**Project 2**

Title

**Craps Casino Game**

Course

**CIS-5**

Section

**41595**

Due Date

**February 12, 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 12
7. Checklist Lines……………………………………………………….pg 29
8. Actual Code……………………………………………………………pg 31
9. Proof of code running…………………………………………….pg 40
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.

For Project 2, we were supposed to add functions, arrays, and sorts to our old Project 1. I moved the money system and Craps game into separate functions. I also moved the switch case into a new function. I added 2 new cases to my switch case menu which allowed the player to check their wins and their losses and allowed them to shut down the program completely if they wanted.

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**

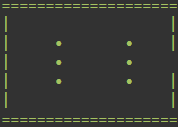
My Project came to be about 8 versions long.

|  |  |
| --- | --- |
| Lines of codes | 411 |
| Commented lines | 83 |
| Total lines | 411 |

**1st version**

The first version started with my first Project’s code. I didn’t add anything specifically in this version because I was trying to put the money system and craps into its own functions without messing the original code up. I also gave the functions a random variable because I wasn’t sure if they were going to be void functions or not. I also kept my switch case menu within the main.

**2nd version**

For the second version, I decided that I was going to take the switch case menu and make it as a function as well. I wanted to have as less code within the main. I did however, keep my do-while loop in the main so that the do while would call the menu function. I also decided to add a dice output aesthetic which would show what the dice would look like whenever the dice roll was said in the code. (Ex: If the program would say the dice rolled a 6,  this would be output.

**3rd version**

For the third version, I realized that most of my functions would end up being void functions because they are only outputting. I also realized that I left my variables in the main. The variables which were used in craps and my money system were left in main and so therefore no variable was working in my functions. I shifted each variable to is respective functions and got rid of them from the main. I also changed the name of the money system to be “balance” as it made the code more readable. I finally, added some parameters to the functions.

**4th version**

The fourth version had me playing with vectors and arrays. I wanted to use vectors and arrays to portray a system which told the user how many wins and losses they were accruing from playing Craps. I went into my Craps function and added a return ‘W’ or return ‘L’ to each section where they won and loss. This way not only would the money they bet be counted but also each win and loss. I made a bubble sort and selection sort here to count the wins and losses.

**5th version**

My fifth version I commented out the dice aesthetic function because I knew it was going to be easier to implement but didn’t think my code was ready for it. Version 5 didn’t have many changes as I was trying to just get the code to run.

**6th version**

For version 6, I was trying to find places where I could implement more from the check list. I also realized that my while loop stopped working. It would loop the code again so that players could play again but after the second play through it wouldn’t allow them to play through the game or check their balance anymore.

**7th version**

In this version I reintroduced the dice function. I originally wanted to have the dice aesthetic side by side because that’s how it would’ve looked in a real casino but I realized that in order to do that I would need and if-elseif-else statement for every single combination of die. Since each die took about 8-10 lines of code. Having all 36 combinations of code could possibly end with around 350 lines. So instead I made it call the function twice and saved about 200 lines of code. It looks less aesthetically pleasing but makes for less clutter. I then had the program cout the function every time a new die is rolled.

**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 and I tried to find any mistakes to fix. I introduced a new function called wrning. This function ended the program immediately using the exit() function. This was because the function was on the checklist but it also allows players to immediately end the code if they start the program but don’t want to play anything on it.

