|  |  |  |  |
| --- | --- | --- | --- |
| Data Item | Data type | Validation | Sample Data |
| DOC | Cards[51] |  | “Ace”, “Hearts”, Properties.Resources.ACEHEARTS,(100,100) |
| RNG | Random |  | 20 |
| P1Cards | Cards[7] |  | “Ace”, “Hearts”, Properties.Resources.ACEHEARTS,(80,250) |
| P2Cards | Cards[7] |  | “Ace”, “Hearts”, Properties.Resources.ACEHEARTS,(80,20) |
| Back | Cards |  | “”,””,Properties.Resources.BackCard,(120,130) |
| LimboCard | Card |  | “Ace”, “Hearts”, Properties.Resources.ACEHEARTS,(400,250) |
| DeckDisplay | List<Cards> |  | “Ace”, “Hearts”, Properties.Resources.ACEHEARTS,(120,130) |
| StackDisplay | List<Cards> |  | “Ace”, “Hearts”, Properties.Resources.ACEHEARTS,(200,130) |
| NameHouse | String[8] |  | “Ace Hearts” |
| P1No | Int[7] |  | 2 |
| P2No | Int[7] |  | 50 |
| Deck | List<int> |  | 49 |
| Stack | List<int> |  | 27 |
| PriorCardName | string |  | “Two Hearts” |
| Limbo | int |  | 0 |
| NumberOfMoves | int |  | 30 |
| P1Points | Float |  | 6f |
| P2Points | Float |  | 10f |
| PickFromDeck | bool |  | False |
| DeckValidation | bool |  | True |
|  |  |  |  |

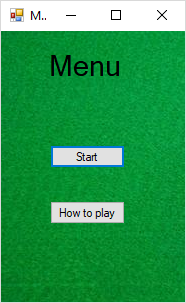
Flow Diagram

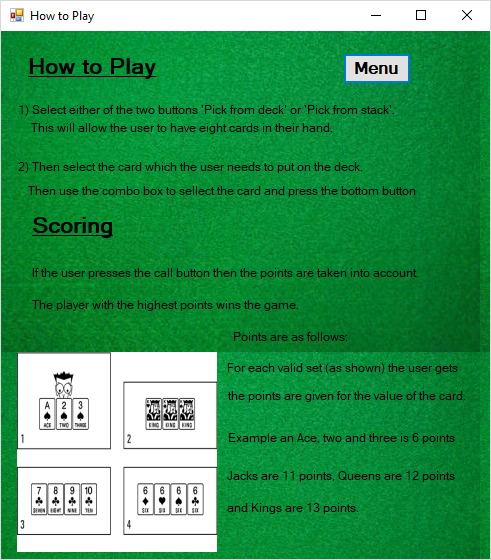
IPSO

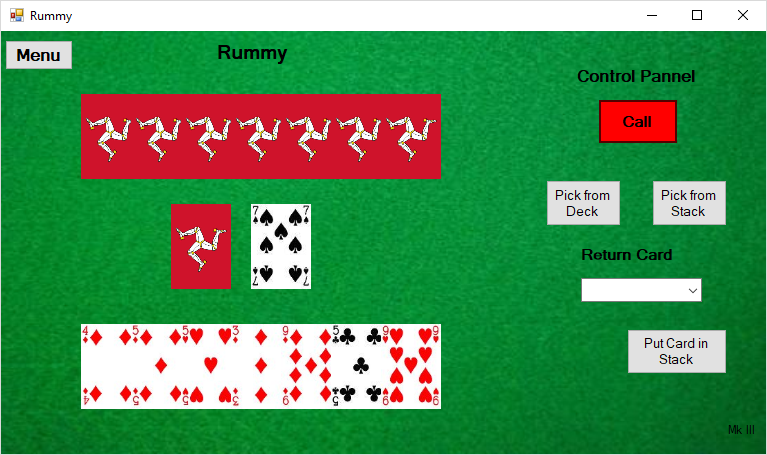
|  |  |  |
| --- | --- | --- |
| IPSO | Program Section | Item |
| Input |  |  |
| Processing |  |  |
| Storage |  |  |
| Output |  |  |

