UNIVERSIDAD SANTO TOMÁS

PRIMER CLAUSTRO UNIVERSITARIO DE COLOMBIA

SECCIONAL TUNJA

VIGILADA MINEDUCACIÓN - SNIES 1732

Acreditación Institucional Internacional

OTORGADA POR EL IAC CINDA ACUERDO 55 DEL 8 DE MAYO-VIGENCIA 5 AÑOS

ACREDITACIÓN INSTITUCIONAL DE ALTA CALIDAD MULTICAMPUS

Res. MEN No. 03455 del 23 de enero de 2020

Vigencia por seis años

QS STARS RATED FOR EXCELLENCE

ISO 9001 Icontec SC4259-1

CERTIFIED IQNet MANAGEMENT SYSTEM

UNIVERSIDAD SANTO TOMÁS
PRIMER CLAUSTRO UNIVERSITARIO DE COLOMBIA
SECCIONAL TUNJA
VIGILADA MINEDUCACIÓN - SNIES 1732

Faculty: Systems engineer
Course: Introduction of Programming
Topic: **Methods**

Socializer: Luis Fernando Castellanos Guarín
Karen Daniela Cuervo Cely
Email: Luis.castellanosg@usantoto.edu.co
Karen.cuervo@usantoto.edu.co
Phone: 3214582098
3105856930

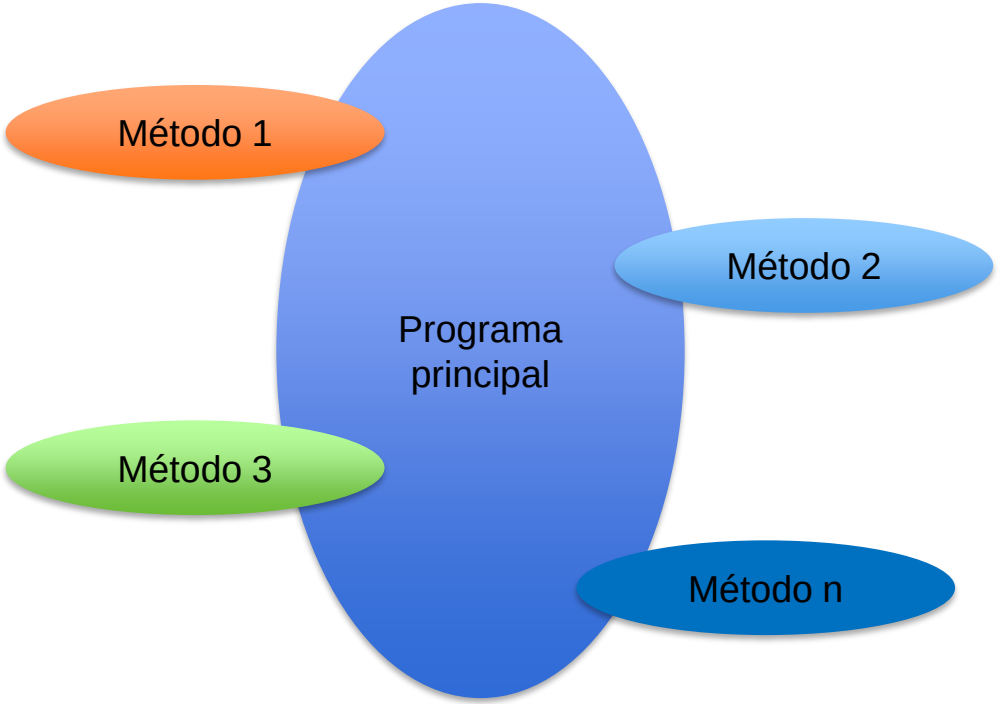
Topics

Metodos – funciones / void

- conceptos básicos / Concept basic
- Sintaxis / Syntax
- Ejemplos / examples
- Ejercicios / Exercises

¡Siempre hacia lo alto!

Methods / functions / void



Concepts basics

Qué son

- Componente de una clase que abarca definiciones de datos locales e instrucciones propias del algoritmo propuesto para la solucion del problema ó parte de éste.

Objetivo

- Ejecutar una tarea especifica cada vez que se la llama por su indentificador.

Formato

- Modificadores Tipo_de_dato_devuelto Nombre_métododo(lista de parámetros)

Tipos

- Métodos din parámetros ni valor devuelto(sin E/S)
- Métodos con parámetro de entrada y sin valor devuelto
- Métodos sin parámetros de entrada y con valor devuelto
- Métodos con paármetros de entrada y valor devuelto(puros)

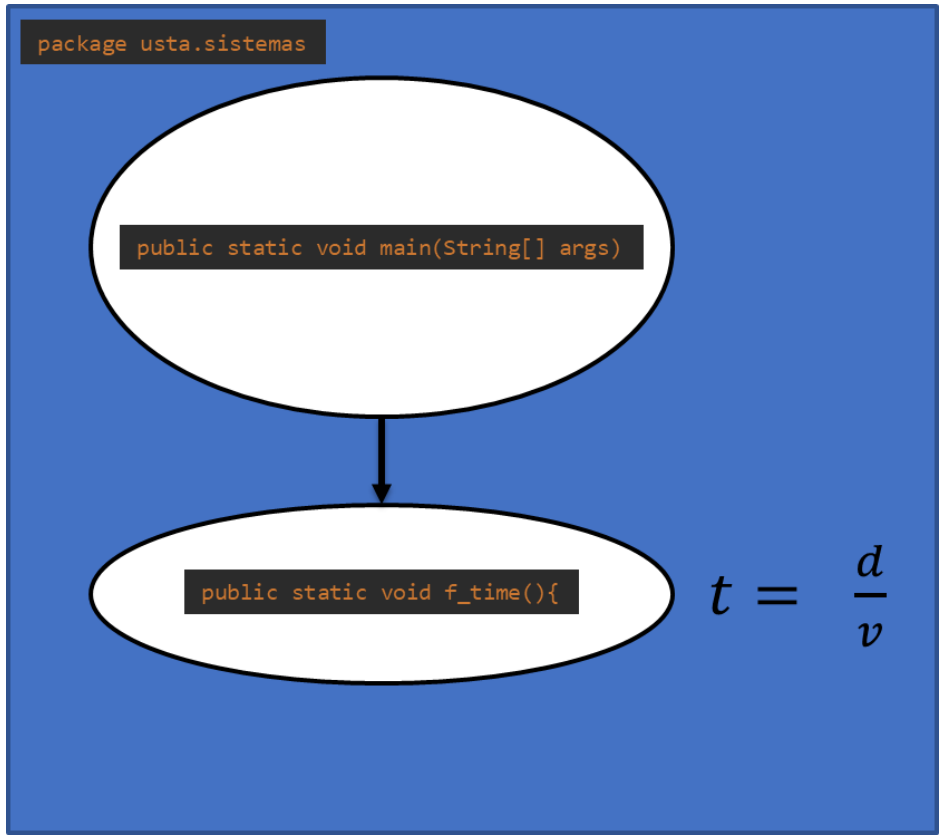
P1T41: Function that returns time

It is required to determine the time it takes a person to get from one city to another by bicycle, considering that it has a constant speed and distance (which is received as a parameter in a **function**).

Se requiere determinar el tiempo que tarda una persona en llegar de una ciudad a otra en bicicleta, considerando que lleva una velocidad constante y la distancia (la cual se recibe como parámetro en una función).

P1T41: Function that returns time

It is required to determine the time it takes a person to get from one city to another by bicycle, considering that it has a constant speed (which is received as a parameter in a **function**).



P1T42: Saved money

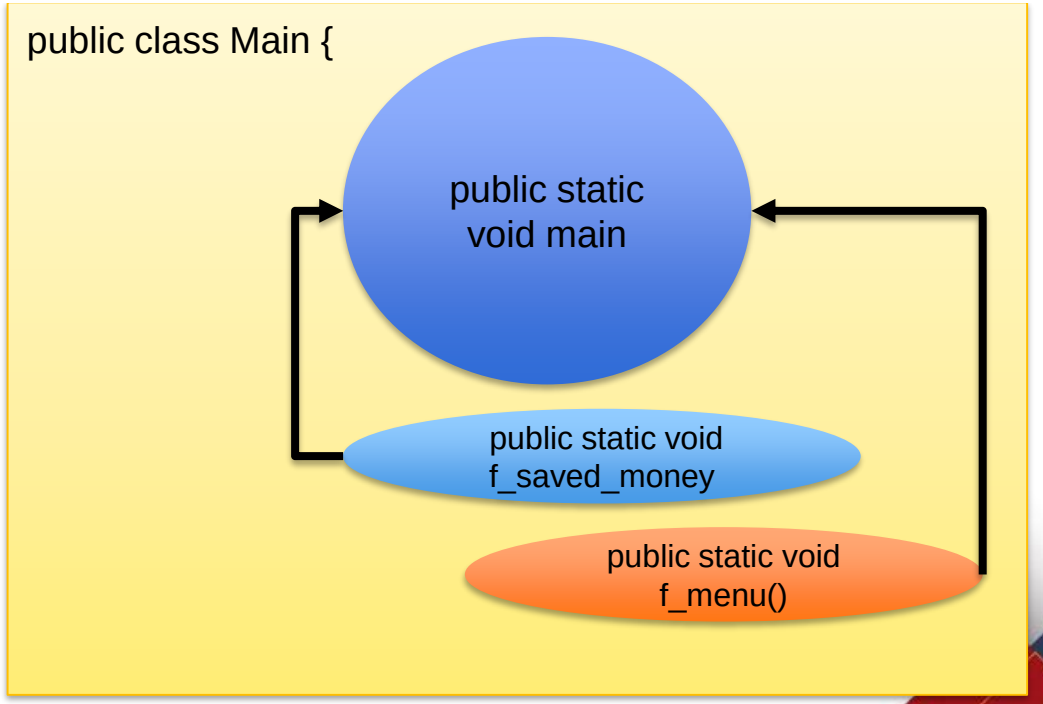
Run a program that determines how much money a person saves in a year if you consider that each week you save 15% of your salary (consider four weeks per month and that the salary does not change). Build a **function** for that purpose.

Realice un programa que determine cuánto dinero ahorra una persona en un año si considera que cada semana ahorra 15% de su sueldo (considere cuatro semanas por mes y que no cambia el sueldo). Construya una función para tal fin.

P1T42: Saved money

Run a program that determines how much money a person saves in a year if you consider that each week you save 15% of your salary (consider four weeks per month and that the salary does not change). Build a **function** for that purpose.

Realice un programa que determine cuánto dinero ahorra una persona en un año si considera que cada semana ahorra 15% de su sueldo (considere cuatro semanas por mes y que no cambia el sueldo). Construya una función para tal fin.



P1T43: Parcel service

An international parcel company has service in some countries in North America, Central America, South America, Europe and Asia. The cost for the parcel service is based on the weight of the package and the area to which it is addressed. The above is shown in the following information:

Zone Location Cost / gram:

- 1 North America $ 11.00
- 2 Central America $ 10.00
- 3 South America $ 12.00
- 4 Europe $ 24.00
- 5 Asia $ 27.00

Determine the shipping cost of 5 different packages (values determined by you).

Una compañía de paquetería internacional tiene servicio en algunos países de América del Norte, América Central, América del Sur, Europa y Asia. El costo por el servicio de paquetería se basa en el peso del paquete y la zona a la que va dirigido. Lo anterior se muestra en la siguiente información:

Zona Ubicación Costo/gramo:

- 1 América del Norte $11.00
- 2 América Central $10.00
- 3 América del Sur $12.00
- 4 Europa $24.00
- 5 Asia $27.00

Determine el costo del envío de 10 paquetes diferentes (valores determinados por ud)

P1T44: Different conversions

Build a program that uses different methods to:

- o Convert from ºC to ºF use the formula: ºF = ºC x 1.8 + 32.
- o Convert from ºF to ºC use the formula: ºC = (ºF-32) ÷ 1.8.
- o Convert from K to ºC use the formula: ºC = K - 273.15.
- o Convert from ºC to K use the formula: K = ºC + 273.15.

Construir un programa que utilice métodos diferentes para:

- o Convertir de ºC a ºF use la fórmula: ºF = ºC x 1.8 + 32.
- o Convertir de ºF a ºC use la fórmula: ºC = (ºF-32) ÷ 1.8.
- o Convertir de K a ºC use la fórmula: ºC = K – 273.15.
- o Convertir de ºC a K use la fórmula: K = ºC + 273.15.

P1T45: Average height

Build a JAVA program that calculates the average height of a group of 5 children.

Construir un programa en JAVA que, calcule la altura promedio de un grupo de 5 niños.

3° Session CodeGym

¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 0 / Nivel 2 – Lección 0

Has terminado el nivel 1

Sintaxis de Java
Level 2, Lesson 0

AVAILABLE

¡Que todo el mundo lo sepa!

¡Hola! Enhorabuena por haber completado este nivel. Esperamos que estés disfrutando con nuestra filosofía de aprendizaje de la programación en Java.

Esta es la última lección traducida actualmente a tu idioma. Queremos que todo sea perfecto, así que estamos dedicando algún tiempo a hacer mejoras. No te preocupes, tienes dos opciones:

1. Puedes continuar tus estudios en inglés. Sin duda necesitarás conocimientos de inglés en tu futura carrera como programador :)
2. No obstante, si no te sientes cómodo estudiando en otro idioma, podemos avisarte cuando lancemos el curso en tu idioma.

Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 1 / Nivel 2 – Lección 1

Object interaction

Sintaxis de Java
Level 2, Lesson 1

AVAILABLE

"Hi, Amigo. Today I'll tell you about a typical Java program. The big news is that every program written in Java consists of classes and objects."

"I already know what classes are. What are objects?"

"Let's start with an analogy. Suppose you want to build a small ship. You work on a design and then send the blueprint to a factory, where a ship will be assembled according to your design. Or a dozen ships, or as many ships as you want. My point is that dozens of identical ships can be made based on one blueprint."

Gamified platform work / Trabajo en plataforma gamificada.

