// Kelvin Kellner

// Pick Up Sticks

// This is a game where the user and a computer will take turns picking up sticks. Whoever picks up the final stick loses.

// Ms. Cooper

// 1 March 2018

import java.util.\*; // Use the Java Utility package

public class PickUpSticks

{

public static void main(String[] args)

{

// INITIALIZATION //

// Welcome the user to the game, and teach them how to play (I only put these as separate print lines to make it easier on the eyes while coding.

System.out.println("\nWelcome to Pick Up Sticks!");

System.out.println("To play this game, you and a computer will take turns picking up sticks.");

System.out.println("There are typically 13 sticks to begin with. On each turn, you can pick up 1-3 sticks.");

System.out.println("Whoever picks up the final stick loses the game.\n");

// Runs the method to start the game. By making the entire game a separate method, I can simply call the startGame() method to start a new game, if the player chooses to do so.

startGame();

}

public static void startGame()

{

Scanner scan = new Scanner(System.in); // Create a scanner so that we can collect user input

boolean validAnswer = false; // Create a boolean that will be used to check whether or not the user's answer is valid

// DOES THE COMPUTER OR THE PLAYER START? //

boolean userFirst = false; // Create a boolean that will be used to store whether ther user or the computer will take the first turn. It is defaulted to false, so unless the program changes it to true, the answer will remain invalid.

// This will ask the user whether or not they want to play first, until they give a valid yes/no answer

do {

// Ask the user if they would like to play first

System.out.println("Would you like to take the first turn? (y/n)");

String answer = scan.nextLine();

// Run this one if the user says yes

if (answer.equalsIgnoreCase("y") || answer.equalsIgnoreCase("yes"))

{

validAnswer = true; // Tell the program not to ask the user again because their answer is valid

userFirst = true; // Make sure the program knows that the USER will take the first turn

}

// Run this one if the user says no

else if (answer.equalsIgnoreCase("n") || answer.equalsIgnoreCase("no"))

{

validAnswer = true; // Tell the program not to ask the user again because their answer is valid

userFirst = false; // Make sure the program knows that the COMPUTER will take the first turn

}

// Run this one if the user didn't properly say yes or no

else

{

System.out.println("Sorry, I didn't get that :(\nPlease make sure you entered a valid answer (y/n)\n");

}

} while (validAnswer == false); // Restart the loop if the user's answer sucks

// HOW MANY STICKS ARE WE USING? //

int sticks = 13; // This variable will keep track of the number of sticks remaining

validAnswer = false; // Reset this boolean to the default value so it can be used in the next loop as well

// This will ask the user how many sticks they want to play with

do {

// Ask the user how many sticks they would like

System.out.println("\nHow many sticks would you like to play with? (default: 13)");

// If the user enters a valid number of sticks, the program will store it, otherwise an error message will be displayed, and the player must enter a new number

try

{

// If the user does not enter an integer, it will tell the if statement below that there was an error, and a message will be displayed, otherwise it will store the users answer

sticks = scan.nextInt(); // Collect the # of sticks the player asks for

scan.nextLine(); // Clear scanner

validAnswer = true; // This will tell the computer that, so far, there are no errors

}

catch (Exception e) // If there's an error (not an integer, for example)

{

scan.nextLine(); // Clear the scanner

validAnswer = false; // Tell the program there was an error, so it knows to display a message

}

// The program will display a message if the user failed to enter a valid number of sticks

if (!validAnswer)

// If the player's answer was not an integer (therefore failing the try-catch function above), tell them that and get them to pick a new answer

System.out.println("I don't think that was an integer :(\n");

else if (sticks > 100)

{

// The program won't let the player play with more than 100 sticks

System.out.println("That's a lot of sticks, I think you'll be playing for a while...\nWhy don't you try a smaller number :)\n");

validAnswer = false; // The answer may be an integer, but it is still not valid. The player will be asked for a new answer.

}

else if (sticks < 5)

{

// The program also won't let the player play with less than 5 sticks

System.out.println("Oh, come on. You're gonna need more sticks than that...\n");

validAnswer = false; // The answer may be an integer, but it is still not valid. The player will be asked for a new answer.

