Pseudo Code for one-on-one Cribbage game against the Computer

Begin Here

//Initialize the Random Number Seed

//Declare Variables

//Input and Output for fstream

//Score of the Player

//Score of the Computer

//Counter for the Round

//Round Score

//Average Score per round throughout Game

//Number of Cards in a Deck

//Random Numbers for Player Cards

//Random Number for Starter Card

//Random Numbers for Computer Cards

//File to Write Scores to

//File to Pull Information From

//Suit to Be Assigned as String

//Number to be Assigned as String

//Player's Cards

//Computer Cards for Play Phase

//Starter Card

//Cards Contributed to the Crib by Computer

//Cards Selected by Player to Put in Crib

//Cards Played By Player in Play Phase

//Cards in Player's Hand during Play Phase

//Cards in Computer's Hand During Play Phase

//~1/3 Max Score in a Game of Cribbage

//Determine who goes first in order of Play//Initialize Variables

//52 Possible Cards to Draw (A,2-10,J,Q,K, then suits)

//Play a Round

//Determine whether or not the Game Continues

//Initialize File Parameters for Card Generation

//Increment Round

//"Shuffle" the Cards

//Generate the First and Second Numbers

//Make Sure Numbers are not Duplicates

//Card 1

//Assign Value Utilizing Switch Case

//Spades

//Hearts

//Clubs

//Diamonds

//Card 2

//Assign Value Utilizing Two Switch Cases and String Addition

//First take the Random Number and Break it into Two Variables

//(Random Number Mod 13)+1 gives a Number Between 1 and 13 to Test

//13 Possible Card Values

//(Random Number Mod 4)+1 gives a Number Between 1 and 4 to Test

//4 Possible Card Suits

//Test Value 1-13 and Assign

//Test Value 1-4 and Assign

//Add Strings to Create Card Value

//Generate Third Number

//Compare with Previous Numbers and Prevent Duplicates

//Card 3

//Assign Value Utilizing If/Else If Statements

//Spades

//Hearts

//Clubs

//Diamonds

//Generate Fourth Number

//Compare with Previous Numbers and Prevent Duplicates

//Card 4

//Assign Value Utilizing Independent If Statements

//Spades

//Hearts

//Clubs

//Diamonds

//Generate Fifth Number

//Compare with Previous Numbers and Prevent Duplicates

//Card 5

//Assign Value Utilizing Ternary Operators

//Spades

//Hearts

//Clubs

//Diamonds

//Generate the Rest of the Cards at once to Reduce Code

//Generate Sixth Number

//Compare with Previous Numbers and Prevent Duplicates

//Generate Seventh Number for Starter Card

//Compare with Previous Numbers and Prevent Duplicates

//Generate Opponent's hand

//Compare with Previous Numbers and Prevent Duplicates

//Compare with Previous Numbers and Prevent Duplicates

//Compare with Previous Numbers and Prevent Duplicates

//Compare with Previous Numbers and Prevent Duplicates

//Compare with Previous Numbers and Prevent Duplicates

//Compare with Previous Numbers and Prevent Duplicates

//Cards 6 Through 13 for the Rest of the Round

//Utilize fstream to Assign Value

//Read in Value from file to cardIn

//Last Card in Player's Hand

//Starter Card for the Round

//First Card for Computer

//Second Card for Computer

//Third Card for Computer

//Fourth Card for Computer

//First Card for the Computer Crib

//Second Card for the Computer Crib

//Display Hand

//Ask for Cards to add to the Crib for the Round

//Hold the Players Hand for Teaching Purposes

//Make Input consistent with Recorded Card Values

//Break Up String Entries and Cast to Upper Case

//Test to see if Input is Valid

//Custom Message for Invalid Input

//If Input is Invalid, Restart Loop

//Make Input consistent with Recorded Card Values

//Break Up String Entries and Cast to Upper Case

//Test to see if Input is Valid

//Custom Message for Invalid Input

//If Input is Invalid, Restart Loop

//Display Selections to the Crib

//Display Hand without Crib Selection and Officially Begin Round

//If Cards One and Two are Thrown into the Crib

//If Cards One and Three are Thrown into the Crib

//If Cards One and Four are Thrown into the Crib

//If Cards One and Five are Thrown into the Crib

//If Cards One and Six are Thrown into the Crib

