**Project 1**

Title

**Craps Casino Game**

Course

**CIS-5**

Section

**41595**

Due Date

**February 6, 2023**

Author

**Justin Arellano**

**Table of Contents**

1. Introduction………………………………………………………….pg 3
2. How to play Craps………………………………………………….pg 4
3. Money system……………………………………………………….pg 5
4. Development Story………………………………………………..pg 6
5. Pseudocode…………………………………………………………..pg 8
6. Flowchart………………………………………………………………pg 13
7. Checklist Lines……………………………………………………….pg 26
8. Actual Code……………………………………………………………pg 27
9. Proof of code running…………………………………………….pg 33
10. **Introduction**

For project 1, We were given the option to write a well known game. It could be either a card game, dice game, or a board game. For my project 1, I decided to do the dice game, Craps. Craps is a casino game in which participants bet on the outcome of the roll of a pair of dice. I went with the Pass Line and Don’t Pass line rule sets for the game.

My code has a money/betting system so that you can bet money on Craps. Before the game begins my code lets you choose between checking your money or playing the game.

1. **How to play Craps**

In order to play Craps, a person known as the “shooter,” throws out a pair of dice at the table. However, all bets must be made before the dice are thrown. The bets can follow one of two rule sets. The Pass Line, in which you are betting on the outcome of the dice, or, Don’t Pass Line, in which you are betting against the outcome of the dice.

**Pass Line Rule Set**

The Pass Line rule set is to bet on either rolling a 7 or an 11. If a 7 or an 11 is rolled, the person(s) who made the Pass Line bet automatically win the bet. However, if the sum adds up to be a 2, 3, or a 12, then the player(s) automatically lose the bet. The most likely outcome is to roll any other number. [4,5,6,8,9,10] In this case, the player gets a “point” (the number they rolled is the point) and will then have to roll the dice again. The dice must be rolled until the player(s) either roll the same number they rolled for the “point” in order to win the bet or until they roll a ‘7’ in which they lose the bet.

**Don’t Pass Line Rule Set**

The Don’t Pass Line rule set is essentially the opposite of a Pass Line rule set. The Don’t Pass Line rule set is to bet on either rolling a 2, 3, or an 12. If a 2, 3 or a 12 is rolled, the person(s) who made the Don’t Pass Line bet automatically win the bet. However, if the sum adds up to be a 7, or an 11, then the player(s) automatically lose the bet. The most likely outcome is to roll any other number. [4,5,6,8,9,10] In this case, the player gets a “point” (the number they rolled is the point) and will then have to roll the dice again. The dice must be rolled until the player(s) either roll the same number they rolled for the “point” in which they lose the bet or until they roll a ‘7’ in which they win the bet (Opposite of Pass Line’s point rules).

1. **Money System**

My game features a money system so that you can bet money and the game keeps track of how much you have based on if you lost or won. You automatically start of with 100 dollars.

If you choose to look at the money system before playing Craps the code will tell you the amount of money you have in terms of 10 dollar bills. (Ex. If you have 50 dollars left from betting, the system will tell you that you have 5 10-dollar bills.) If by chance you decided to bet money that wasn’t divisible by 10 the remainder would be given to you in one-dollar bills. (Ex. If you have 57 dollars left from betting, the system will tell you that you have 5 10-dollar bills and 7 one-dollar bills.)

If you try and bet an amount below a dollar or above what you have, the system will let you know that your bet was invalid and to type in a new bet.

If you win a bet, there is a 1/20 chance of having your bet squared as a token of appreciation for playing my game.

1. **Development Story**

My Project came to be about 8 versions long.

|  |  |
| --- | --- |
| Lines of codes | 205 |
| Commented lines | 57 |
| Total lines | 205 (I commented on the same line as the code) |

**1st version**

The first version of my code was pretty light. It just included me figuring out how to make a dice system. Getting it to work and outputting it so that I can see that it works properly.

