# GAME

/\*\*

\* Game class that contains the flow of the game.

\*/

import java.util.Scanner;

public class Game

{

public static void main(String[] args)

{

// Create the object of the Player class

Player newPlayer = new Player();

// Create the object of the LuckyDipGenerator class

LuckyDipGenerator luckyNumber = new LuckyDipGenerator();

// Set variables with initial values

int choice = -1, guesses = 0;

do

{

displayMenu();

Scanner input = new Scanner(System.in);

choice = input.nextInt();

switch (choice)

{

case 1:

System.out.println("Enter Player's Name:");

String name = input.next();

// Keep asking the name until user enters a valid name

while(name.trim().equals("")){

System.out.println("Player name must not be blank, try again.");

System.out.println("Enter Player's Name:");

}

newPlayer.initiate(name);

break;

case 2:

// Check if name exists (player exists) then prompt user to enter name

if (!newPlayer.nameExists())

{

System.out.println("Please create a player first. (hint: press 1)");

}

// Asking for player intial credit

else

{

System.out.println("Enter Player's Credit between 1 - 20 dollars:");

int credit = input.nextInt();

while (credit > 20 || credit < 1){

System.out.println("Please enter the credit between 1 - 20 dollars");

credit = input.nextInt();

}

newPlayer.setCredit(credit);

}

break;

case 3:

// Check if name exists (player exists) then prompt user to enter name

if (!newPlayer.nameExists())

{

System.out.println("Please create a player first. (hint: press 1)");

}

// Check if credit is 0 then prompt user to set the credit.

else if (newPlayer.getCredit() == 0)

{

System.out.println("Please add the credit first. (hint: press 2)");

}

else

{

// Generate the lucky number

luckyNumber.setLuckyNumber();

guesses = 3;

int guess = -1;

Boolean isWon = false;

while (guesses > 0){

System.out.println("Guess Number between 1 - 50");

// Get user input between 1 - 50

guess = input.nextInt();

int result = luckyNumber.guessLuckyNumber(guess);

// Check if the player guess is correct.

if (result == 0)

{

int winAmount = guesses\*5;

isWon = true;

newPlayer.addCredit(winAmount);

newPlayer.incrementWins(winAmount);

System.out.println("You won! $"+winAmount+" is credited to your balance. You have now $"+newPlayer.getCredit()+" balance.");

break;

}

else if (result == 1)

{

System.out.print("Wrong guess. The guessed number is greater than the lucky number.");

}

else

{

System.out.print("Wrong guess. The guessed number is less than the lucky number.");

}

if (guesses-1 > 0){

System.out.println(" You have " + (guesses - 1) + " guesses left.");

}

guesses--;

}

if (!isWon && guesses == 0)

{

// If guess is within +-5 of the margin then consider a win and add the consolidated price

if (luckyNumber.winWithInMargin(guess))

{

int winAmount = 1 + (int)(Math.random() \* 5);

newPlayer.incrementWins(winAmount);

System.out.println("\nYou are out of guesses, but your last guess is within ±5 of the lucky number so you won a consolation prize of $"+winAmount+".");

}

else

{

newPlayer.addCredit(-5);

newPlayer.incrementLosses();

System.out.println("\nYou lost, $5 is deducted from your credit balance!");

}

System.out.println("Your last guess is " + guess + " and the lucky number is " + luckyNumber.getLuckyNumber() + ".");

}

}

break;

case 4:

newPlayer.displayStastics();

break;

case 5:

System.out.println("This is a number guessing game. In this game, the user has to predit the lucky number within 3 guesses.");

System.out.println("This game has the following steps:");

System.out.println("Press 1 to create an account as a new player and type your name on the console.");

System.out.println("Press 2 to add credit in your account.");

System.out.println("Press 3 to start guessing the number. Please note that you will have only 3 guesses for each round.");

System.out.println("$15 will be credited to your account if your 1st guess is correct.");

System.out.println("$10 will be credited to your account if your 2nd guess is correct.");

System.out.println("$5 will be credited to your account if your 3rd guess is correct.");

System.out.println("$5 will be deducted from your account if none of the guesses is correct.");

System.out.println("All the best.");

break;

case 6:

System.out.println("Goodbye!.");

break;

default:

System.out.println("Invalid input, try again");

break;

}

}

while(choice != 6);

}

/\*\*

\* Method to display the menu of the game

\*/

public static void displayMenu(){

System.out.println("\nWelcome to the Number Guessing Game\n===============================\n(1) Set Up New Player\n(2) Add Credit\n(3) Play One Round\n(4) Display Player Statistics\n(5) Display Game Help \n(6) Exit Game\nChoose an option:");

}

}

# PLAYER

/\*\*

\* Player class that holds the credit, winnings, losses, rounds of the player.

