Diagram, table

Description automatically generated with medium confidence

UML for task

Pseudocode for task

1. Create variables and objects.
2. Get inputs.
   * + 1. Call setter SetName() and accuracy for number of players (in case of program, 2)

Ask user input for name within setter.

Ask user input for accuracy within setter for bull accuracy and single accuracy.

* + - 1. Read input name into player.name
      2. Read input bull accuracy into player.bullacc
      3. Read input single accuracy into player.singleacc
      4. Read input double accuracy into player.DoubleAcc
      5. Read input double accuracy into player.TrebleAcc
      6. Ask user which gamemode they would like to play between 301, 501 or interactive 501
      7. Ask user how many times they want the simulation to run.

1. For (number of times user has asked) run simulation
2. While score of Player one or player two does not equal 0 continue
3. If gamemode is 301 then do section below
   * + 1. Check score if (score = 50) do bullthrow()
       2. Else computer chooses the highest value for user’s throw.

if (score >= 100) do bullthrow() and add one to num of throws

if (score <=99 AND score >= 70) do single throw(20) and add one to num of throws

if (score <= 69 And Score > 0) do single throw(currentscore -50) and add one to num of throws

1. Calculate 301
   * + 1. bullThrow()

Uses a random function

Reads the random value against their bullAccuracy

If (random <= bullAccuracy) do score – 50 and bull hit+1

END IF

Else random score value between 1-20

Score – data value from random score

Prints new score onto console window

END ELSE

If (score < 50 && score > 0) score + throw

Prints message “must be 50 exactly then get a bullseye”.

END IF

* + - 1. singleThrow(throw)

Uses a random function

Reads the random value against singleAccuracy

If(random <= single accuracy) do score – throw

Prints new score onto console window.

END IF

Else if (random <= single accuracy + 10) do score – score - boardNeighbour[0][throw]

Prints new score onto console window

END IF

Else if (random <= single accuracy +20) do score – boardNeighbour[1][throw]

Prints new score onto console window

END IF

ELSE do score - random between 1 – 20

Prints new score onto console window

If (score < 50 && score > 0) score + throw

Prints message “must be 50 exactly then get a bullseye”.

END IF

* + - 1. switch (num of throws) case 7-10

adds one onto whatever case it falls into between 7 to 10

* + - 1. repeat for other player.

END WHILE

* + - 1. divides each case from switch by total games and times by 100 to get percent frequency of games

1. Output results 301
   * + 1. Prints name of winner of game onto console window with how many throws and how many of them were bullseyes.

END FOR

* + - 1. prints frequency of games for each case value onto console window.
      2. Prints total throws.

END IF

1. If gamemode is 501 then do section below
   * + 1. Uses random function for flipping a coin 50/50 and assign value to choose variable.
       2. For matches until matches is equal to 13
       3. Call reset player one round winner and reset player two round winner.
       4. For sets until sets is equal to 5
       5. Set up both player scores as 501
       6. Print who will go first
       7. While both players score is not equal to zero do next steps
       8. For darts until darts is equal to 3
       9. Check score if (score = 50) do bullthrow()
       10. Else computer chooses the highest value for user’s throw.
       11. If (score >= 62)

If (score % 2 does not equal 0 meaning no remainder)

do trebleThrow(19 throw value) and add one to num of throws to make sure it is an even number

Prints new score onto console window

else do treblethrow(20) and add one to num of throws

Prints new score onto console window

* + - 1. Else if (score is less than 62 and score is greater than 51) do singlethrow(score – 50) and add one to num of throw
      2. Prints new score onto console window
      3. Else if (score is less than 50 and score is greater than 40) do singlethrow(10) and add one to num of throw
      4. Prints new score onto console window
      5. Else if (score is less than or equal to 40 and score is greater than 4) do

If (score % 2 does not equal 0) do singlethrow(3) and add one to num of throw

Prints new score onto console window

Else do doublethrow(score/2) and add one to num of throw

Prints new score onto console window

* + - 1. Else if (score equal 2) do doublethrow(1) and add one to num of throw
      2. In dart values[3] to hold previous throws in case they go below zero so it can add them back on
      3. If (score equal 1 or score less than 0) do print “sorry must finish on bullseye or double”

For (R less than 3; R++) do add on to score dartvalues[R]

* + - 1. If (score Player one or Player two == 0) do break while loop
      2. Repeat for Player 2
      3. End while
      4. if player one reaches score 0 set round winner for player one up 1 and print “you have won this round player 1”
      5. vice versa for player 2
      6. if rounds won for player on or player 2 equals 3
      7. print “congrats you have won the set, you have won” setwins “out of 13 matches”
      8. vice versa for player 2
      9. end for
      10. if (player one or player 2 setwins equals 7) print “yay you won the overall match and reset round winner
      11. end for
      12. int freqnum = \_1
      13. for (int I = 5 I is greater than 5 I--)

if (player one set winner == 7 && player two set winner == I+1) do frequency[freqnum] += 1

