CASINO WAR

6/14/2018

By: Prajwal Singh Kandhari

Contents

Casino War Proposal: 3
The Rule Procedures: 3
Commons Rules: 3
Graphical Planning: 3
Structural Planning: 3

FEASIBILTY REPORT: DESIGN AND DEVELOPMENT 4
Introduction: 4
Game Loop: 4
Appearing or Disappearing Buttons: 4
Winning: 4
Winning or Losing: 4
Repeating Loop: 4
Flow Chart: 4
IPO Chart: 6
Grant Table: 6
Testing Cases: 6
Buttons: 6
EasyGUI: 7
Functions and Procedure: 7

IMPLEMENTATION: 7
Plans Sacrificed: 7
Problem Log 7
Code: 8
User Guide: 8
Commons Rules: 8

TESTING: 8
Buttons: 8
EasyGUI: 8
Functions and Procedure: 9
Notes: 9

MAINTANCE: 9
Brags and Drags: 9

CASINO WAR

ANALYSIS:

Casino War Proposal:

Game Casino War is a popular, and the easiest game in a casino. To begin, this game is a two players game; the human players against the dealer. The main component needed for this game is a deck of cards.

The Rule Procedures:

1. A deck of card is shuffled.
2. The player places a bet and the dealer matches his bet with the player's.
3. The player picks the top card from the deck and the dealer follows the same step
4. The two cards picked from the deck are then compared.
5. The higher card in a ranking system wins the bet and the dealer must pay you, however if the dealer's card is better, you must pay him.
6. The card ranking follows from worst to best: 2, 3, 4, 5, 6, 7, 8, 9, 10, J, Q, K, A.
7. In a case of tie:
    a. the player has choice to either surrender or to go to war again.
    b. If the player surrenders, he will play receive half of previous bet back.
    c. If the player decides to go to war again, he will place a bet again, however this time it will be the double of bet previously placed.

Commons Rules:

1. In a game, the player has a limit of losing 1,000 chips
2. The cards are shuffled each time a winner is chosen
3. The cards are randomly chosen: there is no bias for either human player or the computer
4. The player can only spend the money he or she has
5. The player can cash-out or leave after the winner is decided.

Graphical Planning:

The game will be designed and the GUI will be made using pygame. There will be an introduction screen, with two buttons, one for user guide, and the other to start the game. Game Screen: a button will be placed to view the card ranks. A timer will be used for the player's turn of 30s to choose to bet again or to go to war in case of a tie. Some animations will be: going to war, surrender, bet, shuffle cards and pick the top card from the pile.

Structural Planning:

This game will use repetition structed, function, parameters, conditional statements, python based functions and many imported functions from time, random, and pygame.

FEASIBILTY REPORT: DESIGN AND DEVELOPMENT

Introduction:

The game will first begin by showing an introduction screen. Upon the click on "Next", the rules screen will be on the screen. After, the main gaming screen will show up on the screen.

Game Loop:

First, the cards must be shuffled. Second, you must confirm to enter the betting round; the confirmation will restrict the user to quit the game. A bet amount will be set between 0 to amount of coin one has. Two cards from the shuffled deck will be drawn.

Appearing or Disappearing Buttons:

The buttons will appear or reappear depending on its Boolean status. After each button's function, the button will be disabled and then the next button will be enabled. In each function, a selection will be required to recognize the button's variables. For example, after shuffling, the shuffle button's variable will be set to False. So, when the game loop, reaches the function the shuffle button will not appear.

Winning:

Tie: if the same number but from different suit is chosen then one has the choice to go to war or surrender. If they choose to surrender, then the game the shuffle button and leave button will appear back again and they will be awarded half the coins they had bet which will start the game loop over. If the user chooses to go to war then, their bet amount will be doubled, and he will be told to pick card again.

