1. Team name, members.

Team Name: Gamble Inc
Members: Nathan Berry, Josue Vides, and Emiliano Melendrez

2. Project Information and details: (30 points)

- What problems are you solving in this project?

The problem that we are trying to solve in this program is to show the player advantage/probability of not going over 21 in the game blackjack, and the advantage/probability of the player winning/losing against the dealer's hand.

- What solutions are you implementing in the project?

We are implementing a function, in which it keeps count of the total amount of cards left in the deck, and using the information from this, we use another function in order to call the players hand, and calculate the probability based on the remaining cards in the deck. We also use other functions to give players the probability of busting if they were to hit on their hand, and also we use functions that uses casino rules, where a dealer can't hit on a hand greater than 17.

- Provide explanation of calculations and algorithm implementation.

The calculations that we do in the program is finding the percentage that the dealer or player has of hitting (getting another card) without going over. We do this by keeping track of the cards remaining in the deck. This is done by assigning 13 variables with each one pertaining to a card A through K and setting each variable to 4. We draw a card by choosing a random number between 1 and 13 by using rand() and modulo each time you draw a card we remove 1 from the totalCards variable. Each time we subtract 1 from the variable coinciding with that number picked. We add up the values in the players/dealers hand and subtract 21 by the hands worth resulting in a value **a** we than add up all of the variables with values that are less than or equal to the value **a** we then divide that number by total cards. We multiply the resulting number by 100 to get the percentage.

- What is the program objectives? Explain how your program is interacting with the user and its purpose.

The program gives the user the total value of the cards they received and shows the dealer's hand as well, and then it gives the chance of them not busting, and then asks if they would like to hit or stand, allowing the user the choice of continuing the game or standing.

- How is discrete structures implemented in the C++ program?

We use the srand function given by the ctime library in order to get randomized numbers which we can then use to calculate probability of players hand and dealers hand chance of winning or losing. We also use the probability formulas learned/taught during the lecture of Unit 6, where we use the sample space (remaining cards), and also use the players hand and divide those in order to get probability of busting on the next hit. Furthermore, we used logic gates, in a while loop where it prompts the user to pick between 1 and 11, we use “and” logic gate to check user input and make sure its within the bounds of the card value, otherwise, they are stuck within the loop until they’ve chosen the correct value.

What are the limitations of the program?

Some limitations include not being able to split the doubles like in standard Casino Black Jack. Implementing suggestions for the player each time also seems to be tricky when trying to add with the rest of the logic. Another limitation that our program has is that the dealer can’t choose their own ace value (1 and 11), furthermore, the value of the ace isn’t fluid, meaning that once the player makes its choice on the value of the ace, it stays that value, as opposed to real life blackjack, where the ace can change from 1 to 11 based on player convenience.

One of the biggest limitations in our program

One of the biggest limitations is not being able to implement the rule where a player is allowed to split a double and hit on each card individually.

