CST8116 (22S) Exercise 05 Companion File

# Word Problems / Understand the Problem (Provided)

* The program is a variation on “guess the number”[1], the program will select a number at random between 1 to 10, the user will be prompted to enter a number between 1 to 10 and if it matches the randomly selected number[2] they win the game and the loop exits, if they do not get the number correct they are told they were too-high, or too-low, and the loop continues asking them to add another number.
* To add to the game-play the user is given ‘guess-fuel’ as an integer value, and each number they enter to make a guess costs guess-fuel in an equal amount. Example, if the guess-fuel is 27, and they guess 10, then they will have 17 guess-fuel remaining. Starting guess-fuel is the sum of the numbers from 1 to 10 divided by 2.
* The game ends if the user guesses the number, or if they run out of guess-fuel.
  + They can enter numbers greater than, or less than the guess-fuel remaining. If their last guess exceeds the guess-fuel remaining they are not stopped, however the guess-fuel is set to zero for end of game reporting.
* After the game let them know the number of guesses they made and if they won or lost.
* After each game, ask the user if they would like to play another game or exit the program.

Review the UML, pseucode, and starter-code provided, then

* create a flowchart for method reset(), and code the method in Java;
* create a flowchart for method run(), and code the method in Java;
* create a flowchart for method main(), and code the method in Java;
* update the programmer comments in the starter files where indicated
* Complete the trace tables provided, to test your flowcharts, then re-use the tables to test your Java program.

# Detailed UML Class Diagrams (Provided)



# Pseudocode (Provided)

Note that class User is not documented in pseudocode.

## Class Exercise05 method main

start

declarations

User user

HighLowGame game

String CONTINUE\_GAME = "Y"

String EXIT\_GAME = "N"

String shouldContinue = EXIT\_GAME

do

if shouldContinue = CONTINUE\_GAME then // case-insensitive comparison

game.reset()

endif

game.run()

shouldContinue = user.inputString("Would you like to play again? (Y/N)"

while shouldContinue = CONTINUE\_GAME // case-insensitive comparison

output "Program by Student Name" // change Student Name to your actual name

stop

## Class CheckGuessResult (all of it)

public class CheckGuessResult

declarations

private String message

private boolean isWin

public CheckGuessResult()

this("no game", false)

return

public CheckGuessResult(String message, boolean isWin)

this.message = message

this.isWin = isWin

return

public String getMessage()

return message

public void setMessage(String message)

this.message = message

return

public boolean isWin()

return isWin

public void setWin(boolean isWin)

this.isWin = isWin

return

public String toString()

declarations

String report

report = "CheckGuessResult : message " + message + ", "

report = report + "isWin " + isWin

return report

endClass

## Class HighLowGame (all of it)

class HighLowGame

private User user // class User is provided and complete

private Random random // class Random is java.util.Random

private num fuelAvailable // integer

private num numberToGuess // integer

private num MIN = 1

private num MAX = 10

public HighLowGame()

reset()

return

public num getFuelAvailable()

return fuelAvailable

public void setFuelAvailable(num fuelAvailable)

this.fuelAvailable = fuelAvailable

return

public num getNumberToGuess()

return numberToGuess

public void setnumberToGuess(num numberToGuess)

this.numberToGuess = numberToGuess

return

public void reset()

declaration

num sum = 0

// random.nextInt(MAX) returns a value from 0 to (MAX - 1)

// so add 1 to get a number between 1 to MAX inclusive

numberToGuess = random.nextInt(MAX) + 1

for num value = MIN to MAX step 1 // MIN to MAX inclusive

sum = sum + value

endfor

fuelAvailable = sum / 2

return

public void run()

declarations

CheckGuessResult checkGuessResult = null

String message

boolean isWon = false

num guessCount = 0 // int

num userNumber // int

output "Guess the number from " + MIN + " to " + MAX

output "You have " + fuelAvailable + " guess-fuel remaining."

while fuelAvailable > 0 AND isWon = false

guessCount = guessCount + 1

userNumber = user.inputInteger("guess: ")

// first loop control variable adjustment

fuelAvailable = fuelAvailable - userNumber

// let them have this last play, but prevent negative

// fuel values in the output

if fuelAvailable < 0 then

fuelAvailable = 0

endif

checkGuessResult = checkGuess(userNumber)

// second loop control variable adjustment

isWon = checkGuessResult.isWin()

message = checkGuessResult.getMessage()

output message // in game messaging

endwhile

message = reportGameResult(isWon, guessCount)

output message // after game report

return

private CheckGuessResult checkGuess(num guess) // Java would use (int guess)

declarations

boolean isWin = false

String message

if guess = numberToGuess then

message = "you guessed the number"

isWin = true

else

if guess < numberToGuess then

message = "too low, "

else

message = "too high, "

endif

message = message + fuelAvailable + " guess-fuel remaining"

endif

declarations

CheckGuessResult checkGuessResult(message, isWin)

return checkGuessResult

private String reportGameResult(boolean isWon, num guessCount)

declarations

String message

if isWon = true then

message = "You win! It took you "

else

message = "You did not win, you used "

endif

message = message + guessCount + " guesses."

