Team Name:
Bubble Casino

Members:
Kenneth De La Cruz
Jaqueline Gastelum

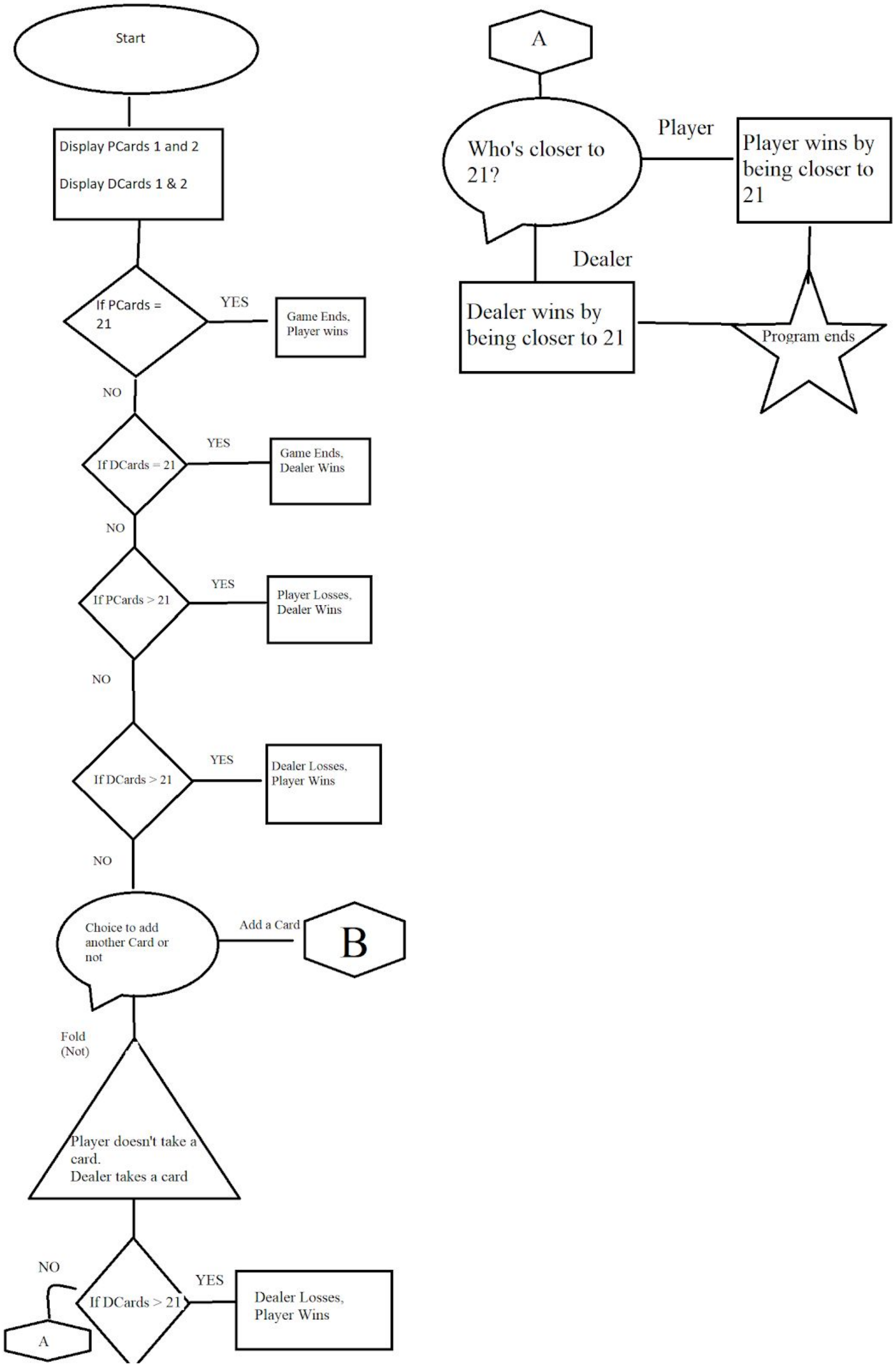
BlackJack Project Documentation

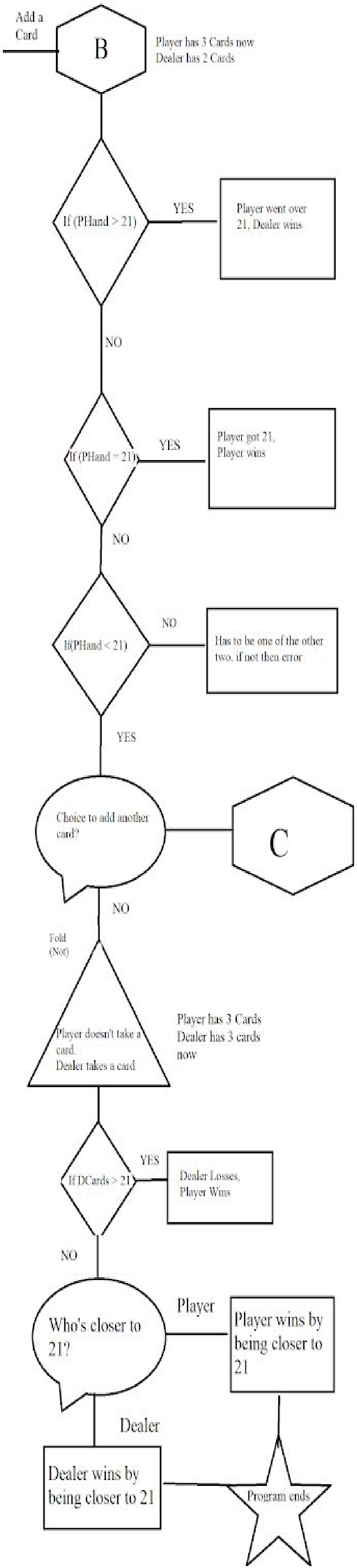
In this project we were solving problems such as how to create certain parts of a program, errors and a broken program. For instance, when creating the title and menu, it had some very simple errors which included: bad escape sequences (\ ) which led to closing quote errors and etc. And, as for the interface, it came to a point where it broke down because the int main () didn't work anymore. Also, to solve some of the problems that were given from the program, we would implement backup programs (programs that we have already worked on and saved onto another device) and we would pick apart pieces from a program when we had issues to decipher what the problem was. Then, if we had an error it was either fixed through looking it up on google or we would search in our CIS 5 textbook.