//If Cards Two and Three are Thrown into the Crib

//If Cards Two and Four are Thrown into the Crib

//If Cards Two and Five are Thrown into the Crib

//If Cards Two and Six are Thrown into the Crib

//If Cards Three and Four are Thrown into the Crib

//If Cards Three and Five are Thrown into the Crib

//If Cards Three and Six are Thrown into the Crib

//If Cards Four and Five are Thrown into the Crib

//If Cards Four and Six are Thrown into the Crib

//If Cards Five and Six are Thrown into the Crib

//Modulus round Number by 2 to Indicate if True or False

//round%2 will equal either 0 or 1

//Pass Boolean Value into Switch Case

//If uGoFirst=1 (or anything but 0), that is equivalent to "true"

//Each Case Starts With A Loop for Input Validation

//Get Input for Round

//Display Input after Converting to uppercase

//Message For Invalid Input

//Check if Card is in Hand or Already been Played

//Award Points for Play

//Give 2 Points Pair

//Get Input for Round

//Display Input after Converting to uppercase

//Message for Invalid Input

//Check if Card is in Hand or has Already been Played

//Award Points for Play

//Give 12 Points for 4 of a Kind

//Give 6 Points for 3 of a Kind

//Give 2 Points for a Pair

//Get Input for Round

//Display Input after Converting to uppercase

//Message for Invalid Input

//Check if Card is in Hand or Already been Played

//Award Points for Play

//Give 12 Points for 4 of a Kind

//Give 6 Points for 3 of a Kind

//Give 2 Points for a Pair

//Get Input for Round

//Display Input after Converting to uppercase

//Message for Invalid Input

//Check if Card is in Hand or Already been Played

//Award Points for Play

//Give 12 Points for 4 of a Kind

//Give 6 Points for 3 of a Kind

//Give 2 Points for a Pair

//Nothing to Check here, just Output the Cards generated

//So the player can Play against them

//This Game is against pure Chance

//Give 6 Points for 3 of a Kind

//Give 2 Points for a Pair

//Give 12 Points for 4 of a Kind

//Give 6 Points for 3 of a Kind

//Give 2 Points for a Pair

//Give 12 Points for 4 of a Kind

//Give 6 Points for 3 of a Kind

//Give 2 Points for a Pair

//Give 12 Points for 4 of a Kind

//Give 6 Points for 3 of a Kind

//Give 2 Points for a Pair

//Break from True Case Switch Statement

//If uGoFirst=0, that is equivalent to "false"

//Nothing to Check here, just Output the Cards generated

//So the player can Play against them

//This Game is against pure Chance

//The Switch Case Counts Down Because of the Decrementing

//Give 12 Points for 4 of a Kind

//Give 6 Points for 3 of a Kind

//Give 2 Points for a Pair

//Give 12 Points for 4 of a Kind

//Give 6 Points for 3 of a Kind

//Give 2 Points for a Pair

//Give 12 Points for 4 of a Kind

//Give 6 Points for 3 of a Kind

//Give 2 Points for a Pair

//Get Input for Round

//Display Input after Converting to uppercase

//Message for Invalid Input

//Check if Card is in Hand or Already been Played

//Award Points for Play

//Give 6 Points for 3 of a Kind

//Give 2 Points for a Pair

//Get Input for Round

//Display Input after Converting to uppercase

//Message for Invalid Input

//Check if Card is in Hand or Already been Played

//Award Points for Play

//Give 12 Points for 4 of a Kind

//Give 6 Points for 3 of a Kind

//Give 2 Points for a Pair

//Get Input for Round

//Display Input after Converting to uppercase

//Message for Invalid Input

//Check if Card is in Hand or Already been Played

//Award Points for Play

//Give 12 Points for 4 of a Kind

//Give 6 Points for 3 of a Kind

//Give 2 Points for a Pair

//Get Input for Round

//Input for Play

//Display Input after Converting to uppercase

//Message for Invalid Input

//Check if Card is in Hand or Already been Played

//Award Points for Play

//Give 12 Points for 4 of a Kind

//Give 6 Points for 3 of a Kind

//Give 2 Points for a Pair

//Break from False Case Switch Statement

//End of Round: Determine whether or not the Game Continues

//if the Player Wins

//If the Computer Wins//If there is a tie

//Initialize File Parameters for Card Generation

//Write Scores to File for Storage

//Exit the Program - Cleanup