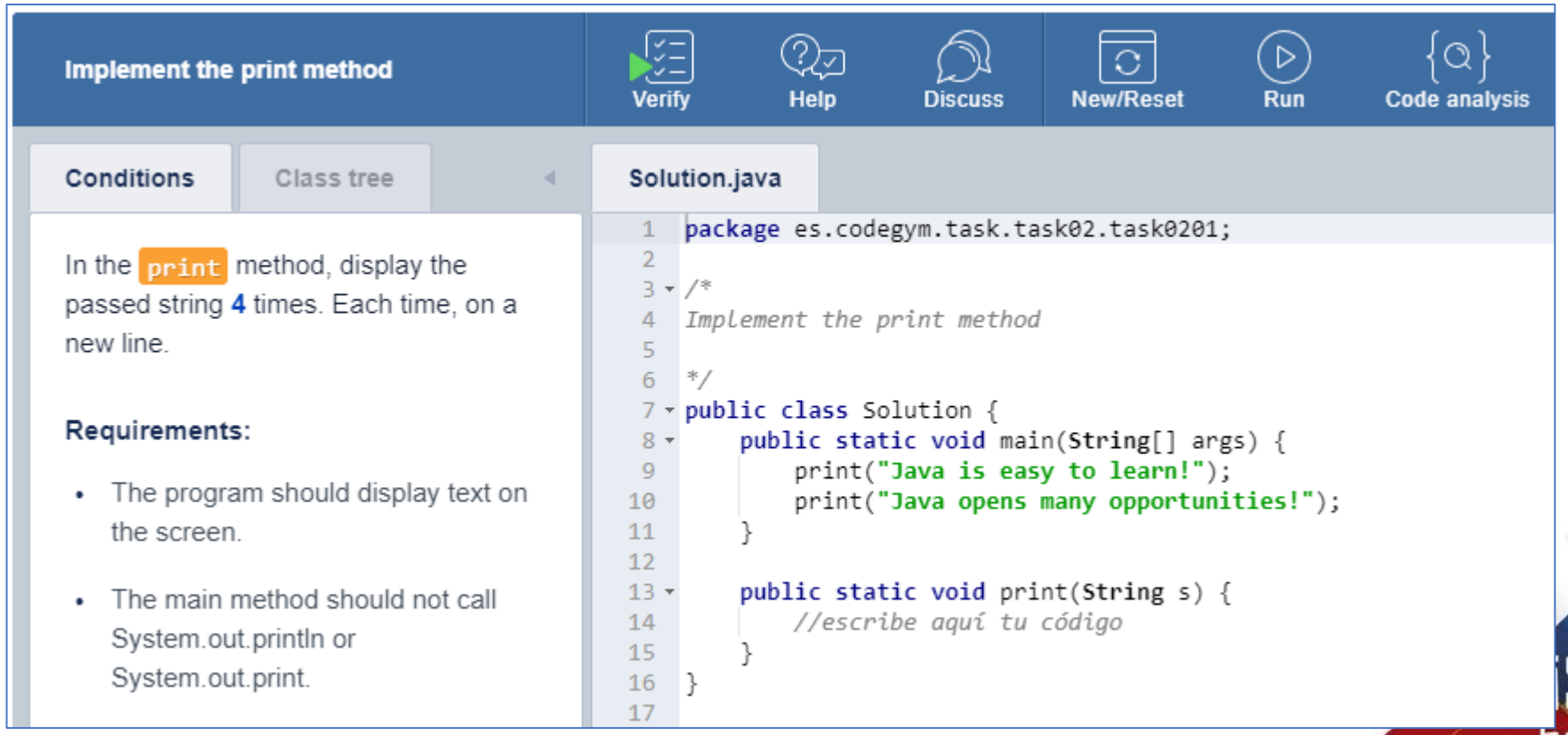
Level 2 - Lesson 1 / Nivel 2 – Lección 1

```java
package com.codegym.lesson2;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.IOException;

public class FileCopy
{
    public static void main(String[] args) throws IOException
    {
        FileInputStream fileInputStream = new FileInputStream("c:\data.txt");
        FileOutputStream fileOutputStream = new FileOutputStream("c:\result.txt");

        while (fileInputStream.available() > 0)
        {
            int data = fileInputStream.read();
            fileOutputStream.write(data);
        }

        fileInputStream.close();
        fileOutputStream.close();
    }
}
```

Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 1 / Nivel 2 – Lección 1

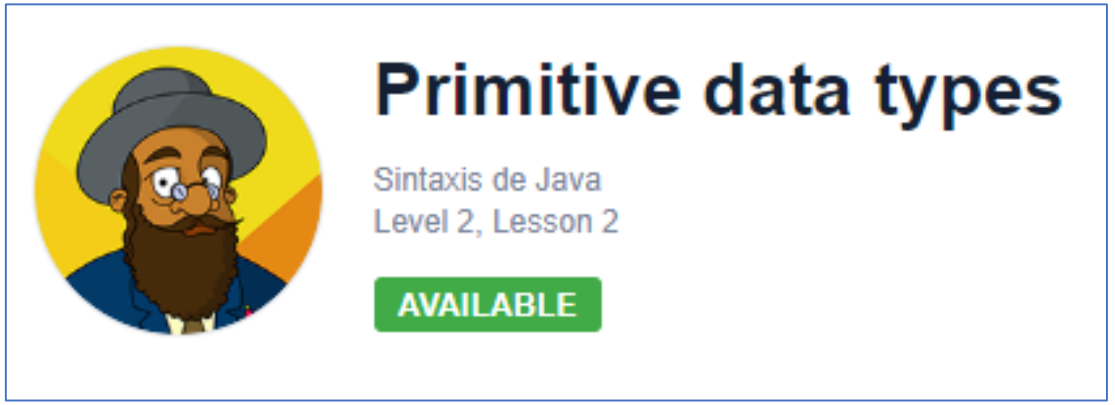
Excercise: **N2_L1_E1**



¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 2 / Nivel 2 – Lección 2



¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 2 / Nivel 2 – Lección 2

Java code	Description

`public class Person`
`{`
`  String name;`
`  int age;`
`}`
	A new composite type is declared – `Person`.
	Its data is stored in the `String` (composite type) variable name and (primitive type) `int` variable `age`

`public class Rectangle`
`{`
`  int x, y, width, height;`
`}`
	A new composite type is declared – `Rectangle`.
	It consists of four `int` (primitive type) variables.

`public class Cat`
`{`
`  Person owner;`
`  Rectangle territory;`
`  int age;`
`  String name;`
`}`
	A new composite type is declared – `Cat`. It has the following variables:
	— `owner`, composite type `Person`
	— `territory`, composite type `Rectangle`
	— `age`, primitive type `int`
	— `name`, composite type `String`

Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 2 / Nivel 2 – Lección 2

Java code	Description
`String s;` `String s = null;`	Equivalent statements.

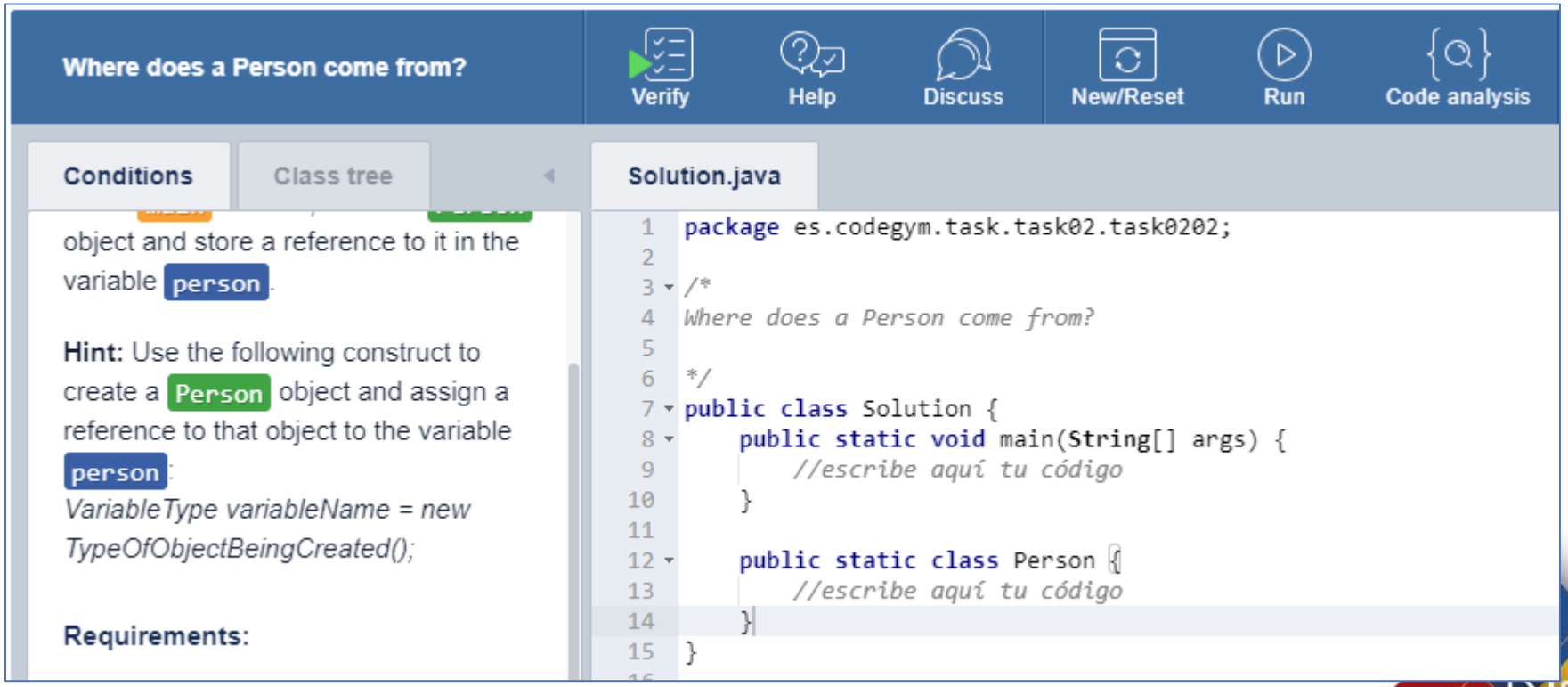
`Person person;`
`person = new Person();`
`person = null;`
	We create a person variable whose value is null.
	We assign to it the address of a newly created Person object.
	We assign null to the variable.

`Cat cat = new Cat();`
`cat.owner = new Person();`
`cat.owner.name = "God";`
	We create a Cat object and store its address in variable cat; cat.owner equals null.
	We set cat.owner equal to the address of a newly created Person object.
	cat.owner.name still equals null.
	We set cat.owner.name equal to "God"

Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 2 / Nivel 2 – Lección 2

Excercise: **N2_L2_E1**

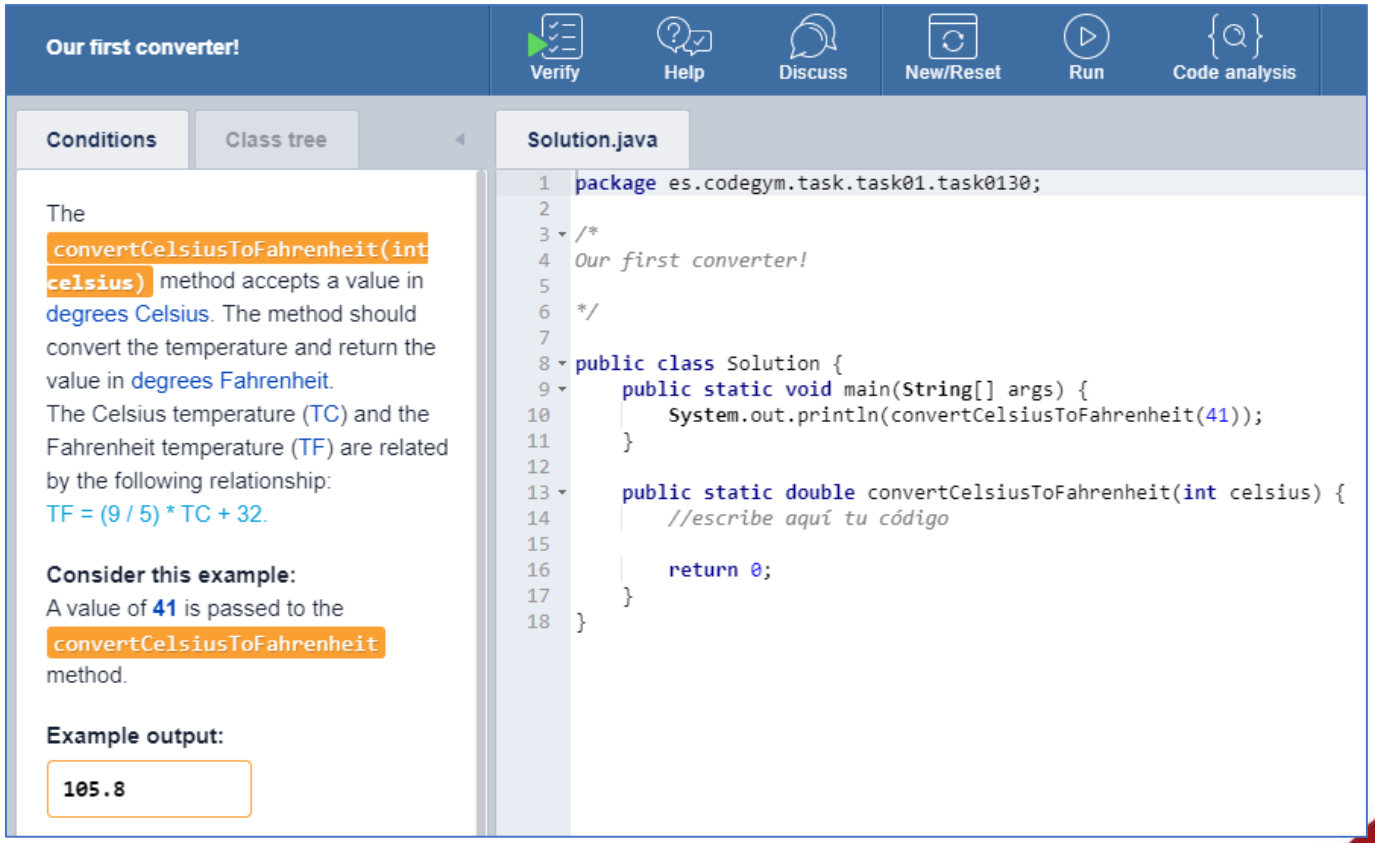


¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 2 / Nivel 2 – Lección 2

Excercise: **N2_L2_E2**

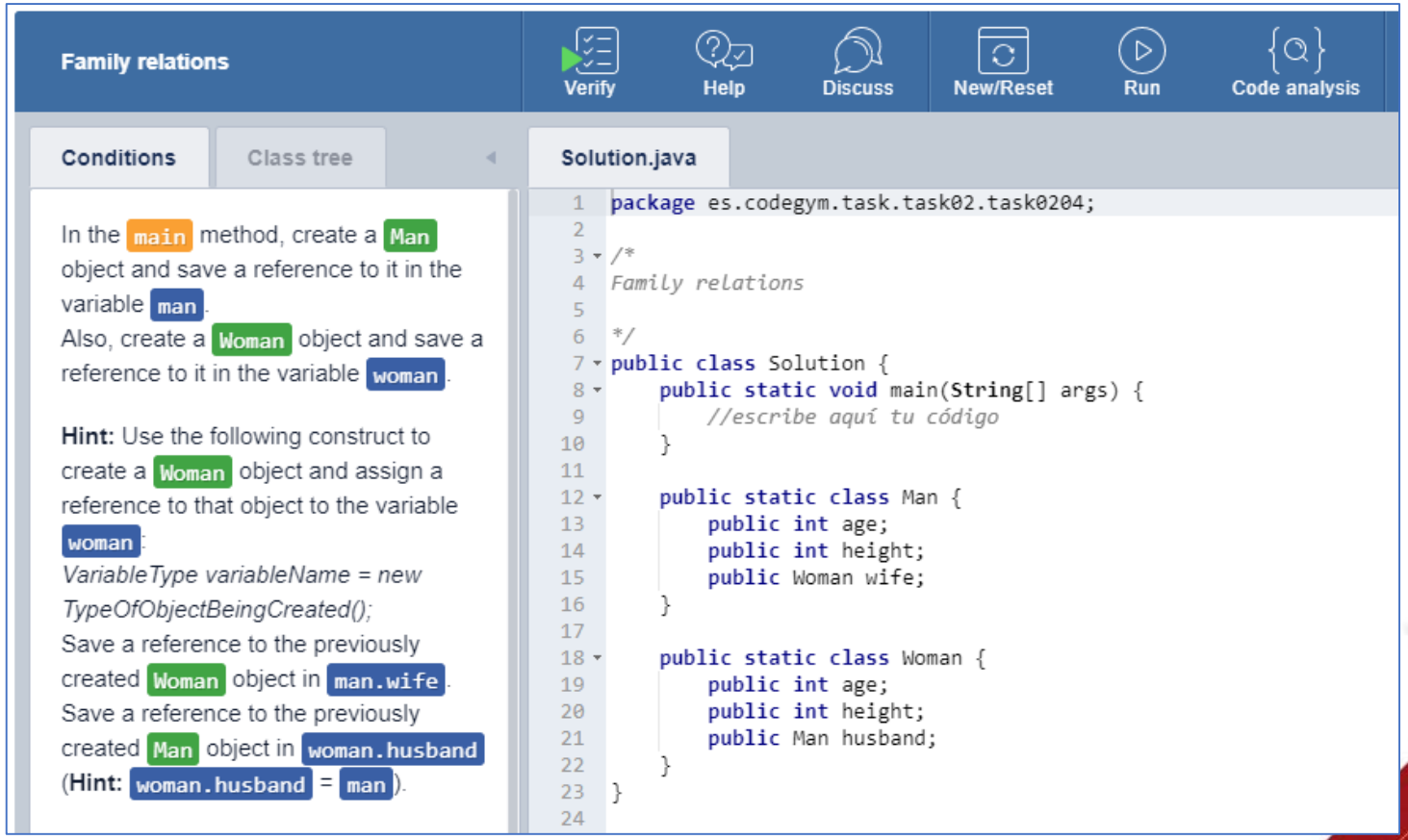


¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 2 / Nivel 2 – Lección 2

Excercise: **N2_L2_E3**

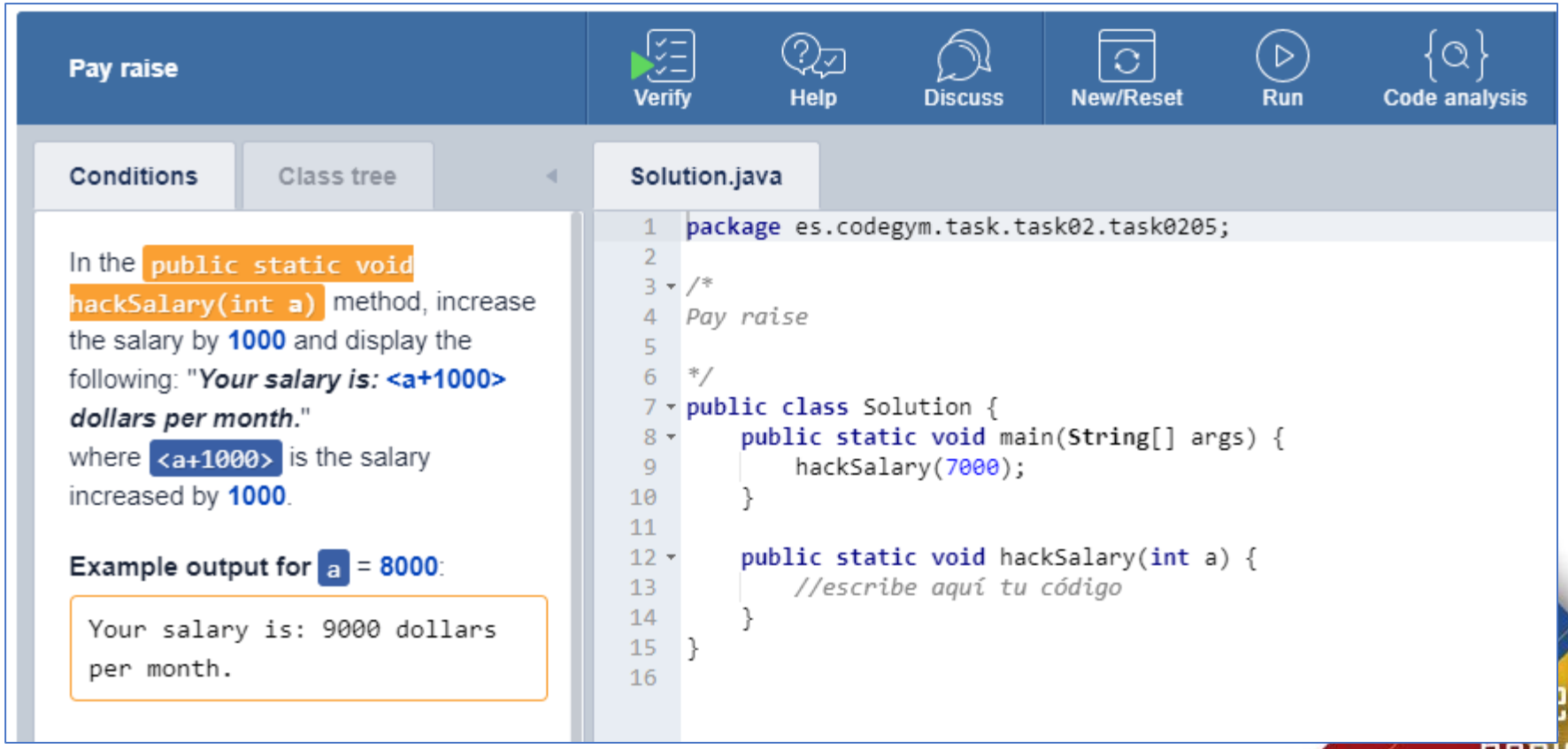


¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 2 / Nivel 2 – Lección 2

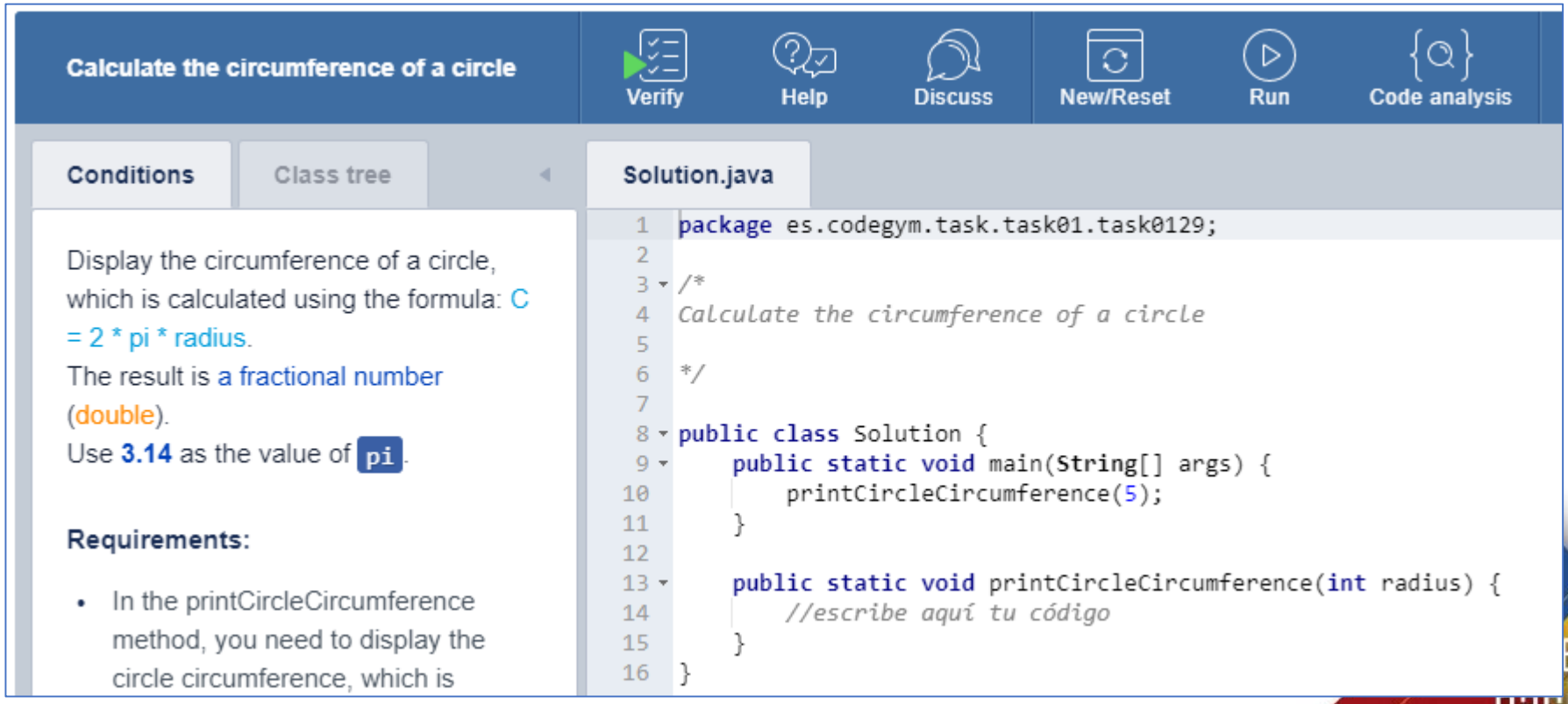
Excercise: **N2_L2_E4**



Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 2 / Nivel 2 – Lección 2

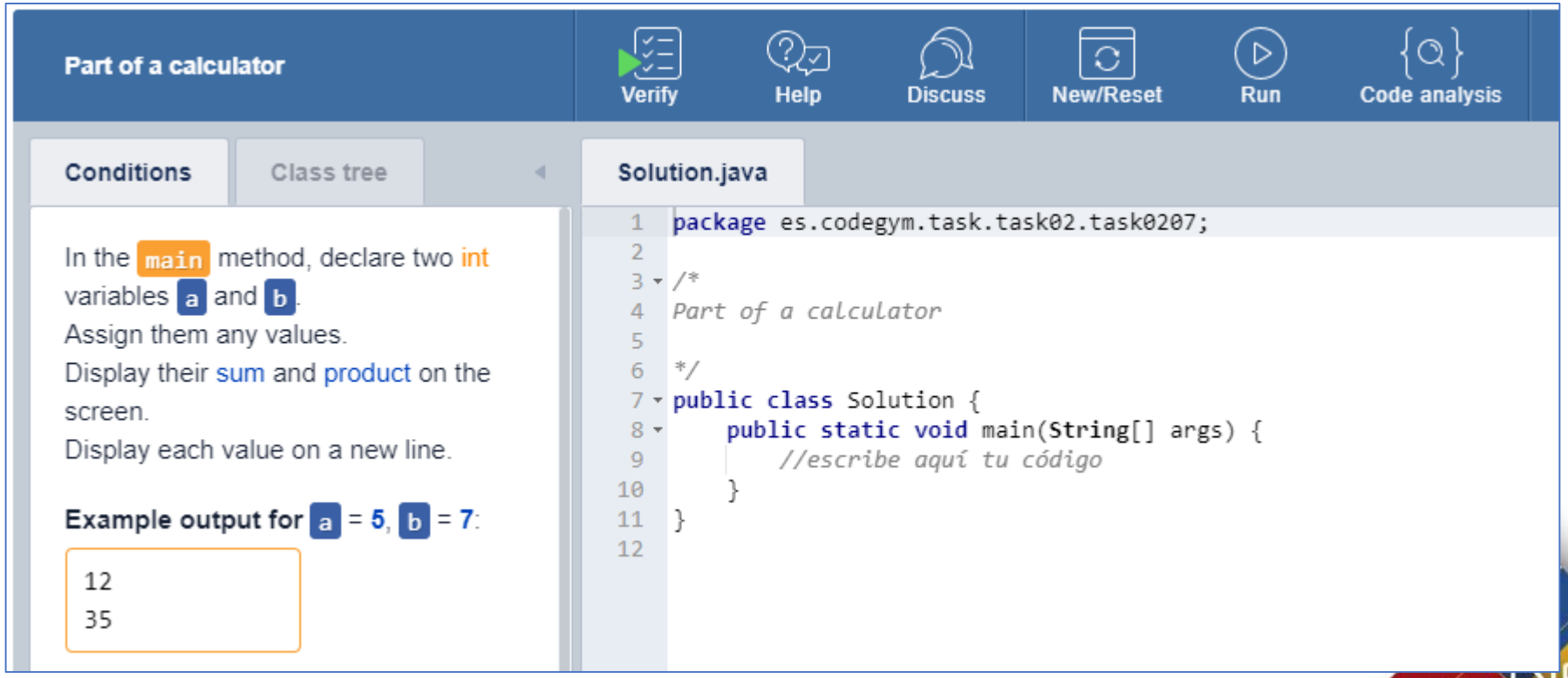
Excercise: **N2_L2_E5**



Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 2 / Nivel 2 – Lección 2

Excercise: **N2_L2_E6**



¡siempre hacia lo alto!

To practice!

¡Siempre hacia lo alto!

P1T45: Methods

Create an application that calculates the area of a circle, square or triangle. The user will be asked to enter the figure to which they want to calculate their area. As entered you will ask for the necessary values to calculate the area. Create a method for each figure to calculate each area, it will return a real number. Display the result on screen. Here we show you what each figure needs:

- Circle: (radius ^ 2) * PI
- Triangle: (base * height) / 2
- Square: side * side

Crea una aplicación que calcule el área de un círculo, cuadrado o triangulo. Se solicitará que el usuario ingrese la figura a la que quiere calcular su área. Según lo introducido pedirá los valores necesarios para calcular el área. Crea un método por cada figura para calcular cada área, este devolverá un número real. Muestra el resultado por pantalla. Aquí te mostramos que necesita cada figura: Circulo: (radio^2)*PI
Triangulo: (base * altura) / 2
Cuadrado: lado * lado

P1T46: Compare Numbers

Create a program that asks for two numbers and says if they are the same or not. The process must be carried out in a method called compareNumbers(), which receives as parameters the numbers

Crear un programa que pida dos números y diga si son iguales o no. Se debe realizar el proceso en un método que se llame comparaNumeros(), el cual recibe como parámetros los números

P1T47: Multiples

Create a program that asks for two numbers and says if one is a multiple of the other.

Crear un programa que pida dos números y diga si uno es múltiplo del otro.

P1T48: Capicúa

Create a program that asks for a number between 0 and 9,999, say if it is capicúa. Create a function to define whether or not it is capicúa.

Crear un programa que pida un número entre 0 y 9.999, decir si es capicúa. Crear una función para que defina si es o no capicúa.

P1T49: Methods with Time

Write two functions that allow you to calculate:

o The number of seconds in a given time in hours, minutes and seconds.

o The number of hours, minutes and seconds of a given time in seconds.

Escribir dos funciones que permitan calcular:

o La cantidad de segundos en un tiempo dado en horas, minutos y segundos.

o La cantidad de horas, minutos y segundos de un tiempo dado en segundos.

4° Session CodeGym

¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 3 / Nivel 2 – Lección 3

¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 3 / Nivel 2 – Lección 3

Code	Description
`Cat cat;`	Declares a Cat reference variable named cat. The variable cat's value is null.
`new Cat();`	Creates a Cat object.
`Cat cat = new Cat();`	Creates a Cat reference variable named cat. Creates a new Cat object. Assigns a reference to the newly created object to the variable cat.
`Cat kitty = new Cat();` `Cat smokey = new Cat();`	Two objects are created. References to them are assigned to two different variables.