}

else

{

// If the user enters a valid number of sticks, the program will move onto the next step in the program :)

System.out.println("Alright! " + sticks + " sticks it is then!\nLet's get started :)\n");

validAnswer = true; // The answer is A okay, totally valid, break the loop!

}

} while (validAnswer == false); // If the answer is invalid, ask the player for a new one

// GETTING STARTED //

// Let the game begin!

boolean gameRunning = true; // This variable will be used so the program knows whether or not there are any sticks left

boolean playerWon = false; // This variable will tell the program whether or not the player was the winner when the game is over

// PLAYING TURNS //

// If the computer starts, let it make the first move

if (!userFirst)

sticks = computersTurn(sticks); // Let the computer play its turn, and then return and store the new total # of sticks

// Keep taking turns until there are no more sticks remaining

do

{

// Let the player take their turn, and save the number of sticks when their turn is over

sticks = playersTurn(sticks);

// If there are still sticks remaining, the computer will go. Otherwise, the loop will break, and the game will end

if (sticks == 0)

{

gameRunning = false; // Ends this loop

playerWon = false; // This tells the program that the USER lost the game

System.out.println("There are no more sticks!\n");

}

else if (sticks == 1)

{

// If there is only one stick remaining on the computers turn, it will lose. However, the final turn still needs to be played first.

gameRunning = false; // Ends this loop after the computer's turn is finished

playerWon = true; // This tells the program that the COMPUTER lost the game

computersTurn(sticks); // Let the computer take the final turn of the game

}

else

{

// Let the computer play their turn casually if there are more turns to be played before anyone loses

sticks = computersTurn(sticks);

}

} while (gameRunning); // If someone loses, this loop will stop, otherwise it will let both parties continue taking turns

// When there are no sticks left over, the game is done, and the game over options are called.

gameOver(playerWon); // The game over menu will also need to know whether the computer or the player won the game

scan.close(); // Close the scanner when finished

} // Close gameRunning Method

// THE COMPUTER'S TURN //

public static int computersTurn(int sticks)

{

int sticksTaken; // The number of sticks the computer will take.

int remainder = sticks % 4; // The amount left over when breaking the sticks into groups of four, sorting into groups of four is part of how the computer rigs the game

// This if/else statement will determine the amount of sticks the computer will take:

if (sticks == 1)

// If there is only one stick remaining, the computer must take the final stick and accept defeat.

// If this section didn't exist, the computer would try to generate a random number, which may be greater than 1, which is not possible.

{

sticksTaken = 1;

}

else if (remainder != 1)

// If this condition is met, than it means that the computer is able to rig the game

{

// Take the number necessary to make a group of four, leaving a remainder of 1 and trapping the player, making certain that the computer will win

sticksTaken = remainder - 1;

// When there is no remainder, it always subtracts 1 from 0, to make -1.

// In reality, you are supposed to subtract 1 from 4, not 0.

// This statement fixes that minor bug.

if (sticksTaken == -1)

sticksTaken = 3;

}

// If the computer cannot rig the game, they will take a random number of sticks

else

{

// Generates a random number of sticks from 1-3 for the computer to take

Random random = new Random();

sticksTaken = random.nextInt(3) + 1;

}

// Update the numer of sticks remaining, and tell the user the new total.

sticks -= sticksTaken;

System.out.println("The computer picked up " + sticksTaken + " " + isPlural(sticksTaken) + "."); // the "isPlural()" method will either return the word "stick" or "sticks" depending on whether or not the number of sticks is plural

drawSticks(sticks); // Draw the sticks graphics

System.out.println("There are now " + sticks + " " + isPlural(sticks) + " remaining.\n");

return sticks; // Update the total number of sticks for the main method

} // Close computersTurn Method

// THE PLAYER'S TURN //

public static int playersTurn(int sticks)

{

Scanner scan = new Scanner(System.in); // Create a scanner for this method

int sticksTaken = 0; // This variable will be used to store how many sticks the player takes

// Ask how many sticks the user would like to pick up

System.out.println("There are currently " + sticks + " " + isPlural(sticks) + " remaining. How many sticks would you like to pick up? (1-3)");

boolean isError = false; // This boolean will be used to throw a message if the player enters something other than an integer