**2nd version**

For the second version, I implemented an if else statement for the Pass line and Don’t Pass line rule sets of Craps. I also added the if-elseif-else statements within both rule sets to determine which conditions are met to gain money or lose money. I also added a command which allows the user to use both lowercase and uppercase characters when choosing between rule sets.

**3rd version**

For the third version, I put the Craps dice game within a do-while loop. This ensures that the user plays a game first and also allows them to play the game again if they want to. I added an option that allows them to stop playing entirely, stopping the loop or if they type “yes” they can continue to keep playing.

**4th version**

The fourth version had me get rid of an integer. This is the version where I started working on the “point” section of the rule sets in which a number other than a winning/losing number is chosen. Within this else statement I needed to implement a second dice roll number because rules indicate that the dice should keep being rolled until a sum of 7 or a sum equaling the previous sum of dice is rolled. In order to do this, I had a while loop which kept looping the second dice roll until it got what was needed. Once the needed number is rolled a new integer would increase by 1 therefor stopping the while loop since the while loop would infinitely continue which the new integer is equal to 0.

**5th version**

My fifth version was me completing both rule sets to the most basic level. At this point, both rule sets “technically” worked. I was originally having problems with the second dice rolls because I didn’t think about how I needed to make completely separate variables. I was done with every part regarding the base rules of Craps.

**6th version**

I was struggling with getting more lines and some of the points from the check off sheet. I was able to solve this problem with adding a money system so that when craps is played, the player can actually bet money. They were given a base 100 dollars at the start of the do-while loop. I also implemented a switch case menu so that the user can choose between playing the game or checking their money. After playing a game, the do-while loop would loop them back to choosing a case from the switch menu. At this time you can now check your money to see the new balance that you have and to decide how to bet more carefully.

**7th version**

At the end of this version I was finished with my code. I fixed invalid options. A few times through the game the user is prompted with some options. Invalid options were simply ending the program as a whole. I made sure that if they choose invalid options, the program would tell the user that they can’t choose that and it reenters them into the loop allowing them to try again instead of ending the program abruptly. I also implemented a random lottery type system whenever a user wins in order to implement cmath into my code. The winner has a 1 in 20 chance of getting their bet money squared.

**8th version**

In this final version, I made sure to go back and comment to make my code easy to understand. I commented on lines that needed an explanation into what they’re doing.