return message

endClass

# Sample Trace Tables (Provided, but incomplete)

## For loop in method reset()

Before loop:

* MIN is 1, MAX is 10, value is loop control variable
* value starts equal to min, sum starts at zero

|  |  |  |  |
| --- | --- | --- | --- |
| value | value <= MAX ? | sum | Notes |
| 1 | Yes | 1 | Loop runs, sum is increased by 1 |
| 2 | Yes | 3 | Loop runs, sum is increased by 2 |
|  |  |  |  |

After the loop:

* Sum should be 55

## While loop in method run() trace table for running out of fuel user does not guess answer

Before loop:

* fuelAvailable is 27, isWon is false
* fuelAvailable decreases by the user guess value each loop until zero
* Test assumes numberToGuess is 1, but the user enters 10 each time.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| fuelAvailable | isWon | fuelAvailable > 0 &&  isWon == false ? | User input | Notes |
| 27 | false | Yes | 10 | Loop runs, user enters 10, fuelAvailable becomes 17, isWon remains false |
| 17 | false | Yes | 10 | Loop runs, user enters 10, fuelAvailable becomes 7, isWon remains false |
|  |  |  |  |  |

After the loop:

* Game should be over with user lost message, and fuelAvailable at zero

## While loop in method run() trace table for user does guess answer

Before loop:

* fuelAvailable is 27, isWon is false
* fuelAvailable decreases by the user guess value each loop until zero
* Test assumes numberToGuess is 4, and the user enters 5 then 4

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| fuelAvailable | isWon | fuelAvailable > 0 &&  isWon == false ? | User input | Notes |
| 27 | false | Yes | 5 | Loop runs, user enters 5, fuelAvailable becomes 22, isWon remains false |
| 22 | false | Yes | 4 | Loop runs, user enters 4, fuelAvailable becomes 18, isWon becomes true |
| 18 |  |  |  |  |

After the loop:

* Game should be over with user won message, and fuelAvailable remaining as 18

## While loop in method run() trace table for invalid input

* Use a third trace table to document one case (row) each of these scenarios. What happens to the fuelAvailable?
  + User guesses zero
  + Use guesses a negative number

## Do-While loop in method main() testing that program continues and exits

Before loop:

* CONTINUE\_GAME is “Y”, EXIT\_GAME is “N”, shouldContinue was set to EXIT\_GAME
* User input is assigned into shouldContinue before checking loop condition

|  |  |  |
| --- | --- | --- |
| User input | shouldContinue = CONTINUE\_GAME ? | Notes |
| y | Yes | User entered “y” to continue the game, loop continues |
| n |  |  |

# References:

[1] I Spy Code. (n.d.). High Low Guessing Game. Last Accessed March 14, 2022 Retrieved from

<https://ispycode.com/Java/Flow-Control/Examples/High-Low-Guessing-Game>

[2] Cay Horstmann. (2019). 6.9.1 Generating Random Numbers. Pages 210 to 211. Big Java Early Objects. 7th Ed. Wiley.

# Appendix: Sample Program Run

Guess the number from 1 to 10

You have 27 guess-fuel remaining

guess: **5**

too low, 22 guess-fuel remaining

guess: **8**

You guessed the number

You win! It took you 2 guesses

Would you like to play again(Y/N)?**y**

Guess the number from 1 to 10

You have 27 guess-fuel remaining

guess: **5**

too high, 22 guess-fuel remaining

guess: **3**

too low, 19 guess-fuel remaining

guess: **4**

You guessed the number

You win! It took you 3 guesses

Would you like to play again(Y/N)?**y**

Guess the number from 1 to 10

You have 27 guess-fuel remaining

guess: **10**

too high, 17 guess-fuel remaining

guess: **10**

too high, 7 guess-fuel remaining

guess: **10**

too high, 0 guess-fuel remaining

You did not win, you used 3 guesses

Would you like to play again(Y/N)?**n**

Program by Stanley Pieda

Note: The sample program run above was formatted to show user input in a **bold-black font with yellow highlighting**. The default color used in Eclipse is a light green.

# Appendix: UMLet UML as Plain Text

## Class Exercise05

Exercise05

--

--

\_+main(args:String[]):void\_

## Class HighLowGame

HighLowGame

--

-user:User

-random:Random

-fuelAvailable:int

-numberToGuess:int

\_-MIN:int = 1\_

\_-MAX:int = 10\_

--

+HighLowGame()

+getGuelAvailable():int

+setFuelAvailable(fuelAvailable):int):void

+getNumberToGuess():int

+setNumberToGuess(numberToGuess:int):void

+reset()

+run()

-checkGuess(guess:int):CheckGuessResult

-reportGameResult(isWon:boolean, guessCount:int):String

## Class CheckGuessResult

CheckGuessResult

--

-message:String

-isWin:boolean

--

+CheckGuessResult()

+CheckGuessResult(message:String, isWin:boolean)

+getMessage():String

+setMessage(message:String):void

+isWin():boolean

+setWin(isWin:boolean):void

+toString():String

## Class User

User

--

-input:Scanner

--

+inputInteger():int

+inputInteger(message:String):int

+inputDouble():double

+inputDouble(message:String):double

+inputString():String

+inputString(message:String):String