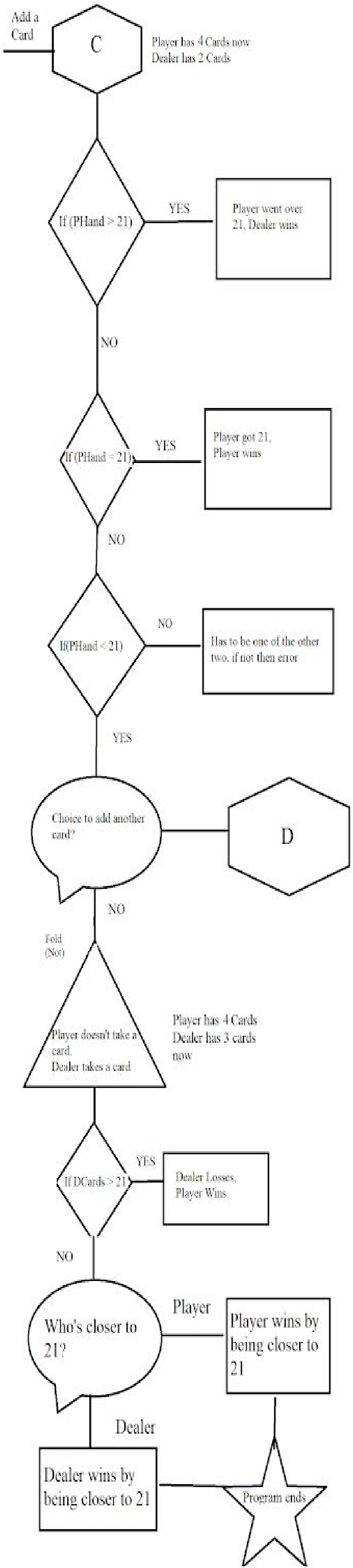
As for calculation implementation, certain calculations were applied when getting the random numbers to add to int for the cards, they use a random number. Since they are all ints they can only be whole numbers so there is a 1 added in case it becomes 0. The random number is then divided by 11 but only the remainder gets used. And, because the highest number in BlackJack is 11, we had to make it go from 12-1. The rest of the program is mostly just subtraction and addition. Now, in the algorithm implementation, it is applied into the project through sets of instructions/tasks that can vary from the program making calculations, asking questions that are stated in the input, sending messages and, in general, running the program. For example, it is used when applying rules as an option for the game, asking the player a question, verifying if the program is running well and setting a menu layout for the game interface.