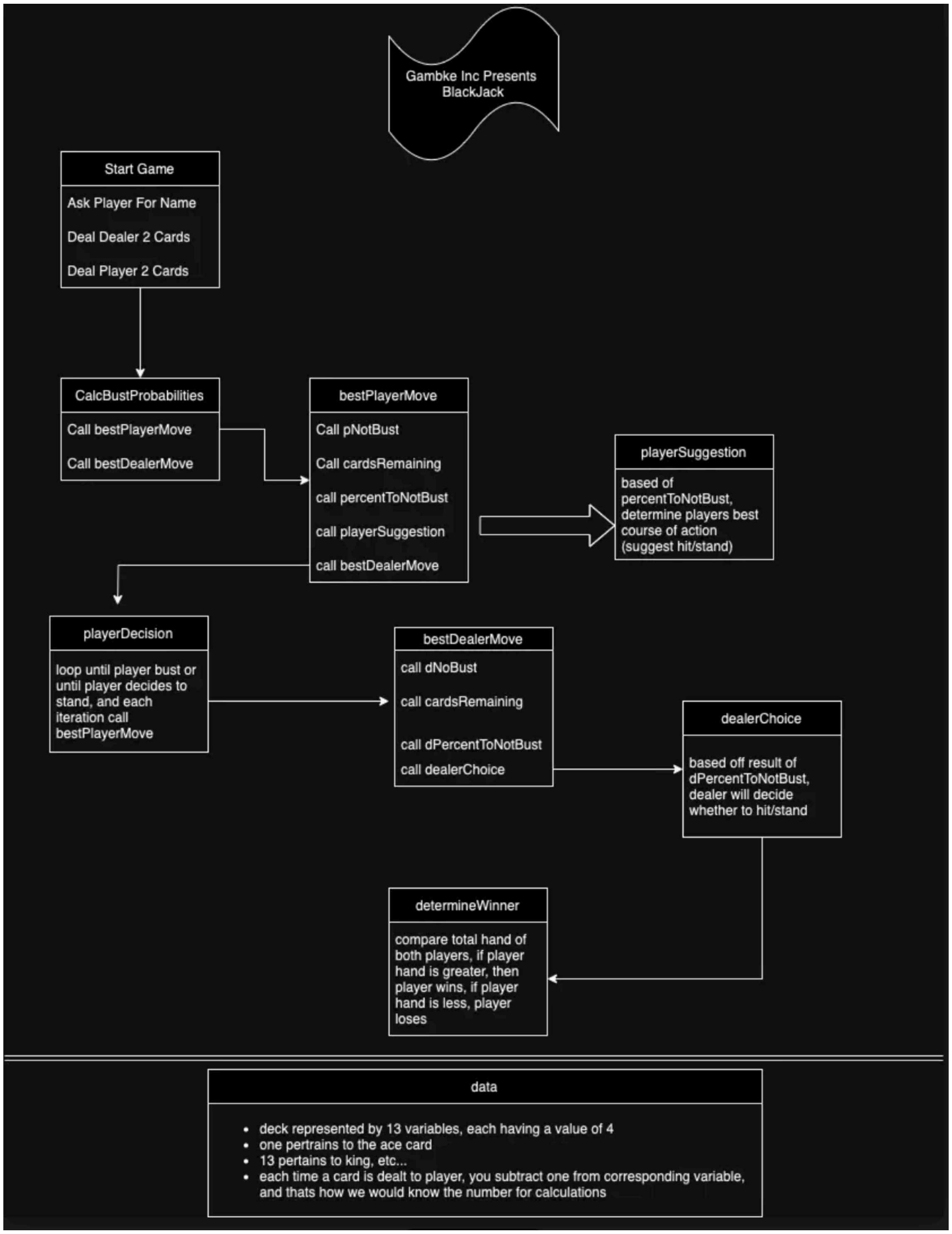
· Provide recommendation on improving the limitations of the program.

When it comes to the suggestions for the player, we could somehow implement an if statement checking and comparing the status for every time it updates the stakes of the user’s and computer’s chance of busting. Furthermore, although our code for the ace works as needed, it’s clunky and can be improved upon. Another thing to improve on, besides the things listed above, it would be a good idea to make a menu in the game, and allow users to play another game, without needing to restart the program.

3. Flowchart OR Pseudocode. (30 points)

· Write the pseudocode for the program, from start to finish. Be sure to include decision-making branching.

· If you choose to do flowchart, use standard shapes for flowchart, be sure to include decision-making branching. You can use web-based tool such as Draw.io to build your flowchart.


Gambke Inc Presents BlackJack
Start Game
Ask Player For Name
Deal Dealer 2 Cards
Deal Player 2 Cards
CalcBustProbabilities
Call bestPlayerMove
Call bestDealerMove
bestPlayerMove
Call pNotBust
Call cardsRemaining
call percentToNotBust
call playerSuggestion
call bestDealerMove
playerSuggestion
based of percentToNotBust, determine players best course of action (suggest hit/stand)
playerDecision
loop until player bust or until player decides to stand, and each iteration call bestPlayerMove
bestDealerMove
call dNoBust
call cardsRemaining
call dPercentToNotBust
call dealerChoice
dealerChoice
based off result of dPercentToNotBust, dealer will decide whether to hit/stand
determineWinner
compare total hand of both players, if player hand is greater, then player wins, if player hand is less, player loses
data
• deck represented by 13 variables, each having a value of 4
• one pertrains to the ace card
• 13 pertains to king, etc...
• each time a card is dealt to player, you subtract one from corresponding variable, and thats how we would know the number for calculations

https://replit.com/@nberryus/CIS-7-Project