Project 5

Kyle Guarco

Output

```
Enter a number between 1 and 100: 33
Your guess should be higher.

Enter a number between 1 and 100: 83
Your guess should be lower.

Enter a number between 1 and 100: 56
Your guess should be lower.

Enter a number between 1 and 100: 46
Your guess should be higher.

Enter a number between 1 and 100: 49
Your guess should be higher.

Enter a number between 1 and 100: 51
Your guess should be higher.

Enter a number between 1 and 100: 53
Congratulations! You got it right! The answer was 53
Would you like to play again (y/n)? n
kyleg%
```

st

```
kyleg% java -jar HiLo.jar
Some information about the number:
        Last Digit: 2

Enter a number between 1 and 100: 22
Your guess should be higher.

Enter a number between 1 and 100: 62
Your guess should be lower.

Enter a number between 1 and 100: 52
Your guess should be lower.

Enter a number between 1 and 100: 42
Congratulations! You got it right! The answer was 42
Would you like to play again (y/n)? n
kyleg%
```

st

HiLo.java

```
import java.util.InputMismatchException;
import java.util.Scanner;

/**
* HiLo: The number-randomizer guessing game!
*
* @author Kyle Guarco
*/
public class HiLo
{
    public static void main(String[] args)
    {
        // Declare the guess prompt here, it's read much nicer.
        final String GUESS_PROMPT = "\nEnter a number between 1 and " +
NumberGame.GAME_MAX + ": ";
        // The prompt for playing again
        final String AGAIN_PROMPT = "Would you like to play again (y/n)? ";
        // The error that's printed when something other than a number is
entered.
        final String ERROR_STATEM = "That's not a valid integer!";
        // Prints when the user gets the number right!
        final String CONGR_STATEM = "Congratulations! You got it right! The
answer was ";

        // Read from standard input
        Scanner stdin = new Scanner(System.in);

        while (true)
        {
            // Create a new number randomizer instance.
            NumberGame game = new NumberGame();

            // Give the initial guess a value
            int guess = NumberGame.SKIP_INT;

            // Play the game!
            game.printHint();
            do {
                // Tells the user whether their guess should be higher or lower.
                game.printOrder(guess);

                System.out.print(GUESS_PROMPT);

                try {
                    guess = stdin.nextInt();
                } catch (InputMismatchException e) {
                    System.out.println(ERROR_STATEM);
                    // nextInt() doesn't skip over invalid input, so invoke
next()
                    stdin.next();
                    continue;
                }
```

```
                // Quit the program silently if the guess is -1
                if (guess == NumberGame.SKIP_INT) System.exit(0);
            } while (!game.isCorrect(guess));

            // Congratulate the user
            System.out.println(CONGR_STATEM + guess);

            // Ask the user to play again?
            System.out.print(AGAIN_PROMPT);
            // If 'n' is typed, quit the game.
            if (stdin.next().equalsIgnoreCase("n"))
                break;
        }

        // Free system resources
        stdin.close();
    }
}
```

NumberGame.java

```
import java.util.Random;

/**
* Functions that assist the user in determining the right number.
*
* @author Kyle Guarco
*/
public class NumberGame
{
    /** The guessing range limit (nextInt limit) */
    public static final int GAME_MAX = 100;
    /** A special number that allows the program to skip over different prints.
*/
    public static final int SKIP_INT = -1;

    private Random rand;
    private NumberUtils utils;

    public NumberGame()
    {
        this.rand = new Random();
        // This will be the number that the user must guess.
        int number = rand.nextInt(NumberGame.GAME_MAX) + 1;

        // Contains the functions for retrieving information on the number.
        // Refer to Project 4 for the template.
        this.utils = new NumberUtils(number);
    }

    /** Prints out some information about the number. */
    public void printHint()
    {
        // It's like flipping a coin. Do we want to tell the user
        // the oddity of the number, or it's last digit?
        boolean chanceHint = rand.nextBoolean();

        String hint = "Last Digit: " + utils.getLastDigit();

        if (!chanceHint) {
            String oddEven = utils.isOdd() ? "Odd" : "Even";
            hint = "Oddity: " + oddEven;
        }

        System.out.println("Some information about the number:\n\t" + hint);
    }

    /** Tells the user whether they're closer to the answer or not. */
    public void printOrder(int guess)
    {
        // Don't print if the guess is -1
        if (guess == NumberGame.SKIP_INT) return;

        // Since we're checking to see if the guess is the right one
```

```
        // before calling this function, we only need to check two cases.
        int order = utils.compare(guess);

        // Is it higher (1), or lower (-1)?
        String height = order == 1 ? "higher" : "lower";

        System.out.println("Your guess should be " + height + ".");
    }

    /** Checks to see if the guess is correct (compare(guess) == 0) */
    public boolean isCorrect(int guess)
    {
        return utils.compare(guess) == 0;
    }
}
```

NumberUtils.java

```java
/**
* Functions that play with numbers.
*
* @author Kyle Guarco
*/
public class NumberUtils
{
    private int number;

    public NumberUtils(int number)
    {
        this.number = number;
    }

    /**
    * Compares the value of this number with the passed number.
    * @param n The other number
    * @return {@code -1 || 0 || 1}, indicating the inequality result
    */
    public int compare(int n)
    {
        if (number < n)
            return -1;
        else if (number > n)
            return 1;

        return 0;
    }

    public boolean isOdd()
    {
        return number % 2 == 1;
    }

    public boolean isEven()
    {
        return !isOdd();
    }

    public int getLastDigit()
    {
        // Lecture: If 100s, (n % 10) or (n - int(n / 10) * 10)
        return number % 10;
    }

    public int getNumber()
    {
        return number;
    }
}
```