// This will keep asking the user for a new number of sticks if their answer is invalid

do

{

// The program will try to accept the users answer, if their answer is not an integer, it will tell the program to throw an error message and try again

try

{

// If the user enters an integer, store the value

sticksTaken = scan.nextInt();

scan.nextLine(); // Clear scanner

isError = false; // There is no error with the data type

}

catch (Exception e)

{

// If the user enters something other than an integer, tell the program that there is an error so it knows to throw a message

scan.nextLine(); // Clear scanner

isError = true; // There is an error with the data type

}

// If the user enters any invalid answer, the program will display the appropriate error message

if (isError)

// Tells the player to try again if they enter something other than an integer

System.out.println("\nSorry, that's not an integer. Try entering a number from 1-3 :)");

else if (sticksTaken > 3 || sticksTaken <= 0)

// If the player enters a number other than 1, 2, or 3, the program will ask for a new number

System.out.println("\nSorry, you can only take 1-3 sticks! Please try again :)");

else if (sticksTaken > sticks)

// Asks the player to enter a new number if they take more sticks than the number of sticks avaliable

System.out.println("\nThere aren't that many sticks left over! Please try a different number :)");

} while (sticksTaken > 3 || sticksTaken <= 0 || sticksTaken > sticks || isError);

// Update the number of sticks remaining, and tell the user the new total.

sticks -= sticksTaken;

System.out.println("You picked up " + sticksTaken + " " + isPlural(sticksTaken) + ".\n");

// I commented these few lines out.

// They used to draw the number of sticks remaining, but I figured since the player just played, and the computer's turn is directly after anyway, this visual feedback is unnecessary and confusing. It makes the game more clear if these lines are not displayed.

//drawSticks(sticks); // Draw the sticks graphics

//System.out.println("There are now " + sticks + " sticks remaining.\n");

scan.close(); // Close the scanner when finished

return sticks; // Update the total number of sticks for the main method

} // Close playersTurn Method

// ENDING THE GAME //

public static void gameOver(boolean playerWon)

{

System.out.println("Game Over!");

// Displays a different message depending on who won the game

if (playerWon)

System.out.println("CONGRATULATIONS! You were smart enough to beat the computer :)");

else

System.out.println("SORRY! The computer beat you :(");

Scanner scan = new Scanner(System.in); // Create a scanner for this method

boolean validAnswer = false; // Create a boolean that will be used to check whether or not the user's answer is valid

// This will ask the user whether or not they want to play again

do {

// Ask the user if they would like to play again

System.out.println("\nWould you like to play another round? (y/n)");

String answer = scan.nextLine();

if (answer.equalsIgnoreCase("y") || answer.equalsIgnoreCase("yes"))

{

validAnswer = true; // Tell the program not to ask the user again

System.out.println("\nAwesome!\nLet's play another :)\n\n\n\n\n\n\n\n"); // This will thank the user, and then add spacing for the new game

startGame(); // Start the new game

}

else if (answer.equalsIgnoreCase("n") || answer.equalsIgnoreCase("no"))

{

validAnswer = true; // Tell the program not to ask the user again

System.out.println("\nOkay, well thanks for playing!\nGoodbye :)\n");

}

else

{

System.out.println("Sorry, I didn't get that :(\nPlease make sure you entered a valid answer (y/n)\n");

}

} while (validAnswer == false); // Restart the loop if the user's answer is invalid

scan.close(); // Close the scanner when finished

} // Close endGame Method

// DRAWING GRAPHICS FOR THE # OF STICKS //

public static void drawSticks(int sticks)

{

// This method will create a visual display of the current number of sticks after every turn

// Create an empty string to store the row of sticks

String line = "";

// Add one line/stick segment for every stick

for (int i = sticks; i > 0; i--) // Create a local variable "i" and set the value to the number of sticks. This loop will run once for each stick there is, until the number of sticks left to draw and add to the String is 0.

line += " |"; // Adds one line segment for every stick, the whole String will be printed after the String is concatenated properly

// Print the rows of sticks. These visuals make the game more appealing

System.out.println(line);

} // Close drawSticks Method

// IS "STICK" PLURAL? //

public static String isPlural(int input)

{

// This program will decide whether or not to write the word "sticks" or "stick" in print lines depending on whether or not the program is referring to one or more stick

if (input == 1)

return "stick";

else

return "sticks";

} // Close isPlural Method

} // End class