1. **Pseudocode**

*Libraries*

*Function prototypes*

*Main*

*Seed random time*

*Declare variables*

*Output “welcome to the casino”*

*Do- while loop*

*Output list of options to choose from menu*

*Calls menu function*

*Outputs option to continue*

*Yes continues no ends program*

*Menu function*

*Declared variables for menu function*

*Switch case menu*

*Case ‘1’ checks balance*

*Case ‘2’ plays Craps*

*Case ‘3’ checks wins and losses*

*Case ‘4’ exits program*

*Balance function*

*Declared variables for balance function*

*For loop to count bills*

*Checks balance and tells them how much money they have in terms of*

*10 dollar bills and 1 dollar bills*

*Craps function*

*Declared variables for craps function*

*Output ask them how much they want to bet*

*Input how much they bet*

*While loop to check if bet is legal*

*If not new input until legal*

*Dice rolls*

*Calls dice function*

*Choose Pass Line or Don’t Pass Line rule set*

*Pass line rule set*

*7 or 11 means they win*

*1 in 20 chance to earn bet squared*

*Returns ‘W’*

*2,3,12 means they lose*

*Returns ‘L’*

*Any other number means roll again until 7 or same number*

*Calls dice function*

*7 means lose*

*Return ‘L’*

*Same number means they win*

*1 in 20 chance to earn bet squared*

*Return ‘W’*

*Don’t Pass Line rule set*

*7 or 11 means they lose*

*Return ‘L’*

*2,3,12 means they win*

*1 in 20 chance to earn bet squared*

*Return ‘W’*

*Any other number means roll again until 7 or same number*

*Calls dice function*

*Same number means lose*

*Return ‘L’*

*7 means they win*

*1 in 20 chance to earn bet squared*

*Return ‘W’*

*If Pass line and Don’t pass line weren’t chosen*

*Output they choose an invalid option*

*Send them back to menu*

*winCnt function*

*Declared variables for winCnt function*

*Counts ‘W’ and ‘L’ returned from game*

*Reorders wins and losses to be in different groups*

*Outputs how many wins and losses they have*

*Sort function with arrays*

*Declared variables for sort function*

*Keeps track of how many wins and losses as an array*

*Bubble sorts them*

*Sort function with Vectors*

*Declared variables for sort function*

*Keeps track of how many wins and losses as vectors*

*Bubble sorts them*

*dcePrnt function*

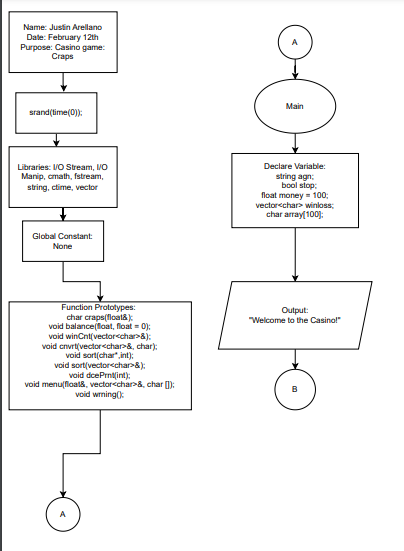
*if-else-if statements to display aesthetic die based on what was rolled randomly d Displayed when called upon*

*wrning Function*

*output they have been warned*

*exit() function ends program*

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 A picture containing text, person, athletic game, sport

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 | 13-21 | 5 |
| Integers | 133 | 1 |
| Characters | 44 | 1 |
| Strings | 40 | 1 |
| Floats | 42 | 1 |
| Bools | 41 | 1 |
| Comments | 39-406 | 2 |
| Type Casting | 134 | 1 |
| Formatting Output | 56 | 1 |
| Strings | 61 | 1 |
| Math Library | 167 | 1 |
| Independent If | 166 | 1 |
| If-else | 139-143 | 1 |
| nesting | 140-273 | 1 |
| If-else-if | 157-178 | 1 |
| Logical operators | 164 | 1 |
| Validating user input | 273-275 | 1 |
| Conditional operator | 119 | 1 |
| Switch | 84 | 1 |
| Increment/decrement | 194 | 1 |
| While | 138 | 1 |
| Do-while | 46-74 | 1 |
| For loop | 115-127 | 1 |
| File input/output |  | 2 |
| Function Prototypes | 24-32 | 4 |
| Pass by Value | 112 | 4 |
| return | 176 | 4 |
| Returning boolean |  | 4 |
| Static Variables |  | 4 |
| Defaulted argument | 25 | 4 |
| Pass by reference | 79 | 4 |
| overloading | 322 and 335 | 5 |
| Exit() function | 410 | 4 |
| Single dimensioned arrays | 322 | 3 |
| Parallel arrays |  | 2 |
| Single dimensioned as function arguments |  | 2 |
| 2-D arrays |  | 2 |
| STL Vectors | 279 | 2 |
| Passing Arrays to and from functions | 312-322 | 5 |
| Passing Vectors to and from functions | 279 and 97 | 5 |
| Bubble sort | 328-330 | 4 |
| Selection Sort | 287-293 | 4 |
| Linear or binary search | 298-10 | 4 |

1. **Actual code**

Text

Description automatically generated Text

Description automatically generated Text

Description automatically generated Text, letter

Description automatically generated Text

Description automatically generated Text

Description automatically generated Text, letter

Description automatically generatedText

Description automatically generated Table

Description automatically generated Table

Description automatically generated

1. **Proof of Code running**

Text

Description automatically generated Text

Description automatically generated

A computer screen capture

Description automatically generated with medium confidence

A computer screen capture

Description automatically generated with medium confidence

Text

Description automatically generated Text

Description automatically generated A screenshot of a computer

Description automatically generated with medium confidence

Text

Description automatically generated