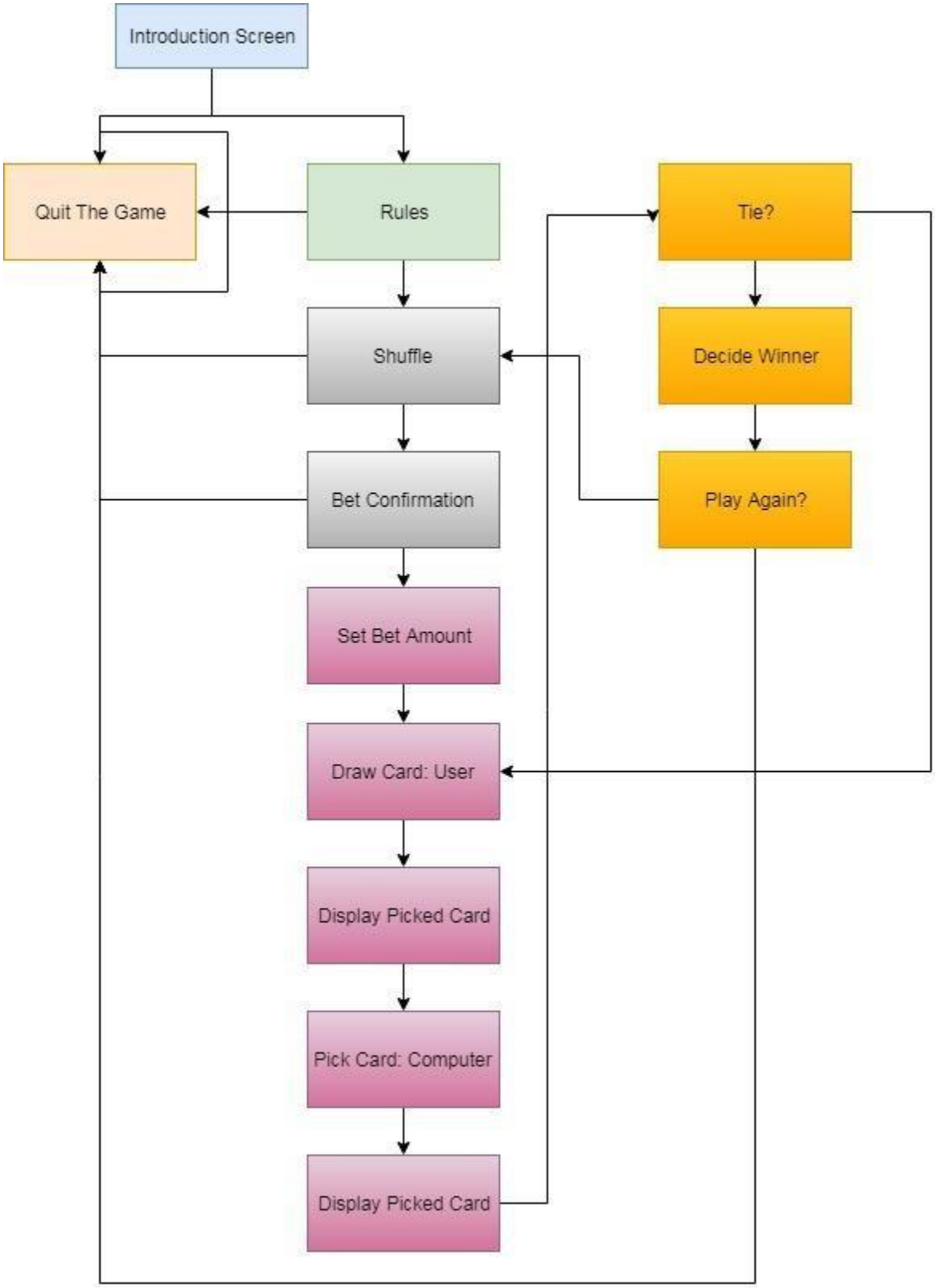
Winning or Losing:

If the user wins, then the coins they had bet must be awarded. If they user loses the bet, the coins they had bet must be subtracted from the total.

Repeating Loop:

User will be given the choice to either play again or leave. If one chooses to play again, then the shuffle button and leave button will appear again. If the user decides to leave, then only the leave button will appear.