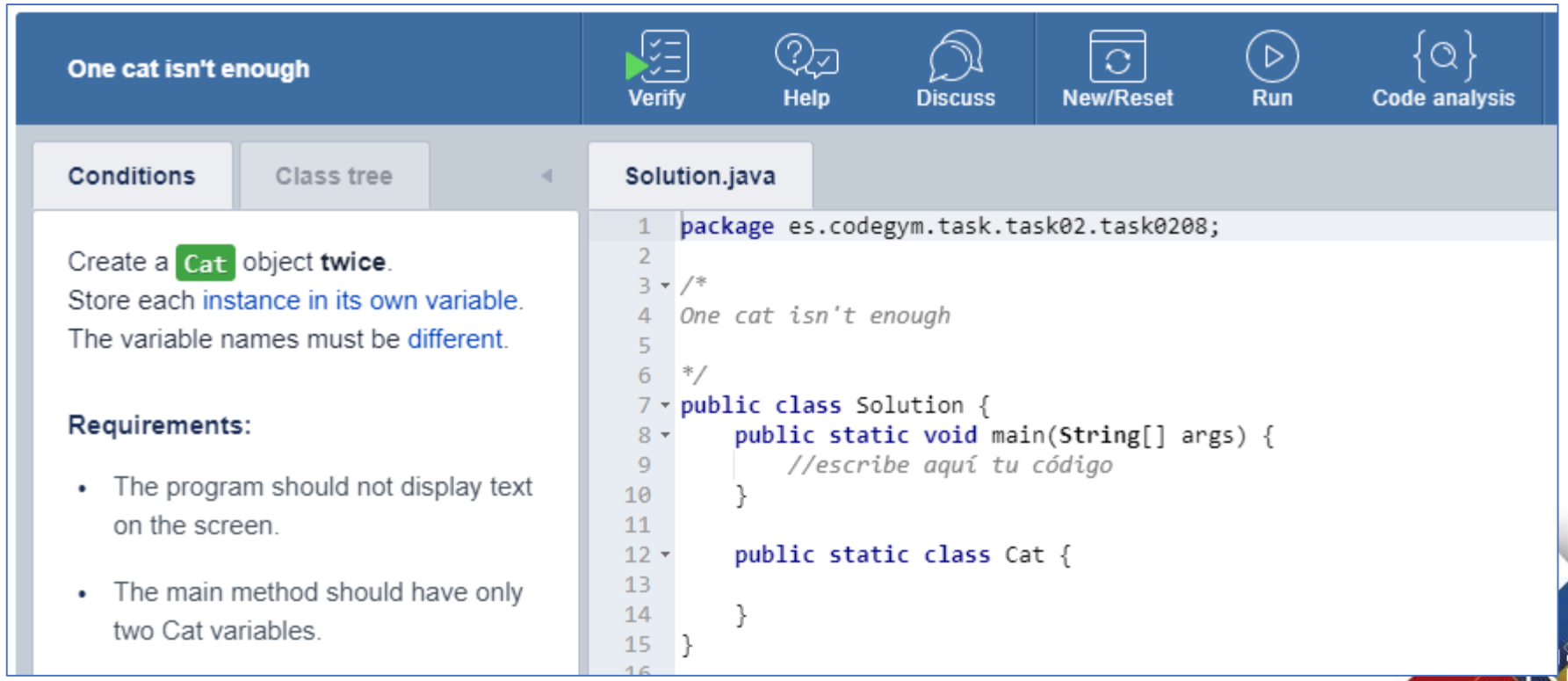
`Cat kitty = new Cat();`
`Cat smokey = new Cat();`
 	
`smokey = kitty;`
	Two objects are created. References to them are assigned to two different variables.
	
	Then we set the variable smokey equal to a reference to the object referenced by the variable kitty. Both variables now refer to the first created objects.
	(Because the second object is no longer referenced anywhere, it is now considered garbage)

`Cat kitty = new Cat();` `Cat smokey = null;`	One Cat object is created, and a reference to it is assigned to the first variable (kitty). The second variable (smokey) stores an empty (null) reference.

Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 3 / Nivel 2 – Lección 3

Excercise: **N2_L3_E1**

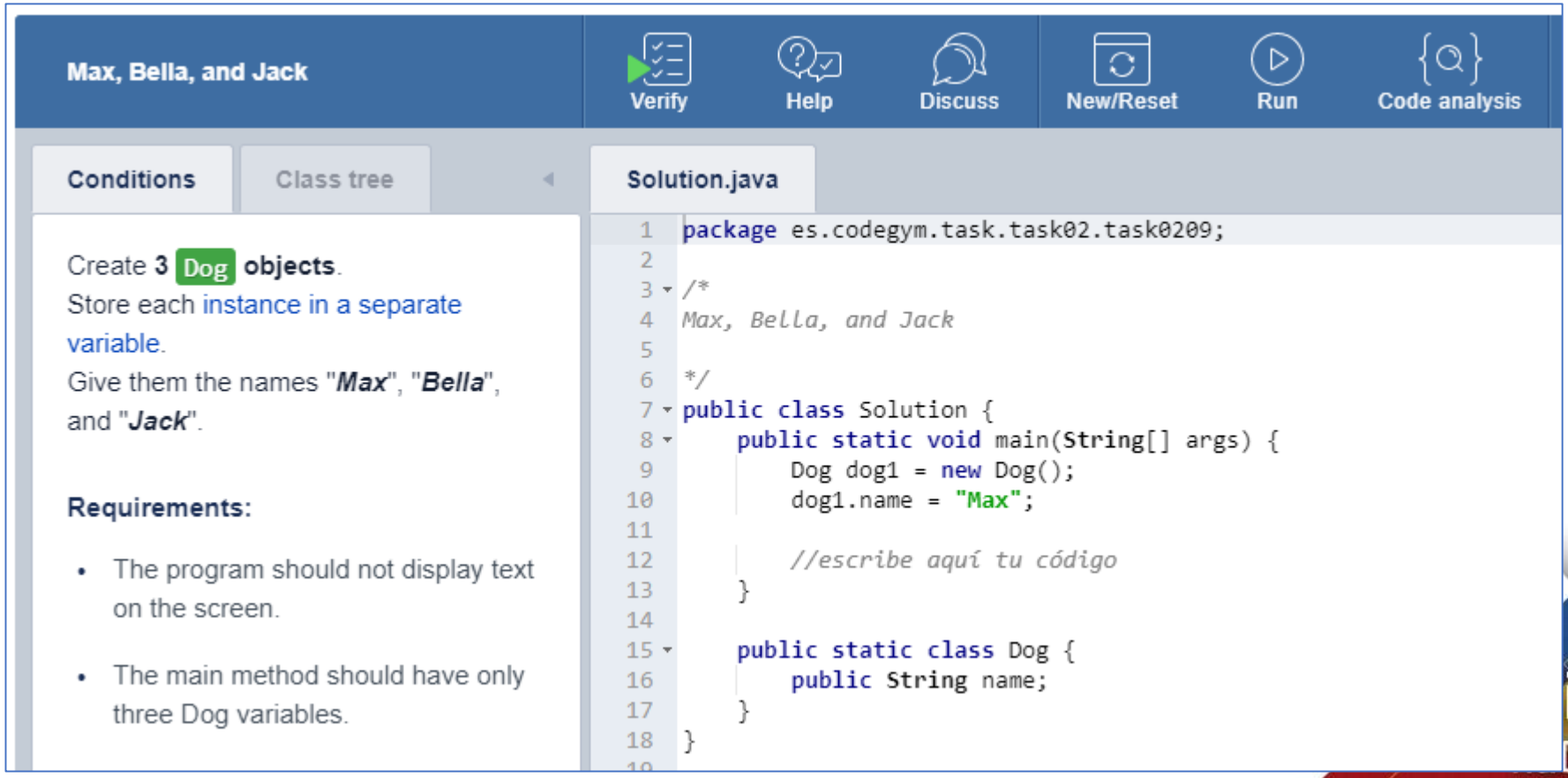


¡siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 3 / Nivel 2 – Lección 3

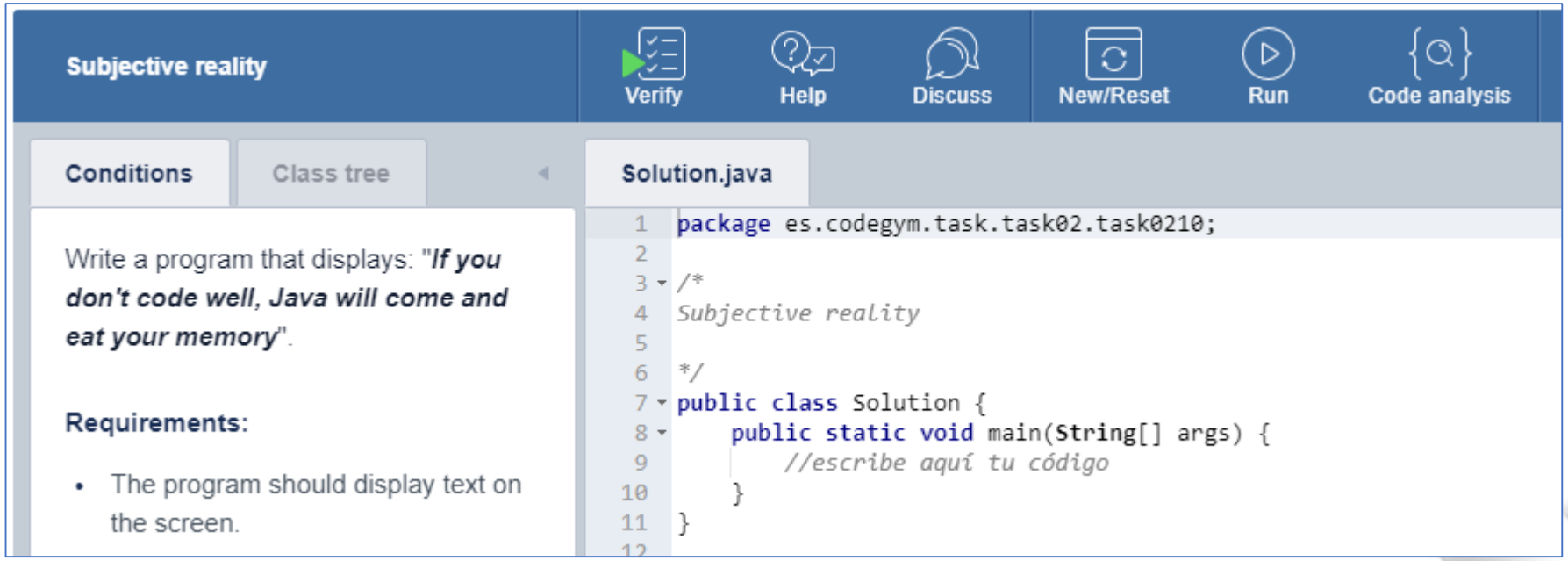
Excercise: **N2_L3_E2**



Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 3 / Nivel 2 – Lección 3

Excercise: **N2_L3_E3**



¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 4 / Nivel 2 – Lección 4

Visibility of variables

Sintaxis de Java
Level 2, Lesson 4

AVAILABLE

"Hello to my favorite student. Now I'm going to tell you about the visibility of variables."

"Huh? Can variables be invisible?"

"No. A variable's 'visibility', or scope, means the places in the code where you can refer to that variable. You may use some variables everywhere in the program, but others can only be used within their class, and still others – only within one method."

"For example, you can't use a variable before it has been declared."

"That makes sense."

"Here are a couple of examples:"

Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 4 / Nivel 2 – Lección 4

```java
public class Variables
{
   private static String TEXT = "The end.";
   public static void main (String[] args)
   {
      System.out.println("Hi");
      String s = "Hi!";
      System.out.println(s);
      if (args != NULL)
      {
         String s2 = s;
         System.out.println(s2);
      }
      Variables variables = new Variables();
      System.out.println(variables.instanceVariable);
      System.out.println(TEXT);
   }
   public String instanceVariable;
   public Variables()
   {
      instanceVariable = "Instance variable test.";
   }
}
```

¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 4 / Nivel 2 – Lección 4

Modifiers	Access from...		
	Own class	Own package	Any class
private	Yes	No	No
No modifier (**package-private**)	Yes	Yes	No
public	Yes	Yes	Yes

Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 5 / Nivel 2 – Lección 5

Different ways to create variables

Sintaxis de Java
Level 2, Lesson 5

AVAILABLE

"Hi, buddy. I made a copy of your contract for you, just in case. Rishi, that cheapskate, is blissfully ignorant. You should see the figures in my contract. Ha!"

"Good job, Diego. I think I'll learn a lot from you."

"Sure thing, Amigo. There are too many stupid people in the world who want to get rich without actually doing something. But there are even more idiots who are ready to work for free."

"OK, let's get back to our lesson. Now I'm going to teach you several ways to create variables:"

Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 5 / Nivel 2 – Lección 5

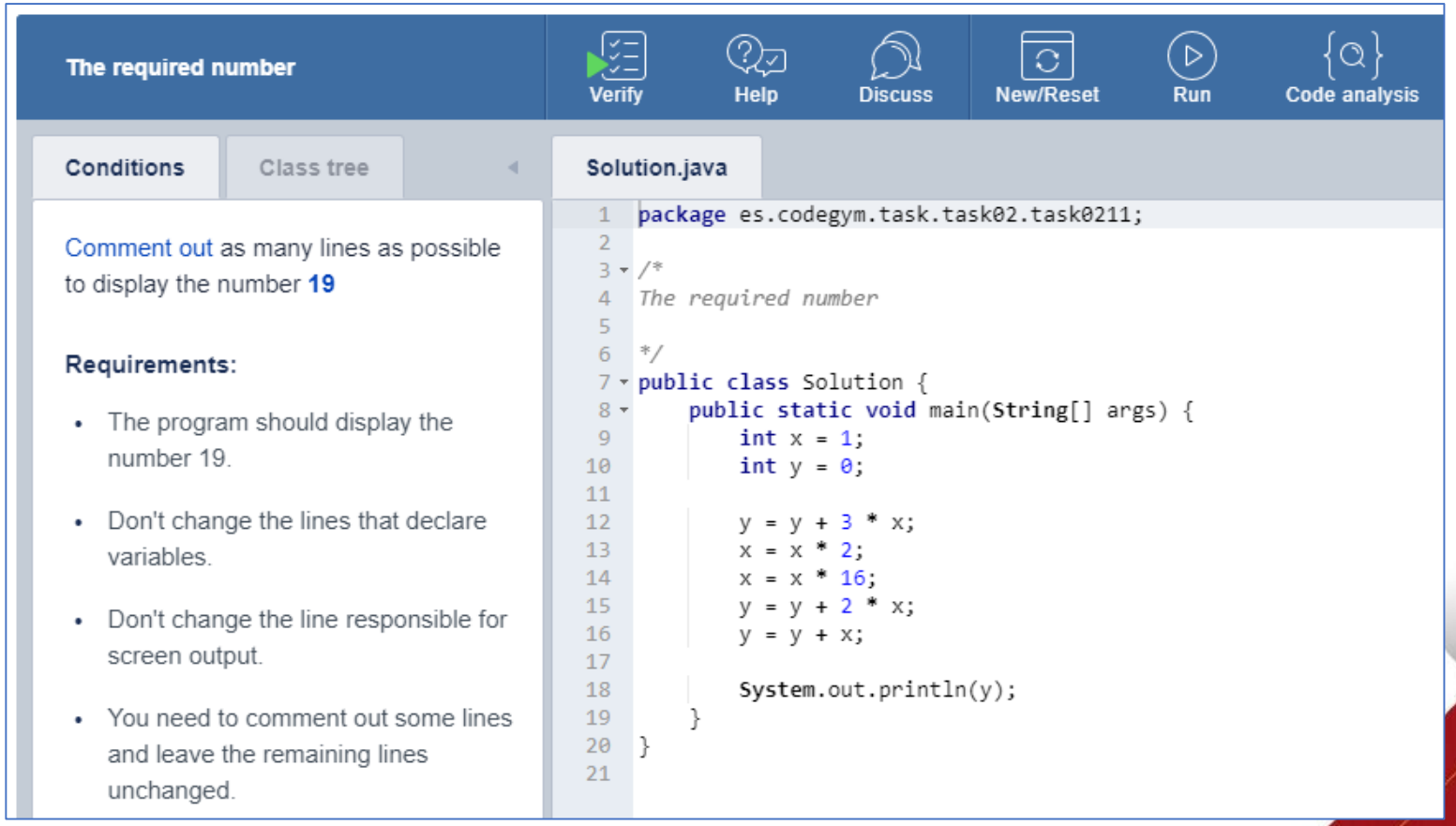
Example	Explanation
`String s1 = new String();` `String s2 = "";`	Create two identical empty strings.
`int a;`	Create an `int` variable;
`int a = 5;`	Create an `int`, variable named a and set its value equal to `5`
`int a = 5, b = 6;`	Create an `int`, variable named `a` and set its value equal to `5` Create an `int`, variable named `b` and set its value equal to `6`
`int a = 5, b = a + 1;`	Create an `int` variable named `a` and set its value equal to `5` Create an `int` variable named `b` and set its value equal to `6`
`Date date = new Date();`	Create a Date object. It is initialized to the current date and time.
`boolean isTrue = true;`	Initialize a `boolean` variable to `true`
`boolean isLess = (5 > 6);`	Assign `false` to the `isLess` variable. `Boolean` variables only accept the values true and false.

Siempre acia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 5 / Nivel 2 – Lección 5

Excercise: **N2_L5_E1**



¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 5 / Nivel 2 – Lección 5

Excercise: **N2_L5_E2**

Crazy eights

Verify | Help | Discuss | New/Reset | Run | Code analysis

Conditions | Class tree

Create **10 Cat variables** and **8 Cat objects**.

Requirements:

- The program should not display text on the screen.
- The main method must have 10 Cat variables.
- 8 variables must be initialized in the main method.
- There should be no variables in the Cat class.
- There should be no methods in the Cat class.

Solution.java

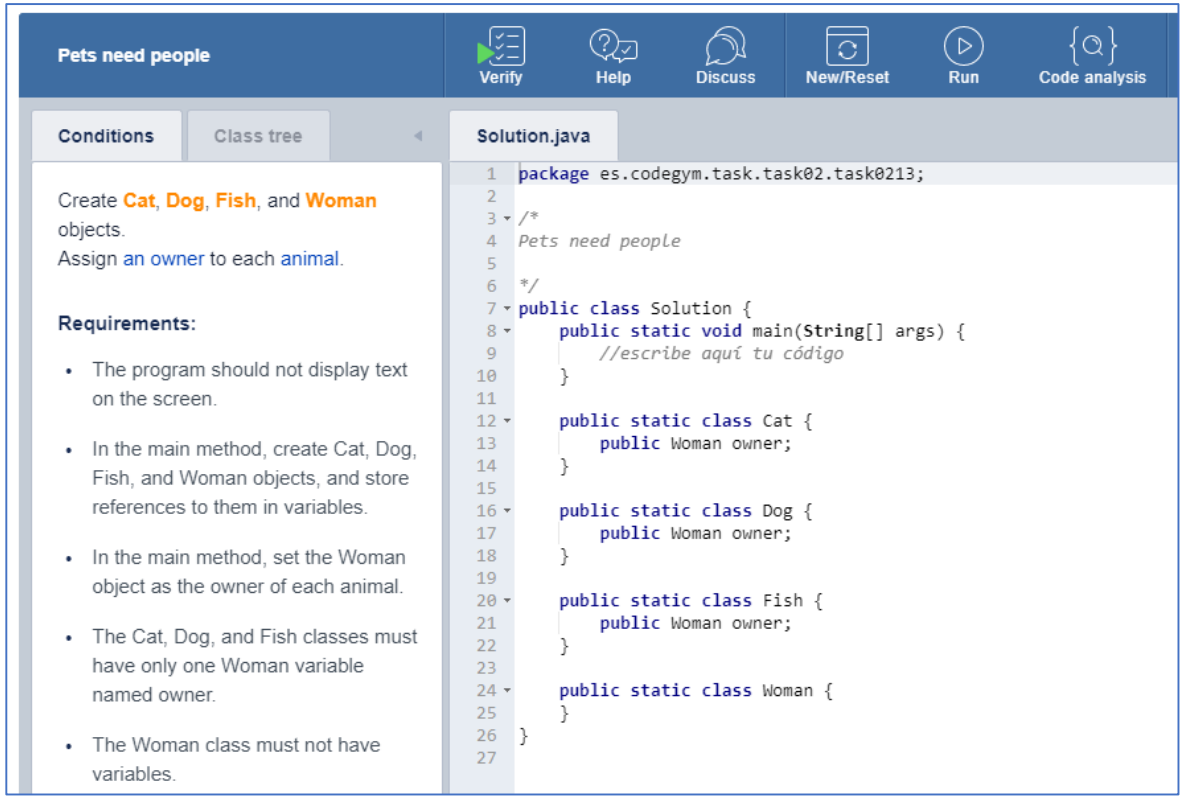
```java
package es.codegym.task.task02.task0212;

/*
Crazy eights

*/
public class Solution {
    public static void main(String[] args) {
        //escribe aquí tu código
        Cat cat8 = new Cat();
        Cat cat9;
        //escribe aquí tu código
    }

    public static class Cat {

    }
}

```

¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 5 / Nivel 2 – Lección 5

Excercise: **N2_L5_E3**



¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 6 / Nivel 2 – Lección 6

Reference variables

Sintaxis de Java
Level 2, Lesson 6

AVAILABLE

"Hi, Amigo, it's me again, Ellie. Sorry for saying that over and over again, but it's customary on the Earth in the 31st century. I'd like to give you more details about reference variables and passing reference variables to functions (methods)."

"I'm ready."

"Great, then listen. Reference variables are any non-primitive variables. Such variables only contain an object reference (a reference to an object)."

"Primitive variables contain values, while reference variables store references to objects or null. Am I right?"

"Absolutely."

"What's a reference?"

Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 6 / Nivel 2 – Lección 6

Example 1
The values m and n don't change here.

```
1  public class References
2  {
3    public static void main (String[] args)
4    {
5      int m = 5;
6      int n = 6;
7
8      System.out.println("M=" + m + " N=" + n);
9      swap( m, n );
10     System.out.println("M=" + m + " N=" + n);
11   }
12
13   private static void swap( int a, int b )
14   {
15     int c = a;
16     a = b;
17     b = c;
18   }
19 }
```

And here's why.
This code is analogous to the code on the left

```
1  public class References
2  {
3    public static void main (String[] args)
4    {
5      int m = 5;
6      int n = 6;
7
8      System.out.println("M=" + m + " N=" + n);
9      int a = m, b = n;
10
11     int c = a;
12     a = b;
13     b = c;
14
15     System.out.println("M=" + m + " N=" + n);
16   }
17 }
```

Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 6 / Nivel 2 – Lección 6

Example 2
The objects' data changes in this code

```
1  public class Primitives
2  {
3    public static void main(String[] args)
4    {
5      Student jen = new Student();
6      jen.name = "Jen";
7      jen.age = 21;
8
9      Student beth = new Student();
10     beth.name = "Beth";
11     beth.age = 15;
12
13     System.out.println("Jen is " + jen.age);
14     System.out.println("Beth is " + beth.age);
15
16     ageSwap(jen, beth);
17
18     System.out.println("Jen is " + jen.age);
19     System.out.println("Beth is " + beth.age);
20   }
21
22   private static void ageSwap(Student a,
23                               Student b)
24   {
25     int c = a.age;
26     a.age = b.age;
27     b.age = c;
28   }
29
30   static class Student
31   {
32     String name;
33     int age;
34   }
```

And here's why.
This code is analogous to the code on the left

```
1  public class Primitives
2  {
3    public static void main(String[] args)
4    {
5      Student jen = new Student();
6      jen.name = "Jen";
7      jen.age = 21;
8
9      Student beth = new Student();
10     beth.name = "Beth";
11     beth.age = 15;
12
13     System.out.println("Jen is " + jen.age)
14     System.out.println("Beth is " + beth.ag
15
16     Student a = jen, b = beth;
17
18     int c = a.age;
19     a.age = b.age;
20     b.age = c;
21
22     System.out.println("Jen is " + jen.age)
23     System.out.println("Beth is " + beth.ag
24   }
25
26
27
28
29
30   static class Student
31   {
32     String name;
33     int age;
34   }
```

¡Siempre hacia lo alto!

Feedback

¡Siempre hacia lo alto!

Survey

¡Siempre hacia lo alto!

Clarification of doubts

¡Siempre hacia lo alto!

5° Session CodeGym

¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 7 / Nivel 2 – Lección 7

Calling methods, returning values

Sintaxis de Java
Level 2, Lesson 7

AVAILABLE

"OK, let's try another approach. I'll show you how calling methods works, and then you try to go through the previous lesson again, OK?"

"Let's do it."

"Great. I'll tell you about calling functions/methods and the values they return (return values)."

"Commands, or statements, are grouped into methods so they can be executed as a single block, like a single complex command. To do this, you need to write the method (function) name and then list the method's arguments inside parentheses."

Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 7 / Nivel 2 – Lección 7

Example

```
1  package com.codegym.lesson2;
2  public class MethodCall
3  {
4      public static void main(String[] args)
5      {
6          print4("I like to move it, move it.");
7      }
8
9      public static void print4(String s)
10     {
11         System.out.println(s);
12         System.out.println(s);
13         System.out.println(s);
14         System.out.println(s);
15     }
16 }
```

Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 7 / Nivel 2 – Lección 7

Example 1.
Determine the minimum of two numbers.

```
1  public class MethodCall
2  {
3     public static void main(String[] args)
4     {
5        int a = 5, b = 7;
6        int m = min(a, b);
7        System.out.println("The minimum is "+ m);
8     }
9
10    public static int min(int c, int d)
11    {
12       int m2;
13       if (c < d)
14            m2 = c;
15       else
16            m2 = d;
17
18       return m2;
19    }
20 }
```

Here's how it works:

```
1  public class MethodCall
2  {
3     public static void main(String[] args)
4     {
5        int a = 5, b = 7;
6        int c = a, d = b;
7        int m2;
8        if (c < d)
9             m2 = c;
10       else
11            m2 = d;
12
13       int m = m2;
14       System.out.println("The minimum is "+ m
15    }
16 }
```

Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 7 / Nivel 2 – Lección 7

Repeat the following code:

```
1  public class MethodCall
2  {
3      public static void main(String[] args)
4      {
5          int a = 5, b = 7;
6          int m = min(a, b);
7          System.out.println("The minimum is " + m);
8      }
9
10     public static int min(int c, int d)
11     {
12         int m2;
13         if (c < d)
14             m2 = c;
15         else
```

Your code:

```
1
```

Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 8 / Nivel 2 – Lección 8

Practice calling methods

Sintaxis de Java
Level 2, Lesson 8

AVAILABLE

"Hi, Amigo. I'd like to give you a couple of tasks:"

Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 8 / Nivel 2 – Lección 8

Excercise: **N2_L8_E1**

Minimum of two numbers

Verify | Help | Discuss | New/Reset | Run | Code analysis

Conditions | Class tree

Write a function that returns the minimum of two numbers.

Hint:

You need to write the body of the existing `min` *function.*

Requirements:

- The program should display text on the screen.
- The min method should not display text on the screen.

Solution.java

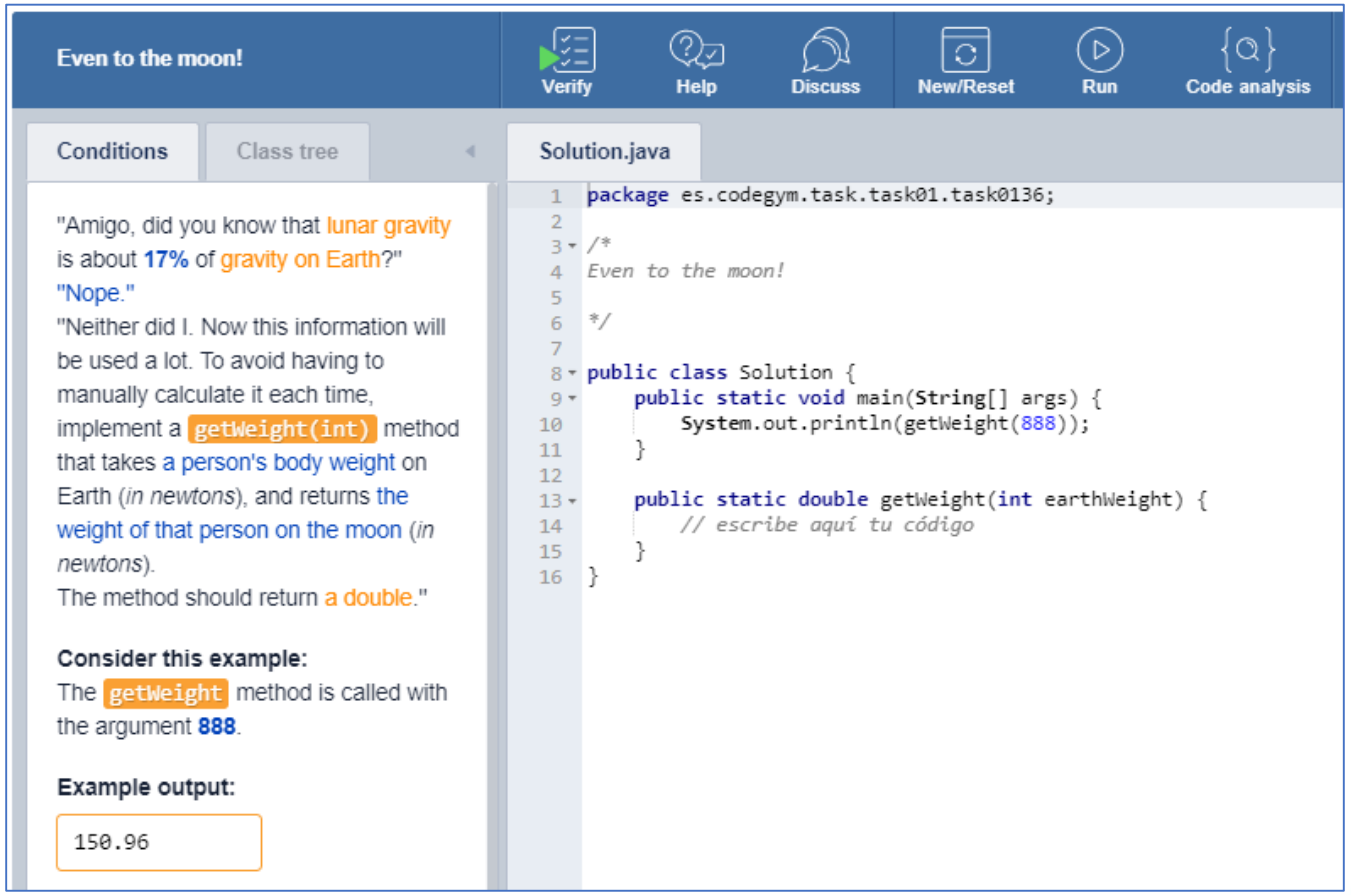
```
1  package es.codegym.task.task02.task0214;
2
3  /*
4  Minimum of two numbers
5
6  */
7  public class Solution {
8      public static int min(int a, int b) {
9          //escribe aquí tu código
10     }
11
12     public static void main(String[] args) throws Exception {
13         System.out.println(min(12, 33));
14         System.out.println(min(-20, 0));
15         System.out.println(min(-10, -20));
16     }
17 }
```

¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 8 / Nivel 2 – Lección 8

Excercise: **N2_L8_E2**



¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 8 / Nivel 2 – Lección 8

Excercise: **N2_L8_E3**

Minimum of three numbers

Verify | Help | Discuss | New/Reset | Run | Code analysis

Conditions | Class tree

Write a function that computes the minimum of three numbers.