If (Player one set winner == I +1 && Player two set winner == 7) do frequency[freqnum] += 1

* + - 1. End for
      2. For (int I = 0; I less than 6 I++)

Winratefrequncies[i] = (frequency[i] / numofgames) \* 100

Print frequencies for how many games one with a certain number of set wins for player 1 i.e. 7:1, 7:3 etc etc

End for

* + - 1. For (int I = 6; I less than 12 I++)

Winratefrequncies[i] = (frequency[i] / numofgames) \* 100

Print frequencies for how many games one with a certain number of set wins for player 2 i.e. 1:7, 3:7 etc etc

End for

* + - 1. Print “simulation of 501 finished”

1. Calculate 501 functions
   * + 1. BullThrow()

Random equals random function

While (throwvalue is equal to 50) do

If (Random is less than or equal to Bullacc - 20) print “you hit the bullseye” and return 50

else if(random is less than or equal to bullacc + 5) print you hit the outer ring of the bullseye” and return 25

else print “you missed the bullseye” and return a random number between 1-20

end while

while (throwvalue is equal to 25) do

if(random is less than or equal to bullacc + 5) print you hit the outer ring of the bullseye” and return 25

else If (Random is less than or equal to Bullacc - 20) print “you hit the bullseye” and return 50

else print “you missed the outer ring” and return a random number between 1-20

end while

* + - 1. TrebleThrow()

Random equals random function

If (random is less than or equal to trebleaccuracy) print “you hit your target:” and return throwvalue \* 3

Else if (random is less than or equal to trebleaccuracy + 10) print “ you hit: throwvalue” return throwvalue

Else if (random is less than or equal to trebleaccuracy + 13) print “you hit: 3 \*throwvalue left side” return 3 \* throw value left side

Else if (random is less than or equal to trebleaccuracy + 16) print “you hit: 3\* throwvalue right side” return 3\* throwvalue rightside

Else if (random is less than or equal to trebleaccuracy +18) print “you hit: throw value left side” return throwvalue leftside

Else print “you hit: throw value right side” return throwvalue right side

* + - 1. DoubleThrow()

Random equals random function

If (random is less than or equal to doubleaccuracy) print “you hit your target:” and return throwvalue \* 2

Else if (random is less than or equal to doubleaccuracy + 5) print “ you missed entirely” return 0

Else if (random is less than or equal to doubleaccuracy + 10) print “you hit: throwvalue” return throw value

Else if (random is less than or equal to doubleaccuracy + 13) print “you hit: 2 \*throwvalue left side” return 2 \* throw value left side

Else if (random is less than or equal to doubleaccuracy + 16) print you hit: 2\* throwvalue right side” return 2\* throwvalue rightside

Else if (random is less than or equal to doubleaccuracy +18) print “you hit: throw value left side” return throwvalue leftside

Else print “you hit: throw value right side” return throwvalue right side

* + - 1. singleThrow

Random equals random function

If (random is less than or equal to singleaccuracy) print “you hit your target:” and return throwvalue

Else if (random is less than or equal to singleaccuracy + 8) print “ you hit: throw value left side” return throwvalue left side

Else if (random is less than or equal to singleaccuracy + 10) print “ you hit: throw value right side” return throwvalue right side

Else if (random is less than or equal to singleaccuracy + 12) print “ you hit: throw value \* 3” return throwvalue \* 3

Else if (random is less than or equal to singleaccuracy + 14) print “ you hit: throw value \* 2” return throwvalue \* 2

Else if (random is less than or equal to singleaccuracy + 16) print “ you hit: throw value \* 3 left side” return throwvalue \* 3 left side

Else if (random is less than or equal to singleaccuracy + 18) print “ you hit: throw value \* 3 right side” return throwvalue \* 3 right side

Else if (random is less than or equal to singleaccuracy + 20) print “ you hit:the outer ring of bullseye return 25

Else print “you hit: random value between 1-20” return random value between 1 - 20

1. If gamemode is 501 interactive do below
   * + 1. Uses random function for flipping a coin 50/50 and assign value to choose variable.
       2. For matches until matches is equal to 13
       3. Call reset player one round winner and reset player two round winner.
       4. For sets until sets is equal to 5
       5. Set up both player scores as 501
       6. Print who will go first
       7. While both players score is not equal to zero do next steps
       8. For darts until darts is equal to 3
       9. If (choice is player 1(the interactive player))

While (throwvalue does not equal 1-20 or 25 or 50) do

Ask user what value they would like to throw for either 1-20, 25 or 50

End while

If (user input is 25 or 50) do bullthrow()

Else ask user if it is single double or treble throw they would like to go for

Switch (userthrowtype)

Case 1 do singlethrow()

Case 2 do doublethrow()

Case 3 do treblethrow()

* + - 1. The functions from the switch in 501 interactive are the same as the ones from 501
      2. If (choice is player 2) do the same as 501 for player 2
      3. End for
      4. End while