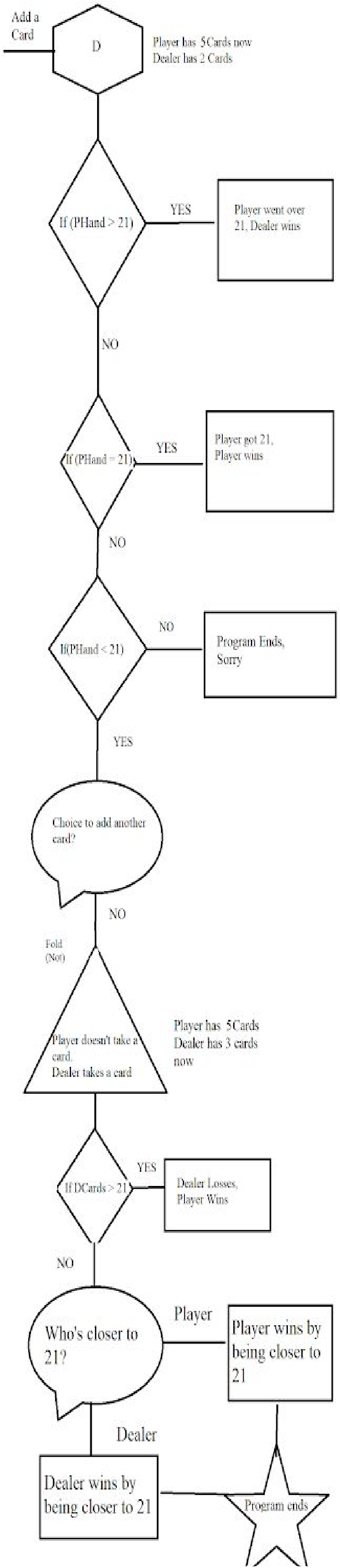
Aside from the algorithm and calculation implementations in the program, the objective of this is to run an application that is based around the game of BlackJack. It is meant to interact with the user by playing the game accompanied by a dealer and you as the player are supposed to try to either win by not exceeding an amount of 21 in cards or by getting blackjack (an Ace and 10). While that has described the purpose of the program, it is also important to say that Discrete Structures is being applied into the application by the use of probability and other mathematics, which is mostly utilized in the interface. Probability is shown as a result of randomization in the display of cards that are given when you start the game. Also, as stated before by the implementation of calculation, estimations were applied when getting the random numbers and they were also in use of sums and subtractions. All of these demonstrate the use of discrete structures for our project.

Lastly, the limitations of the program are: the very rare appearance of an error that occurs during a randomization of cards given. And, the menu choices are also a bit limited in the sense that you have to type the choice you want 2 times before the program gives it to you. However, if we were to improve more on the limitations of this program then we would most likely recommend to discover and fix the minor problem when the error is shown and research the issue so that it won't happen again. As for the menu choices, we would also recommend to take the time and figure out how to limit the amount of choices you should enter before you get the result from one of the options in the menu.




Add a Card
B
Player has 3 Cards now
Dealer has 2 Cards
If (PHand > 21)
YES
Player went over 21, Dealer wins
NO
If (PHand = 21)
YES
Player got 21,
Player wins
NO
If(PHand < 21)
NO
Has to be one of the other two. if not then error
YES
Choice to add another card?
C
NO
Fold (Not)
Player doesn't take a card.
Dealer takes a card
Player has 3 Cards
Dealer has 3 cards now
If DCards > 21
YES
Dealer Losses,
Player Wins
NO
Who's closer to 21?
Player
Player wins by being closer to 21
Dealer
Dealer wins by being closer to 21
Program ends