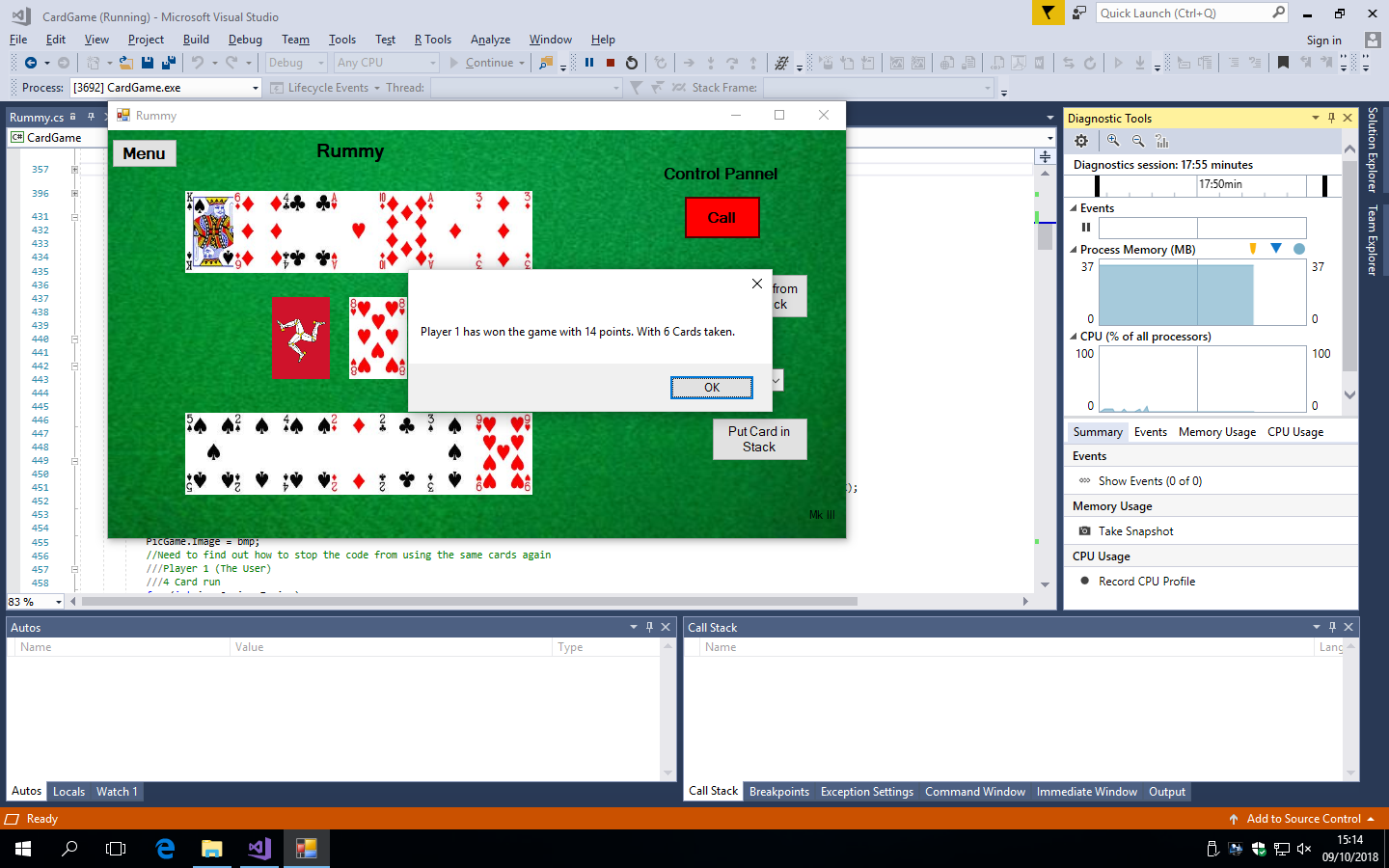
Data Dictionaries

Interface Design









**Algorithms**

Players Cards (Cards[] DOC, Cards[] P1Cards, int[] P1No, int[] P2No, Cards[] P2Cards)

For i = 0 to 2

For CD = 0 to 8

If I = 0 Then

P1Cards[CD] ← DOC[P1No[CD]]

P1Cards[CD].Location ←(30+ (CD \* 50), 250)

Next i

Else if

If I = 1 Then

P2Cards[CD] ← DOC[P2No[CD]]

P2Cards[CD].Location ← (30+ (CD \* 50), 20)

End If

End For

End For

End Sub

**Justification:**

Why

Sub DecPlayerCards(int[] P1No, int[] P2No)

List[] CardsAvailable ← new List[n];

Int PlayerCard;

Random RNG ← new Random();

For i ← 0 to 52

CardsAvailable.Add(i);

Next i

End For

For I ← 0 to 7

PlayerCard ← RNG(CardsAvailabel.count);

Player1No ← CardsAvailable[PlayerCard];

CardsAvailable.RemoveAt(PlayerCard);

Next i

End For

For I ← 0 to 7

PlayerCard ← RNG(CardsAvailabel.count);

Player2No ← CardsAvailable[PlayerCard];

CardsAvailable.RemoveAt(PlayerCard);

Next i

End For

End Sub

**Justification:**

Why

Sub DeclareCards(Cards[] DeckofCards)

For I ← 0 to 52

If I < 13 Then

DeckofCards[i].House ← “Clubs”;

Else if (I >= 13 AND I < 26) Then

DeckOfCards[i].House ← “Hearts”;

Else if (I >= 26 AND I < 39) Then

DeckOfCards[i].House ← “Diamonds”;