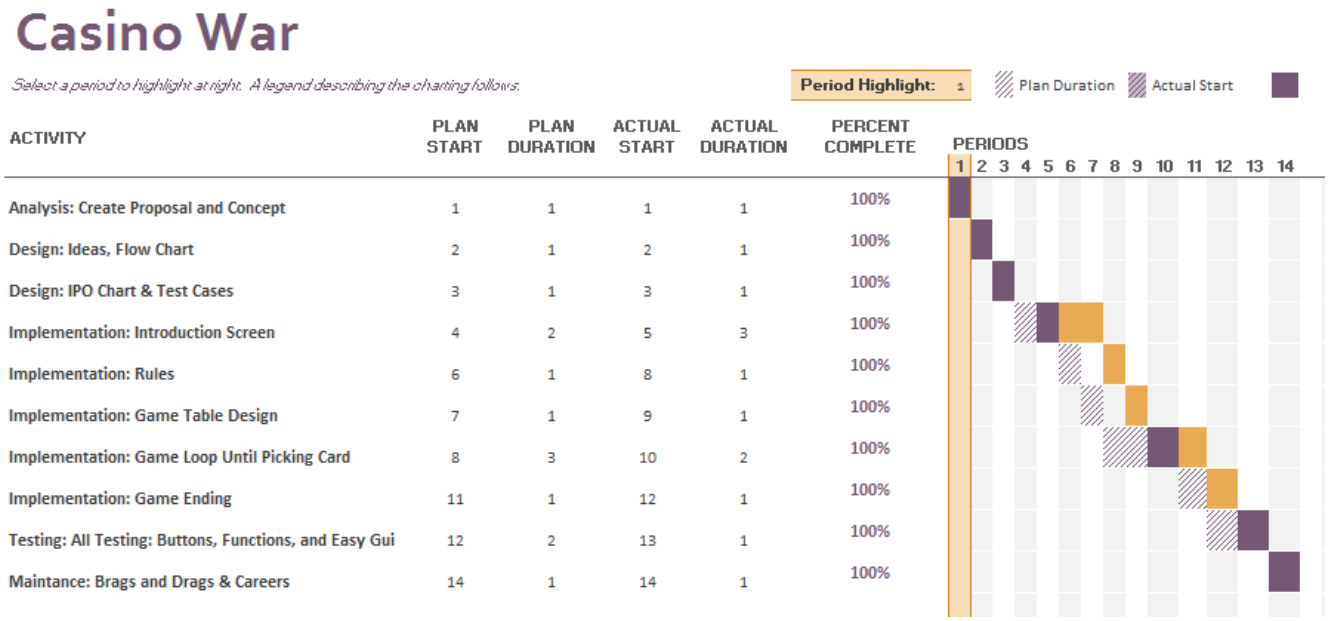
Flow Chart:


Introduction Screen
Quit The Game
Rules
Tie?
Shuffle
Decide Winner
Bet Confirmation
Play Again?
Set Bet Amount
Draw Card: User
Display Picked Card
Pick Card: Computer
Display Picked Card

IPO Chart:

Input	Processing	Output
Introduction Screen	End Program	----------
Rules	Loop a rules screen with images	Leave Button Start
Game Loop: Leave or Shuffle	End Program Shuffle Card	Leave Button Shuffle Button
Game Loop: Leave or Bet Confirmation	Confirm the bet Leave	Set Bet Amount
Game Loop: Set Bet Amount	Set an amount bet 0 to the # of coins	Pick card
Game Loop: Pick	Display Card	User Card Pick
Game Loop: Pick	Display Card	Compare Cards
Compare Cards	Check for tie, winner position of card	Play Again
Play Again	Play Again Quit	Start Game Loop Again Exit

Grant Table:



Testing Cases:

Buttons:

Button:									
Functioning:									

EasyGUI:

GUI:									
Taking Variables:									
Functioning:									

Functions and Procedure:

Functions:									
Functioning:									

IMPLEMENTATION:

Plans Sacrificed:

- Animations: Some animations were not created because of time limits and knowledge restrictions
- Sounds: Sounds were not created because of time limits
- Timer: Timer was not set because of knowledge limitations