Add a Card
C
Player has 4 Cards now
Dealer has 2 Cards
If (PHand > 21)
YES
Player went over 21, Dealer wins
NO
If (PHand = 21)
YES
Player got 21,
Player wins
NO
If(PHand < 21)
NO
Has to be one of the other two. if not then error
YES
Choice to add another card?
D
NO
Fold (Not)
Player doesn't take a card.
Dealer takes a card
Player has 4 Cards
Dealer has 3 cards now
If DCards > 21
YES
Dealer Losses,
Player Wins
NO
Who's closer to 21?
Player
Player wins by being closer to 21
Dealer
Dealer wins by being closer to 21
Program ends


Add a Card
D
Player has 5 Cards now
Dealer has 2 Cards
If (PHand > 21)
YES
Player went over 21, Dealer wins
NO
If (PHand = 21)
YES
Player got 21,
Player wins
NO
If(PHand < 21)
NO
Program Ends,
Sorry
YES
Choice to add another card?
NO
Fold (Not)
Player doesn't take a card.
Dealer takes a card
Player has 5Cards
Dealer has 3 cards now
If DCards > 21
YES
Dealer Losses,
Player Wins
NO
Who's closer to 21?
Player
Player wins by being closer to 21
Dealer
Dealer wins by being closer to 21
Program ends

Pseudo Code for the Project : Bubble Casino

```
Player Card 1 = random # between 1-11
Player Card 2 = random # between 1-11
Player Card 3 = random # between 1-11
Player Card 4 = random # between 1-11
Player Card 5 = random # between 1-11
Dealer Card 1 = random # between 1-11
Dealer Card 2 = random # between 1-11
Dealer Card 3 = random # between 1-11
Player Hand= Player Card 1 + Player Card 2
Dealer Hand = Dealer Card 1 + Dealer Card 2
Player results = Player Cards – 21
Dealer Results = Dealer Cards – 21
Prints Player Card 1 + Player Card 2
Prints Dealer Card 1 + Dealer Card 2
If (Player Card 1 + Player Card 2 < 21)
{
Then your given a choice between another card or folding (Folding means no more cards)
If yes then given another card
{
Player Card 1 + Player Card 2 + Player Card 3 = Player Hand
If (Player Hand > 21)
Then you lose.
Else if (Player Hand < 21)
```

```
Then your given another choice to pick a card.
{
If (Pick a card)
{
Then Player Cards 1+ 2 + 3 + 4 = Player Hand
Then given another choice of cards
If (Player Hand < 21)
{
Given another choice to pick cards
If (Pick a card)
{
Then Player Cards 1 + 2 + 3 + 4 + 5 = Player Hand
I stopped at 5 cards so
If (Player Hand < 21)
{
The game ends.
}
Else if (Player Hand > 21)
{
Player loses
}
Else if (Player Hand = 21)
{
Player wins
}
}
Else (Player refuses to draw a card)
{
Player hand = Player Cards 1 + 2 + 3 + 4
Dealer draws a card
Dealer Hand = Dealer Cards 1 + 2 + 3
Then by some math to see who is closer to 21, wins
If (P results < D results)
{
Player wins by being closer to 21
}
Else (D results < P Results)
{
Dealer wins by closer to 21
}
}
Else (Player refuses to draw a card)
{
Player hand = Player Cards 1 + 2 + 3
Dealer draws a card
Dealer Hand = Dealer Cards 1 + 2 + 3
Then by some math to see who is closer to 21, wins
If (P results < D results)
```

```
{
Player wins by being closer to 21
}
Else (D results < P Results)
{
Dealer wins by closer to 21
}
}
Else (Player refuses to draw a card)
{
Player hand = Player Cards 1 + 2
Dealer draws a card
Dealer Hand = Dealer Cards 1 + 2 + 3
Then by some math to see who is closer to 21, wins
If (P results < D results)
{
Player wins by being closer to 21
}
Else (D results < P Results)
{
Dealer wins by closer to 21
}
}
Else if (Player Hand > 21
{
Player automatically loses
}
Else if (Player Hand = 21)
{
Player automatically wins
}
Else if (Dealer Hand = 21)
{
Dealer automatically wins
}
Else if (Dealer Hand > 21)
{
Dealer automatically loses
}
Program ends
```