Hint:

You need to write the body of the existing `min` *function.*

Requirements:

- The program should display text on the screen.
- The min method should not display text on the screen.

Solution.java

```java
package es.codegym.task.task02.task0216;

/*
Minimum of three numbers

*/
public class Solution {
    public static int min(int a, int b, int c) {
        //escribe aquí tu código
    }

    public static void main(String[] args) throws Exception {
        System.out.println(min(1, 2, 3));
        System.out.println(min(-1, -2, -3));
        System.out.println(min(3, 5, 3));
        System.out.println(min(5, 5, 10));
    }

}
```

¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 8 / Nivel 2 – Lección 8

Excercise: **N2_L8_E4**

Minimum of four numbers

Verify | Help | Discuss | New/Reset | Run | Code analysis

Conditions | Class tree

Write a function that computes the minimum of four numbers.
The function `min(a, b, c, d)` should use (*call*) the function `min(a, b)`

Hint:
You need to write the body of the two existing `min` *functions.*

Requirements:

- The program should display text on the screen.
- The min methods should not display text on the screen.
- The main method should call min(a, b) and min(a, b, c, d).

Solution.java

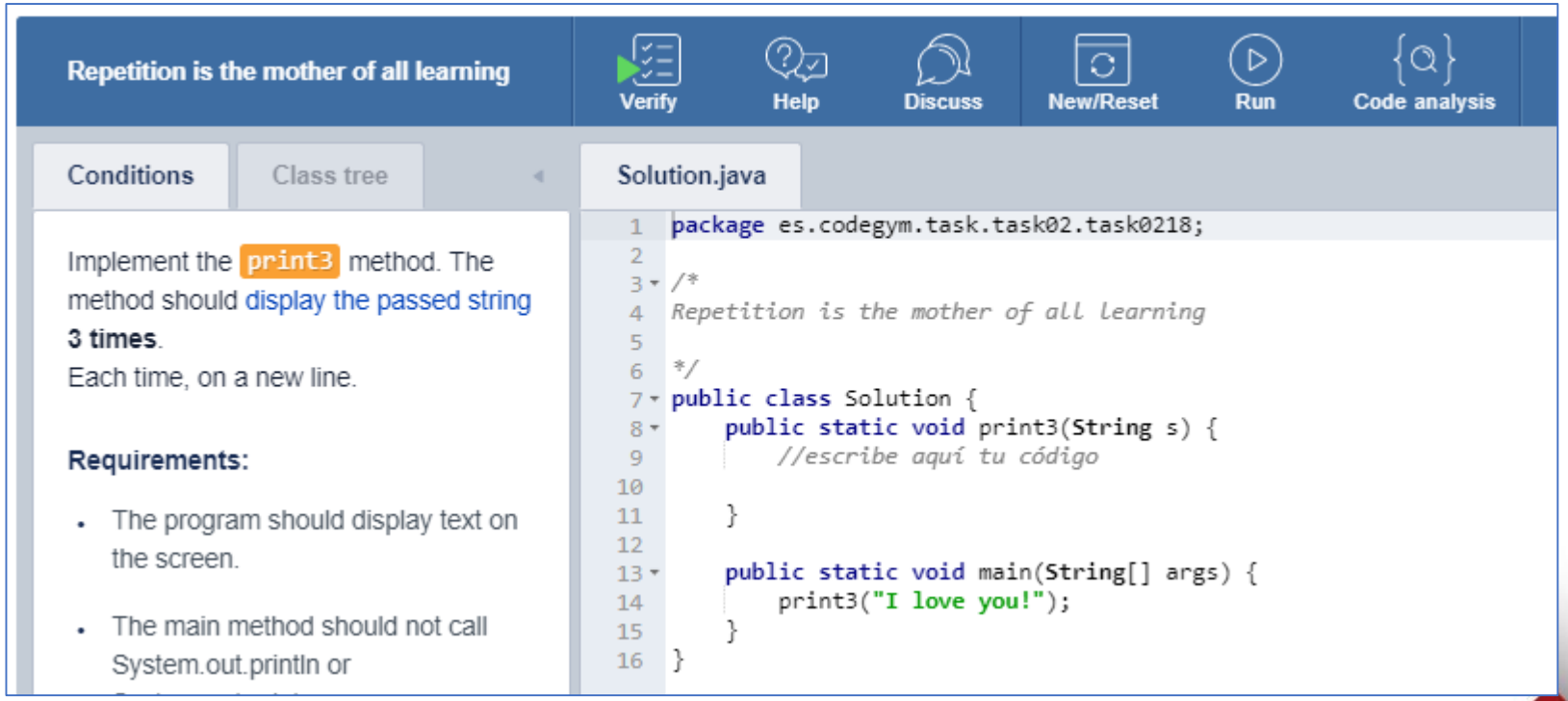
```
1  package es.codegym.task.task02.task0217;
2
3  /*
4  Minimum of four numbers
5
6  */
7  public class Solution {
8      public static int min(int a, int b, int c, int d) {
9          //escribe aquí tu código
10
11     }
12
13     public static int min(int a, int b) {
14         //escribe aquí tu código
15
16     }
17
18     public static void main(String[] args) throws Exception {
19         System.out.println(min(-20, -10));
20         System.out.println(min(-20, -10, -30, -40));
21         System.out.println(min(-20, -10, -30, 40));
22         System.out.println(min(-40, -10, -30, 40));
23     }
24 }
```

¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 8 / Nivel 2 – Lección 8

Excercise: **N2_L8_E5**

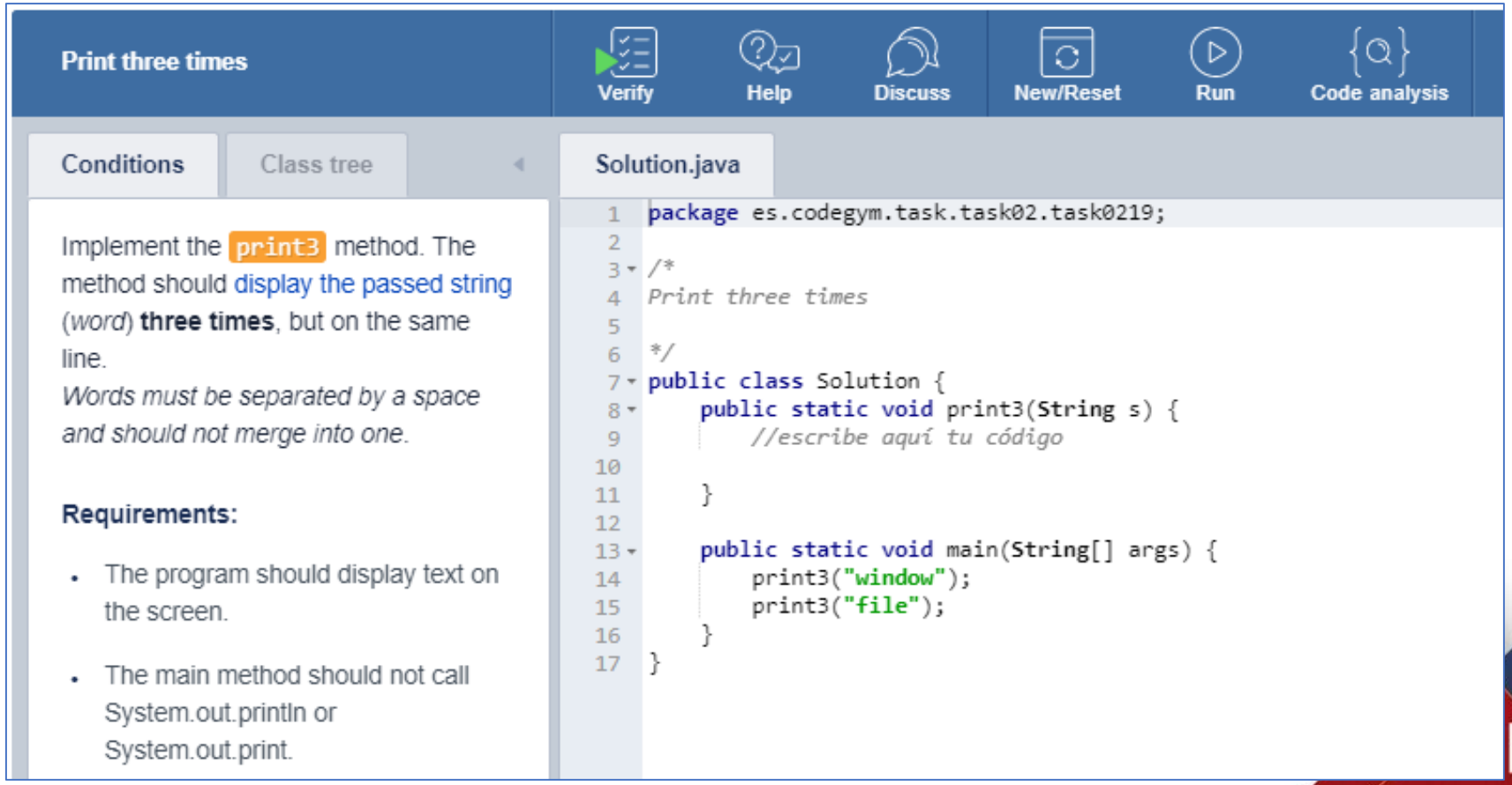


¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 8 / Nivel 2 – Lección 8

Excercise: **N2_L8_E6**



¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 9 / Nivel 2 – Lección 9

¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 9 / Nivel 2 – Lección 9

Class name	Package name	Full name
`String`	java.lang	java.lang.**String**
`FileInputStream`	java.io	java.io.**FileInputStream**
`ArrayList`	java.util	java.util.**ArrayList**
`IOException`	java.io	java.io.**IOException**;

Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 9 / Nivel 2 – Lección 9

Full class name:

```
1  package com.codegym.lesson2;
2
3  public class FileCopy2
4  {
5      public static void main(String[] args) throws java.io.IOException
6      {
7          java.io.FileInputStream fileInputStream =
8                          new java.io.FileInputStream("c:\\data.txt");
9          java.io.FileOutputStream fileOutputStream =
10                         new java.io.FileOutputStream("c:\\result.txt");
11
12         while (fileInputStream.available() > 0)
13         {
14             int data = fileInputStream.read();
15             fileOutputStream.write(data);
16         }
17
18         fileInputStream.close();
19         fileOutputStream.close();
20     }
21 }
```

Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 9 / Nivel 2 – Lección 9

Short class name:

```
1  package com.codegym.lesson2;
2
3  import java.io.FileInputStream;
4  import java.io.FileOutputStream;
5  import java.io.IOException;
6
7  public class FileCopy
8  {
9      public static void main(String[] args) throws IOException
10     {
11         FileInputStream fileInputStream =
12                         new FileInputStream("c:\\data.txt");
13         FileOutputStream fileOutputStream =
14                         new FileOutputStream("c:\\result.txt");
15
16         while (fileInputStream.available() > 0)
17         {
18             int data = fileInputStream.read();
19             fileOutputStream.write(data);
20         }
21
22         fileInputStream.close();
23         fileOutputStream.close();
24     }
25 }
```

Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 10 / Nivel 2 – Lección 10

¡Siempre hacia lo alto!

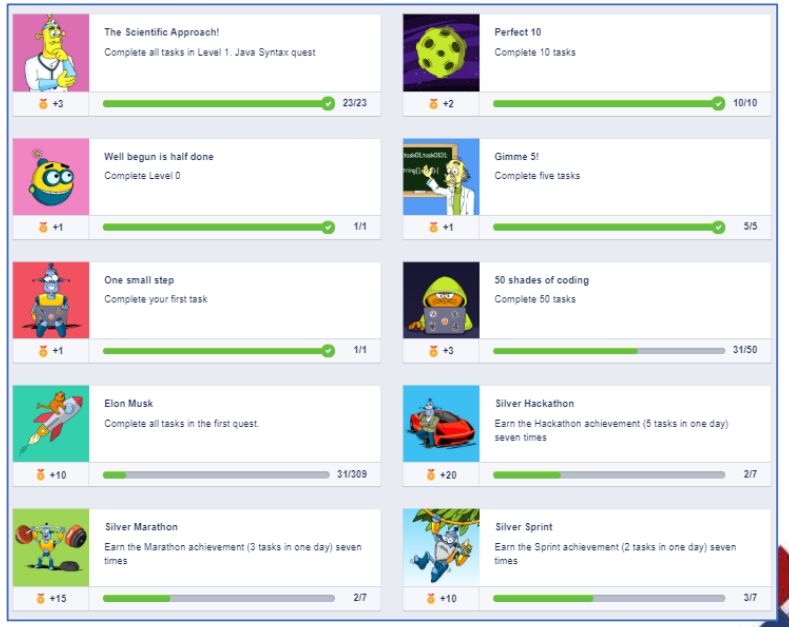
Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 11 / Nivel 2 – Lección 11

¡Siempre hacia lo alto!

Working document with full level 2! / Documento de trabajo con nivel 2 completo!

Do not forget screenshot of the achievement board / No olvides pantallazo del tablero de logros



Upload it to your drive folder: /

Súbelo a tu carpeta en drive.:

¡Siempre hacia lo alto!

Feedback

¡Siempre hacia lo alto!

Survey

¡Siempre hacia lo alto!

Clarification of doubts

¡Siempre hacia lo alto!

To practice!

¡Siempre hacia lo alto!

P1T50: Methods of geometric figures

Create three classes of different geometric figures, which have variables according to each figure. Within the class, create the functions or methods corresponding to the area and perimeter of each one.