Else if (I >= 39 AND I < 52) Then

DeckOfCards[i].House ← “Spades”;

End If

Int MODNo ← I Mod 13;

If ModNo ← 0 Then

DeckOfCards[i].Name ← “Ace”;

Else If ModNo ← 1 Then

DeckOfCards[i].Name ← “Two”;

Else If ModNo ← 2 Then

DeckOfCards[i].Name ← “Three”;

Else If ModNo ← 3 Then

DeckOfCards[i].Name ← “Four”;

Else If ModNo ← 4 Then

DeckOfCards[i].Name ← “Five”;

Else If ModNo ← 5 Then

DeckOfCards[i].Name ← “Six”;

Else If ModNo ← 6 Then

DeckOfCards[i].Name ← “Seven”;

Else If ModNo ← 7 Then

DeckOfCards[i].Name ← “Eight”;

Else If ModNo ← 8 Then

DeckOfCards[i].Name ← “Nine”;

Else If ModNo ← 9 Then

DeckOfCards[i].Name ← “Ten”;

Else If ModNo ← 10 Then

DeckOfCards[i].Name ← “Jack”;

Else If ModNo ← 11 Then

DeckOfCards[i].Name ← “Queen”;

Else If ModNo ← 12 Then

DeckOfCards[i].Name ← “King”;

End If

Next I

End For

* Iterates for the 52 elements of the array and gives them the images of the different 52 cards.

Later…

End Sub

**Justification:**

Why

Sub DeclareDeck (List<int> Deck, ref int[] P1No, ref int[] P2No)

Int[] Player12Combined ← new int[14];

For I ← 0 to 14

If I <7 Then

Player12Combined[i] = P1No[i];

Else if I >= 7 Then

Player12Combined[i] = P2No[i- 7];

End If

Next I

End For

For I ← 0 to 52

Deck.Add(I);

Next I

End For

For I ← 0 to 14

For J ←0 to 52

If Player12Combined[i] ← J Then

Deck.Remove(J);

End If

Next J

End For

Next I

End For

End Sub

**Justification:**

Why

Sub DeckCards (Cards[] DOC, List<int> Deck, List<Cards> DeckDis)

For I ← 0 to Deck.Count

DeckDis.Add(Deck[I]);

Next I

End For

End Sub

**Justification:**

Why

Sub StackCards (List<int> Stack, List<Cards> StackDis, List<Cards> DeckDis, List<int> Deck, Cards[] DOC, Random RNG)

Int RNGCount ← 0;

Int CurrentStack ← 0;

RNGCount ← RNG.Next(Deck.Count);

CurrentStack ← Deck[RNGCount];

Stack.Add(CurrentStack);

Deck.Add(CurrentStack);

StackDis.Add(DOC[CurrentStack]);

DeckDis.Remove(DOC[CurrentStack]);

StackDis.ToArray();

For I ← 0 to StackDis.Count

StackDis[i].Location ← new Point(200,130);

Next I

End For

StackDis.ToList();

End Sub

**Justification:**

Why

Sub CheckDeck ()

For I ← 0 to Stack.Count

Do

If Deck[ElementLim ← Stack[I]] Then

ElementLim ← RNG.Next(Deck.Count);

CheckDeck(Stack, Deck, ElementLim);

End If

While Deck[ElementLim] ← Stack[I]]

Next I

End For

End Sub

**Justification:**

Why

Sub ScoringRules

* The Program iterates through all of the Player1’s Cards
* If 3 or 4 of the Players Hand in a Sequence (Consecutive order)
* Then Player1Points Equals Player1 Points plus the Collective points of the three or four cards.
* If 3 or 4 of the of the players hand is then same number or face of the card Then
* Player1Points Equals Player1Pointsplus the other three or four cards, selected.
* Cards then added to an array for future detection of the cards if they are used in future scoring.
* The Program iterates through all of the Player2’s Cards
* If 3 or 4 of the Players Hand in a Sequence (Consecutive order)
* Then Player1Points Equals Player2 Points plus the Collective points of the three or four cards.
* If 3 or 4 of the of the players hand is then same number or face of the card Then
* Player2Points Equals Player1Pointsplus the other three or four cards, selected.
* Cards then added to an array for future detection of the cards if they are used in future scoring.

**Justification:**

Why

15th October

Objectives

Rummy

* Update display of cards in the picture grid
* Declare the 52 cards in the class.
* It needs to split the DOC array into 6 random groups:
* Player 1 Picks the button Pick from Dec, this will use a Random Number of the Deck
* Player 1 Pick the button Pick from stack, it will take the last card on the stack list.
* Moves card selected to the limbo variable
* Player 1 using a combo box pick of on the eight card names and houses to remove from their hand and put on the stack.
* Player 1 Picks the Button Call: Loops through the cards in player 1’s and the opposing players hand and using the rules of the game.
* Player 2 in a selection of Neural Networks, which contribute to the decisions similar to player 1’s decisions of playing the game.