Problem Log

Problems	Solutions
Unable to start a new screen	- Started another gameloop - Screen.fill command was used
Making a button	- Used a function with parameters - Used online resources to find mouse position and click commands
Making the background for the game table	- Used Photoshop
Making input boxes using pygame	- Used easygui instead
Display number of coins on each page	- Used .format to update the game
Shuffle button will stay on and the bet button will overlap it	- Used Boolean command to make shufflebutton=False
Setting pixel (y,x) for each button	- Mainly experimented with different options
Text displaying under the game table	- Put the text command after the table graphics is displayed
Bet button looping	- Used boolean commands
Unable to access the cards list	- Globaled it in the function
Unable to reload a new list after each round	- A new list was created /updated after each round of play
Unable to double the bet on go to war	- Did amount= amount *2
Runtime error for asking the user to play again	- Used Booleans command
Game would continue when coins = 0	- Used selection if statements to quit the game
Not able to tell if a tie has occurred	- Used 13 different selection statements to determine if both human and computer choose the same number

Unable to start the game again	Used Boolean expression to turn only the shuffle and leave buttons on
Unable to close the game	Used Boolean expression to turn only the leave button on
Rules screen would not display	Made a button
Program would respond late	Lowered the ffs

Code:

The Code of this program can be found in the Culminating Folder

User Guide:

1. A deck of card is shuffled.
2. The player places a bet and the dealer matches his bet with the player's.
3. The player picks the top card from the deck and the dealer follows the same step
4. The two cards picked from the deck are then compared.
5. The higher card in a ranking system wins the bet and the dealer must pay you, however if the dealer's card is better, you must pay him.
6. The card ranking follows from worst to best: 2, 3, 4, 5, 6, 7, 8, 9, 10, J, Q, K, A.
7. In a case of tie:
8. the player has choice to either surrender or to go to war again.
9. If the player surrenders, he will play receive half of previous bet back.
10. If the player decides to go to war again, he will place a bet again, however this time it will be the double of bet previously placed.

Commons Rules:

1. In a game, the player has a limit of losing 1,000 chips
2. The cards are shuffled each time a winner is chosen
3. The cards are randomly chosen: there is no bias for either human player or the computer
4. The player can only spend the money he or she has
5. The player can cash-out or leave after the winner is decided.

TESTING:

Buttons:

Button:	Quit	Next	Shuffle	Bet Now	Pick a card	Start
Functioning:	Yes	Yes	Yes	Yes	Yes	Yes

EasyGUI:

GUI:	Shuffling Card	Betting Round Confirmation	Taking Bet Amount	Card Picking Prompt	Bet Winner	Bet Again	Tie: Go to War or Surrender
Taking Variables:	N. A	Yes	Yes	N. A	N. A	Yes	Yes
Functioning:	Yes	Yes	Yes	Yes	Yes	Yes	Yes

Functions and Procedure:

Functions:	The Cards Are Shuffled	The Rules Screen Starts	Updates The Amount Of Coins	Display Card	Pick The Top Card	Shows The Amount Bet	Tie: Doubles The Amount Of Bet On Go To War Surrender Leave And Take Half The Bet	Play Again: Starts The Loop When Told To Restart The Game	End Games: Only displays the Leave button
Functioning:	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes

Notes:

- Printing value of variables
- No logical or syntax errors were encountered while testing the program
- All functions are working properly

MAINTANCE:

Brags and Drags:

Every programmer experiences brags and drags when creating a simple program like rock, paper, and scissors or a complex app. I was nonetheless also experienced some brags and drags while making Casino War. First, some drags experienced while making this program was encountering the "choppiness of pygame", changing the schedule of pygame and easygui commands and creating delay. To encounter the choppiness of the images on pygame commands, I used photoshop to create new images and background and images. To schedule when pygame and easygui commands were executed I used time commands like time.delay(#). Second, as a young programmer, I have many brags to talk about. I used many Boolean statements to control when buttons will appear and disappear. Next, I added many visuals to make the program imagined into reality. Furthermore, I also used the .format command to load up the images of the cards. I set a variable when the top card is picked from the pile(list), subsequently I changed the name of card to the name of objects in the list.