Crear tres clases de diferentes figuras geométricas, las cuales tengan variables acordes a cada figura. Dentro de la clase, crear las funciones o métodos correspondientes al área y perímetro de cada una.

P1T51: Calculating

Create a calculating class, which has as its methods the basic functions (addition, subtraction, multiplication, division, module), each in different functions. The methods receive two numbers as parameters. The user enters the two numbers and the desired operation.

Crear una clase calculadora, la cual tenga como métodos las funciones básicas (suma, resta, multiplicación, división, modulo), cada una en funciones diferentes. Los métodos reciben como parámetro dos números. El usuario ingresa los dos números y la operación deseada.

P1T52: Equation in 2nd grade

Create a program that asks for the coefficients of an equation in 2nd grade, and shows its real solutions. If they do not exist, you must indicate it.

Crear un programa que pida los coeficientes de una ecuación se 2º grado, y muestre sus soluciones reales. Si no existen, debe indicarlo.

P1T53: Conversion

Build a function that converts kilometers into miles and another that converts Colombian pesos into dollars.

Construir una función que convierta kilómetros en millas y otra que convierta pesos colombianos en dólares.

P1T54: First letter of a word

Create a method that returns the first letter of a word entered as a parameter.

Crear un método que retorne la primera letra de una palabra ingresada como parámetro.

P1T55: Average

Create a method that receives as parameters 5 numbers and returns its calculated average.

Crear un método que reciba como parámetros 5 números y retorne su promedio calculado.

P1T56: Person

In a class called a person, create a function that calculates a worker's salary. Keep in mind that the worker is paid for hours worked. If the hours worked are greater than 80 hours, the salary increases by 20%. The hourly rate is entered by the user.

En una clase llamada persona, cree una función que calcule el salario de un trabajador. Tenga en cuenta que al trabajador se le paga por horas trabajadas. Si las horas trabajadas es mayor que 80 horas, el sueldo incrementa en 20%. La tarifa por hora la ingresa el usuario.

P1T57: Salary

A worker's salary decreases by 3% if he earns less than $ 800,000. If the salary is above 800,000 and is less than 1,500,000, a 5% discount is generated. If, on the contrary, the salary is above 1,500,000 the discount will be 8%. Create a function that calculates the net salary and discount, based on the salary you receive by parameter.

El salario de un trabajador disminuye en 3% si gana menos de $800.000. Si el salario está por encima de 800.000 y es inferior a 1.500.000 se genera un 5% de descuento al mismo. Si por el contrario, el salario esta por encima de 1.500.000 el descuento será del 8%. Crear una función que calcule el salario neto y el descuento, en base al salario que recibe por parámetro

P1T58: Notes

Create a function that receives as a parameter five notes corresponding to five subjects. The first subject will have a weighted 35% of the total value of the average. The second and third subjects will have a value of 15% each. The fourth subject has a weight of 25%, and the fifth will be worth 10% of the total average. If a student obtains a total average of 4.7, he will be awarded a scholarship with 100% of the tuition fee. If the average is between 4.3 and 4.6 the student will get a partial scholarship of 40%. Otherwise, the student will not be awarded a scholarship.

Crear una función que reciba como parámetro cinco notas correspondientes a cinco asignaturas. La primera asignatura tendrá un ponderado del 35% del valor total del promedio. La segunda y tercera asignatura tendrán un valor de 15% cada una. La cuarta asignatura tiene un peso del 25%, y la quinta valdrá un 10% del promedio total. Si un estudiante obtiene un promedio total de 4.7 será becado con el 100% del valor de la matriculo. Si el promedio se encuentra entre 4.3 y 4.6 el estudiante obtendrá una beca parcial del 40%. De lo contrario, el estudiante no será becado

P1T58: beverage machine

Crear un software en JAVA similar al de la maquina expendedora de bebidas calientes (como la que esta ubicada en la USTA), el software le mostrara la disponibilidad de productos (menú) al usuario:

Producto	Valor	Tiempo (creación)
Aromática:	$1.300	30 segundos
Café negro:	$1.000	30 segundos
Café con leche:	$1.900	45 segundos
Capuchino:	$2.500	60 segundos
Mocachino:	$2.700	70 segundos

El usuario debe ingresar el valor del billete y que producto desea, con esa información el software debe validar que:

- El producto exista de lo contrario debe generar un error y volver a mostrarle el menú.
- Que el dinero le alcanza para comprar el producto de lo contrario debe generarle una solicitud que ingrese el dinero faltante (WHILE).

Una vez validado lo anterior debe generar un mensaje diciendo cuanto tiempo se va a demorar procesando el producto el total de los vueltos.

6° Session CodeGym

¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 3 - Lesson 0 / Nivel 3 – Lección 0

¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 3 - Lesson 1 / Nivel 3 – Lección 1

The simplest program

Java Syntax
Level 3, Lesson 1

AVAILABLE

"Hi, Diego!"

"Hi, Amigo!"

"The professor praised me recently. He was glad that I'm making such fast progress thanks to his lessons."

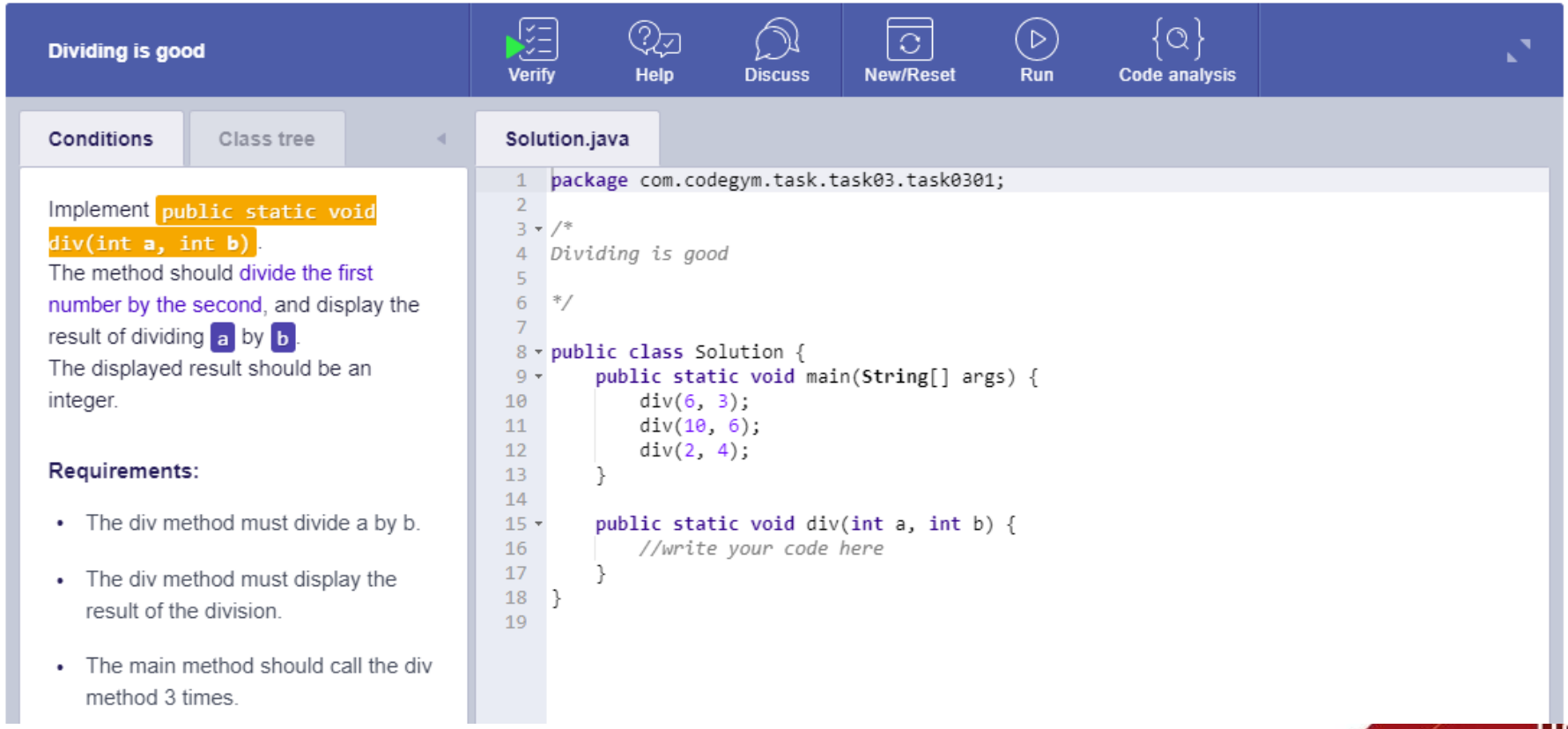
"You're making progress thanks to the professor's lessons?! Oh, sure! Doesn't he realize how funny that is?"

"Well, never mind. I have something interesting for you today. I'll teach you how to write the simplest (or minimal) program. It's very easy. A minimal program consists of one class and contains one method - main(). This is how it looks."

Gamified platform work / Trabajo en plataforma gamificada.

Level 3 - Lesson 1 / Nivel 3 – Lección 1

Excercise: **N3_L1_E1**



Gamified platform work / Trabajo en plataforma gamificada.

Level 3 - Lesson 2 / Nivel 3 – Lección 2

Connecting IntelliJ IDEA

Java Syntax
Level 3, Lesson 2

AVAILABLE

"I'd like to tell you how to do your homework with IntelliJ IDEA. I'm sure you'll soon appreciate how powerful it is. For starters, let's stick to the following procedure."

Installing the Plugin

STEP 1. Click the 'Download plugin' link.

STEP 2. Run IntelliJ IDEA. Go to File -> Settings, and find Plugins. For MacOS, click IntelliJ IDEA -> Preferences -> Plugins.

STEP 3. Click on the gear and select 'Install plugins from disk'

Gamified platform work / Trabajo en plataforma gamificada.

Level 3 - Lesson 3 / Nivel 3 – Lección 3

Screen output once again

Java Syntax
Level 3, Lesson 3

AVAILABLE

"Long ago, computers could only display text. Programs displayed data on the screen after receiving input from the keyboard. This is called a 'console user interface' or simply the 'console'. A window interface is an alternative to the console. With this type of interface, the user interacts with the program through one or more windows. Since we're just learning how to program, we'll start by working with the console."

"All right."

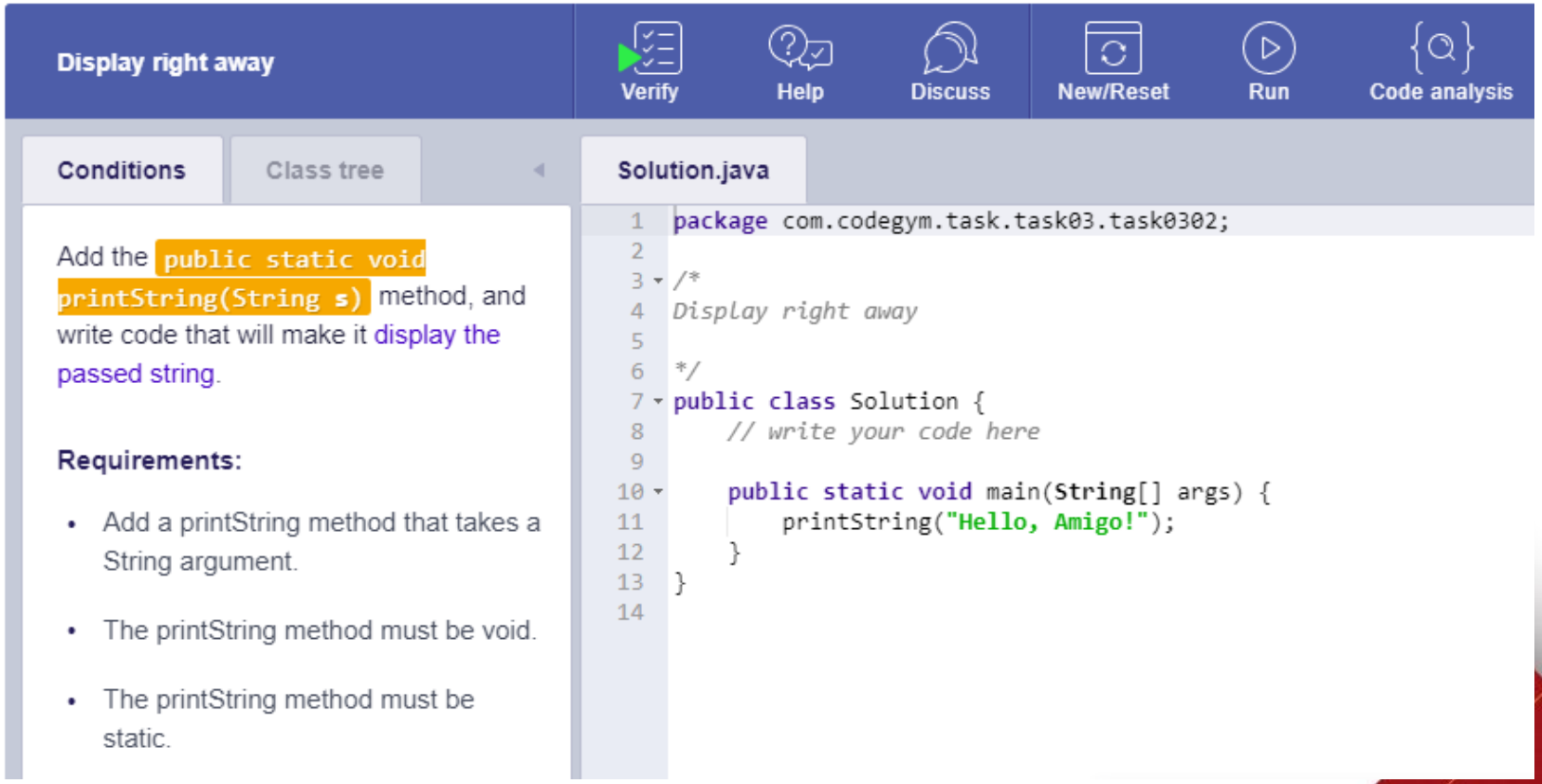
"Text is displayed on the console (screen) consecutively, line by line. The text is entered using the keyboard. To avoid mistakes, the keyboard input is displayed on the screen. Sometimes it looks like the human user and the program are taking turns writing things on the screen."

"You can use the **System.out.print**() method to display text on the screen. This method simply displays the text, while **System.out.println**() displays the text and moves the cursor to the next line."

Gamified platform work / Trabajo en plataforma gamificada.

