “do, while”), however, the wins, losses, and ties need to keep accumulating.