1. **Pseudocode**

*Calculate random time seed*

*Initialize variables*

*Output welcomes user to game*

*do*

*first dice roll logic*

*second dice roll logic*

*sum of both dice logic*

*output: ask how much they want to bet*

*input: bet amount*

*while bet amount is less than 1 dollar or more than the amount they have*

*if bet is less than 1 dollar*

*output: you need more money*

*else*

*output: you can’t bet more money than you have*

*input: new bet amount*

*output: Ask if they want to check money system or play game*

*output: 1 for money*

*output: 2 for Craps*

*input: choice between both options*

*switch case menu for both choices*

*case 1 for money*

*for loop to divide user’s money into groups of 10*

*output: amount of money in 10 dollar bills*

*ternary operator to assign remaining 1 dollar bills into variable*

*if remaining money is greater than 0*

*output: user has this much remaining in 1 dollar bills*

*else*

*output: user has no remaining money*

*output: bet returned since game was not played*

*end of case 1*

*case 2 for Craps*

*output: how much they bet*

*output: ask what rule set they want*

*output: Y for Pass Line, N for Don’t Pass line*

*input: what rule set they want*

*command to make upper and lower case characters work*

*if Pass Line rule set*

*output: dice rolls*

*if dice = 7 or 11*

*output: they won*

*if 1 in 20 chance lands*

*bet squared is added to money*

*output: congrats*

*bet is added to balance*

*else if dice = 2, 3, or 12*

*output: they lost*

*bet subtracted from balance*

*else*

*point variable equals 0*

*while point variable is less than 1*

*2nd dice roll logic*

*2nd sum of dice roll logic*

*If dice = 7*

*Output: dice rolls and sum*

*Output: they lost*

*Bet subtracted from balance*

*Point now equals 1 while loop stops*

*If 2nd dice sum is equal to 1st dice sum*

*Output: dice rolls and sum*

*Output: they won*

*If 1 in 20 lands*

*if 1 in 20 chance lands*

*bet squared is added to money*

*output: congrats*

*bet is added to balance*

*Point now equals 1 while loop stops*

*If Don’t Pass Line rule set*

*output: dice rolls*

*if dice = 7 or 11*

*output: they lost*

*bet subtracted from balance*

*else if dice = 2, 3, or 12*

*output: they won*

*if 1 in 20 chance lands*

*bet squared is added to money*

*output: congrats*

*bet is added to balance*

*else*

*point variable equals 0*

*while point variable is less than 1*

*2nd dice roll logic*

*2nd sum of dice roll logic*

*If dice = 7*

*Output: dice rolls and sum*

*Output: they won*

*If 1 in 20 lands*

*if 1 in 20 chance lands*

*bet squared is added to money*

*output: congrats*

*bet is added to balance*

*Point now equals 1 while loop stops*

*If 2nd dice sum is equal to 1st dice sum*

*Output: dice rolls and sum*

*Output: they lost*

*Bet subtracted from balance*

*Point now equals 1 while loop stops*

*Else*

*Output: non valid option*

*end of case 2*

*default if something is chosen other than 1 or 2*

*output: Invalid option*

*output: shows them their current balance*

*output: ask if they want to continue*

*while the Boolean stop is not equal*

*input: yes or no to stop or continue*

*if string agn is yes*

*stop is true*

*else if string agn is no*

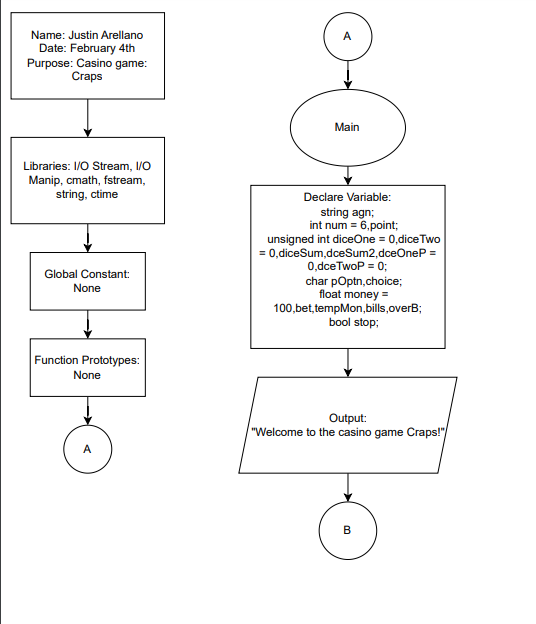
*stop is true*

*else*

*output: invalid option*

*while string agn is yes stop program but if no then program loops back the do-while*