\*/

import java.util.ArrayList;

public class Player

{

private String name = "";

private int initialCredit, credit, wins, losses, winnings;

// ArrayList to keep the track of the user history

private ArrayList<String []> history = new ArrayList<String []>();

// Initiate the player with the name and null credit.

public void initiate(String \_name)

{

this.name = \_name;

this.credit = 0;

this.initialCredit = 0;

this.wins = 0;

this.losses = 0;

this.winnings = 0;

history = new ArrayList<String []>();

}

// Return the name of the player

public String getName()

{

return this.name;

}

// If name is found (player exists) then return true else return false

public Boolean nameExists()

{

if (name.trim().equals(""))

{

return false;

}

return true;

}

// Set initial credit for the player

public void setCredit(int \_credit)

{

if (this.credit > 0)

{

this.credit += \_credit;

this.initialCredit += \_credit;

}

else

{

this.credit = \_credit;

this.initialCredit = this.credit;

}

}

// Adding more credit in the player's balance.

public void addCredit(int \_credit)

{

if ((this.credit + \_credit) < 0)

{

this.credit = 0;

}

else

{

this.credit += \_credit;

}

}

// Get the current credit (after winnings/losses)

public int getCredit()

{

return this.credit;

}

// If player wins, add the record to the list with the winning amount

public void incrementWins(int \_winAmount)

{

this.wins += 1;

history.add(new String [] { "Win", String.valueOf(\_winAmount) });

}

// If player loses, add the record to the list with the losing amount

public void incrementLosses()

{

this.losses += 1;

// If the current credit is below 5 then set credit to 0 (avoiding credits in negative). Else decrement the credit by 5.

if ((this.credit - 5) < 0)

{

history.add(new String [] { "Loss", String.valueOf(this.credit) });

}

else

{

history.add(new String [] { "Loss", "-5" });

}

}

// Function to display the complete stastics of the player

public void displayStastics()

{

// Calculate total rounds

int totalRounds = this.wins + this.losses;

System.out.println("Player's Name : " + this.name);

System.out.println("Rounds You Played : " + totalRounds);

System.out.println("Total Rounds Won : " + (this.wins));

System.out.println("Total Rounds Lost : " + (this.losses));

if (totalRounds==0)

{

System.out.println("Overall Winning Ratio : 0.00");

}

else

{

// Display the winning the ratio with 2 decimal places accuracy.

System.out.println("Overall Winning Ratio : " + (Math.round((1.0 \* this.wins) / totalRounds \* 100.0) / 100.0));

}

System.out.println("Your Current Balance : $" + (this.initialCredit));

System.out.println("Your Winnings/Losses : " + (this.credit - this.initialCredit) + "$");

System.out.println("Your Final Balance : $" + (this.credit));

// If there is a record of the history then show the record.

if (history.size() != 0)

{

System.out.println("\n--> HISTORY <--");

for (int i = 0; i < history.size(); i++)

{

String[] round = history.get(i);

System.out.println("Round " + (i + 1) + ". Result : " + round[0] + ". Amount Won/Lost: " + round[1] + "$.");

}

}

}

}

# LUCKY DIP GENERATOR

/\*\*

\* LuckyDipGenerator class generates, guess and shows the lucky number.

\*/

public class LuckyDipGenerator

{

// The variable that contains the luckyNumber

private int luckyNumber = 0;

// Method to set the lucky number between 1 - 50

public void setLuckyNumber()

{

this.luckyNumber = 1 + (int)(Math.random() \* 50);

}

// Method to check the lucky number, if the guessed number is the lucky number then return 0, if it is greater than the lucky number then return 1, else return -1.

public int guessLuckyNumber(int number)

{

if (number == luckyNumber)

{

return 0;

}

else if (number > luckyNumber)

{

return 1;

}

else

{

return -1;

}

}

// Get the lucky number

public int getLuckyNumber()

{

return this.luckyNumber;

}

// If the guessed number is within +-5 margin from the lucky number then return True else return False

public Boolean winWithInMargin(int guess)

{

if (guess <= (this.luckyNumber + 5) && guess >= (this.luckyNumber - 5))

{

return true;

}

return false;

}

}