1. **Flowchart**

**** **Diagram

Description automatically generated** Diagram

Description automatically generated Diagram

Description automatically generated Diagram

Description automatically generated Diagram

Description automatically generated Diagram

Description automatically generated Diagram

Description automatically generated Diagram

Description automatically generated Diagram

Description automatically generated Diagram

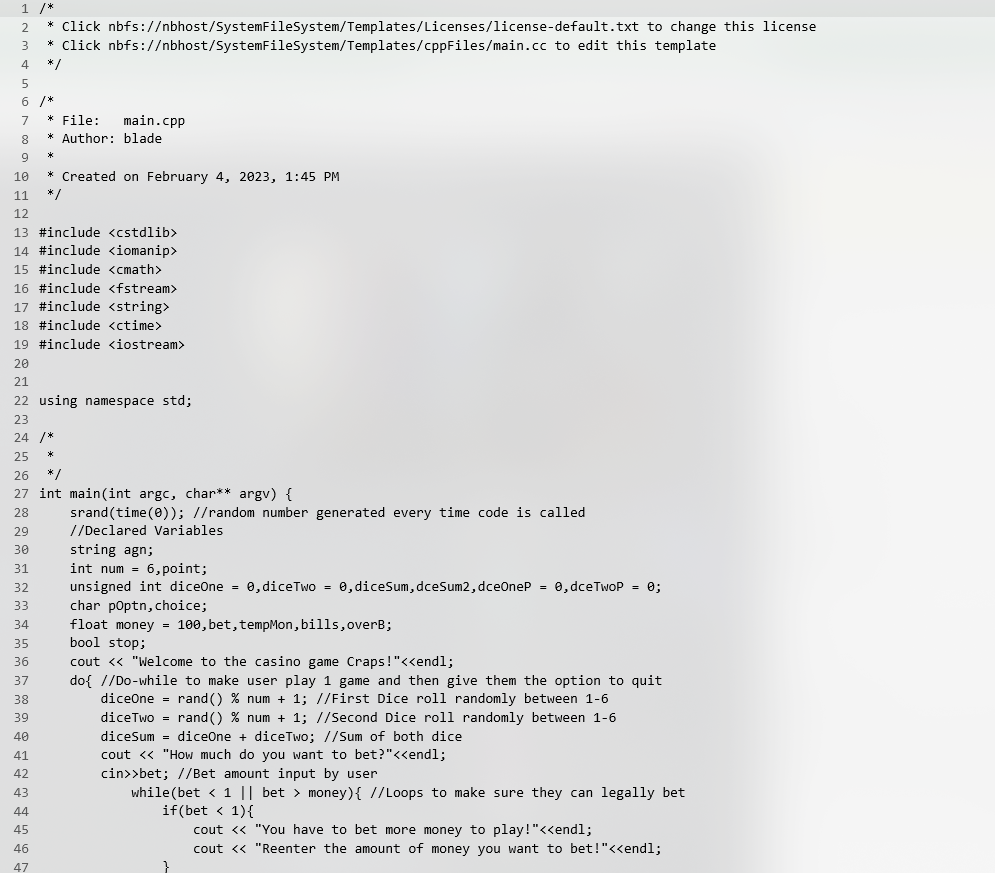
Description automatically generated Diagram

Description automatically generated

1. **Checklist Lines**

|  |  |  |
| --- | --- | --- |
| **Topic** | **Line location #** | **Points** |
| Libraries | 30-35 | 8 |
| Integers | 31 | 3 |
| Characters | 33 | 3 |
| Strings | 30 | 3 |
| Floats | 34 | 3 |
| Bools | 35 | 4 |
| Comments | 28-203 | 5 |
| Type Casting | 32 | 4 |
| Formatting Output | 75 | 4 |
| Strings | 190 | 3 |
| Math Library | 87 | 4 |
| Independent If | 87 | 4 |
| If-else | 44-48 | 4 |
| nesting | 37-203 | 4 |
| If-else-if | 80-97 | 4 |
| Logical operators | 85 | 4 |
| Validating user input | 181 | 4 |
| Conditional operator | 64 | 4 |
| Switch | 58 | 4 |
| Increment/decrement | 111 | 4 |
| While | 101 | 4 |
| Do-while | 37-203 | 4 |
| For loop | 60 | 4 |
| File input/output |  | 8 |

1. **Actual code**

**** **Text, letter

Description automatically generated**

Text, letter

Description automatically generated Text

Description automatically generated Graphical user interface, text, application

Description automatically generated

1. **Proof of Code running**

Text

Description automatically generatedText

Description automatically generatedText

Description automatically generatedText

Description automatically generatedText

Description automatically generated