Level 3 - Lesson 3 / Nivel 3 – Lección 3

Excercise: **N3_L3_E1**

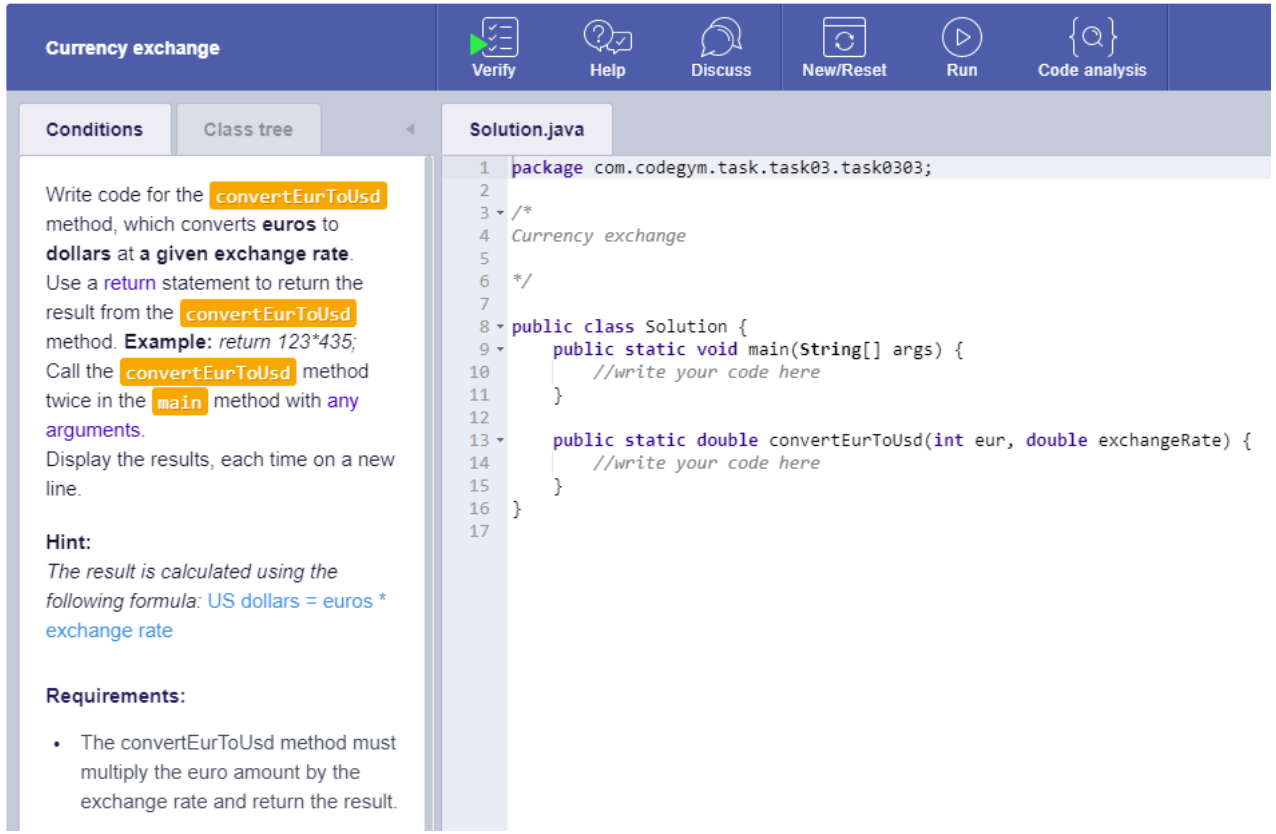


¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 3 - Lesson 3 / Nivel 3 – Lección 3

Excercise: **N3_L3_E2**

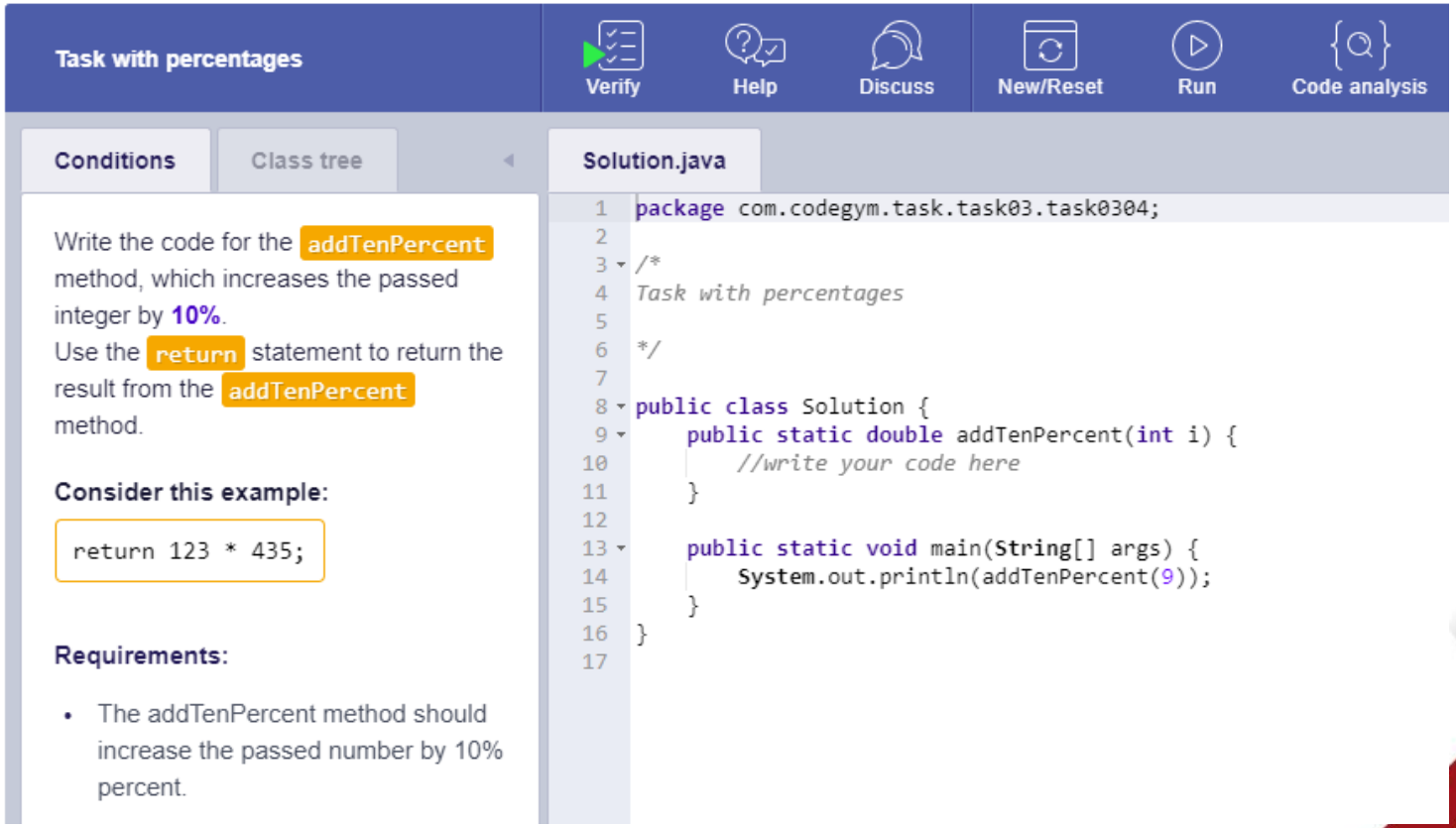


¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 3 - Lesson 3 / Nivel 3 – Lección 3

Excercise: **N3_L3_E3**



¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 3 - Lesson 4 / Nivel 3 – Lección 4

Practice displaying stuff

Java Syntax
Level 3, Lesson 4

AVAILABLE

"Hi, Amigo. Here are a few tasks to help you practice displaying things on the screen:"

×1

Task Java Syntax, level 3, lesson 4

EASY

AVAILABLE

Red scare

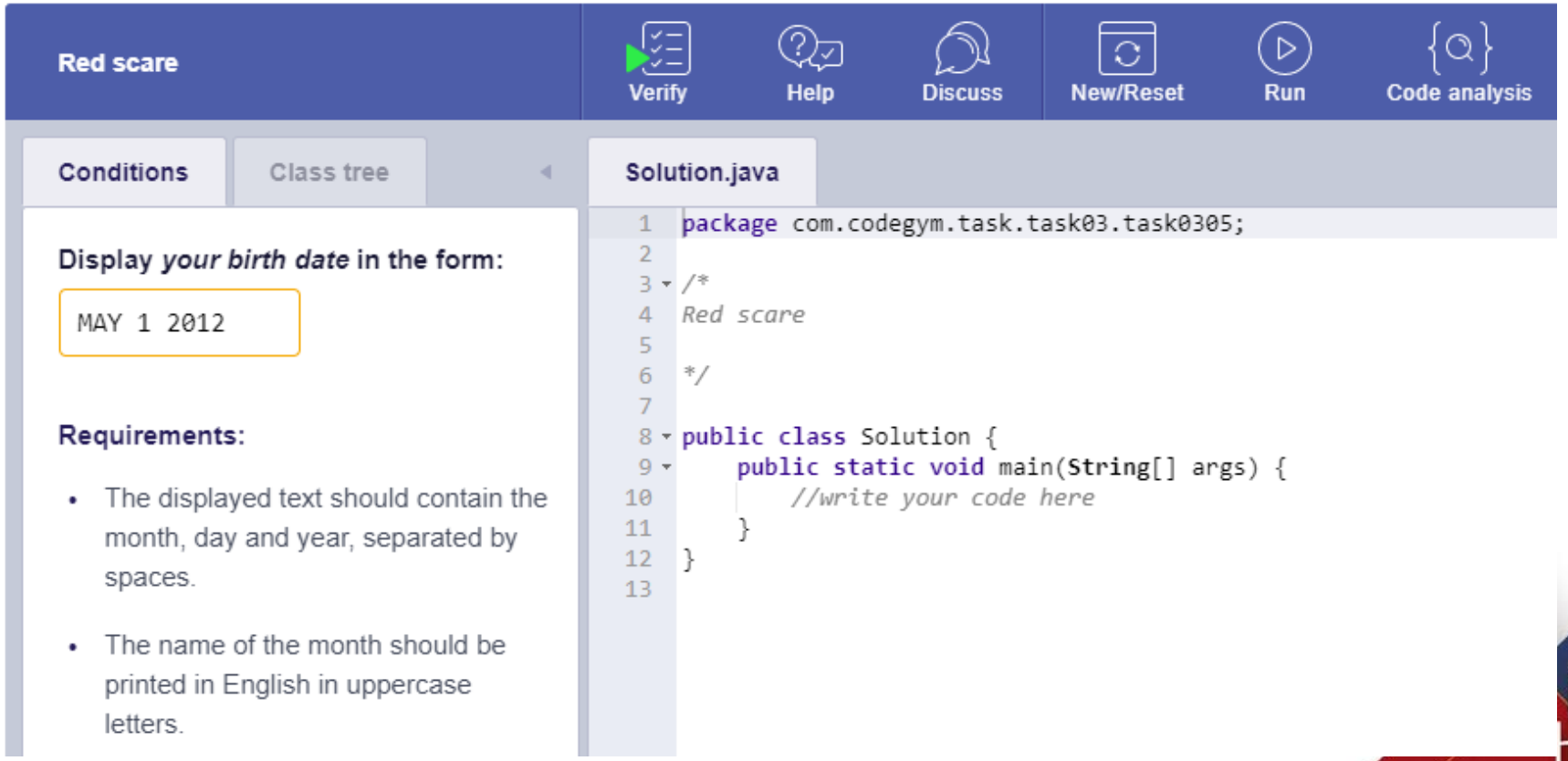
Dear CodeGym student, let's add a couple of lines to your dossier. No, no. Nothing like that. Just your date of birth. Why do we need this? We don't. It's for you. For educational purposes, of course. Just display your birth date in the format MAY 1 2012. And who are your parents? And do you have any subversive tendencies?

Open

Gamified platform work / Trabajo en plataforma gamificada.

Level 3 - Lesson 4 / Nivel 3 – Lección 4

Excercise: **N3_L4_E1**

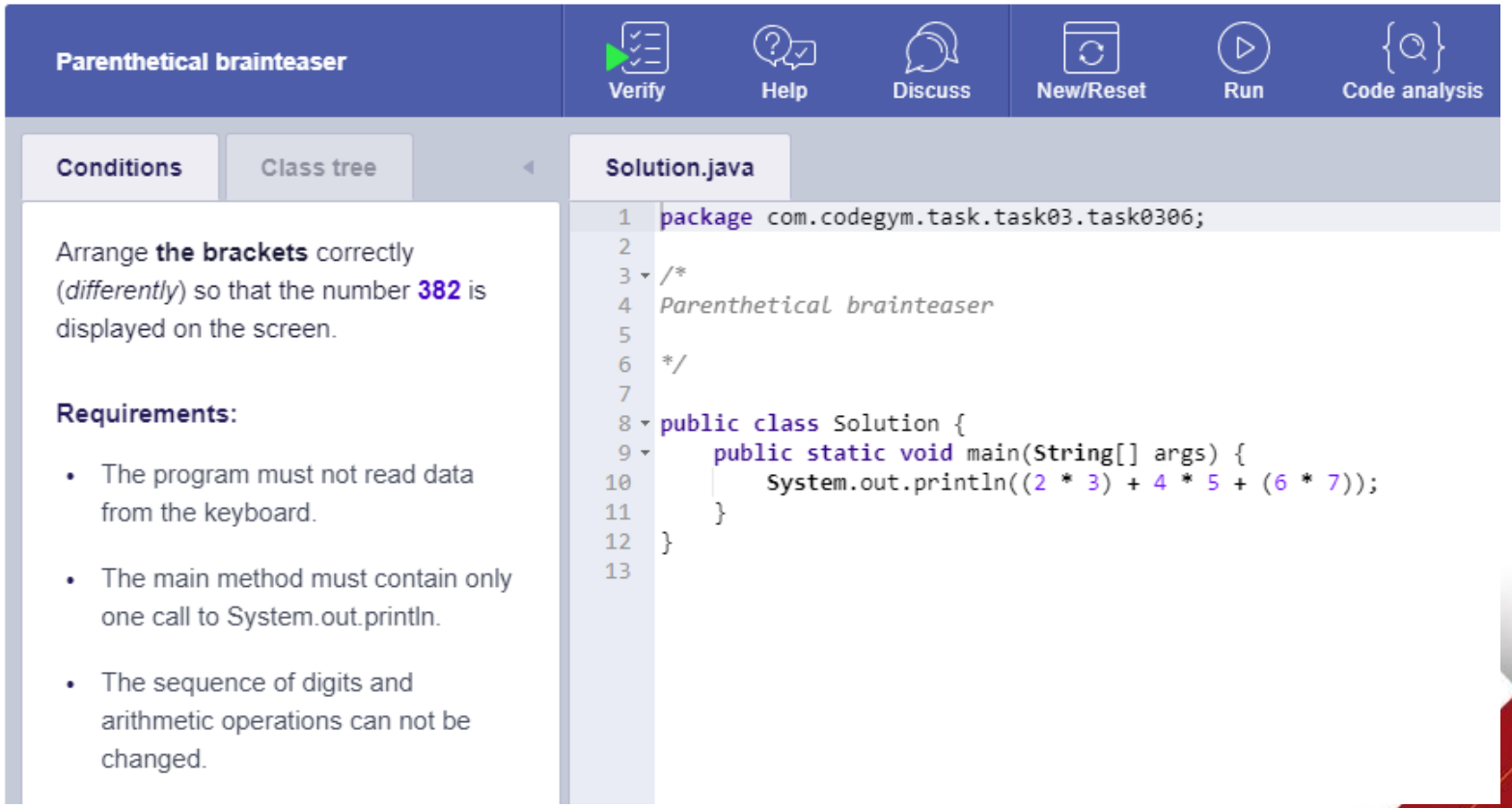


¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 3 - Lesson 4 / Nivel 3 – Lección 4

Excercise: **N3_L4_E2**

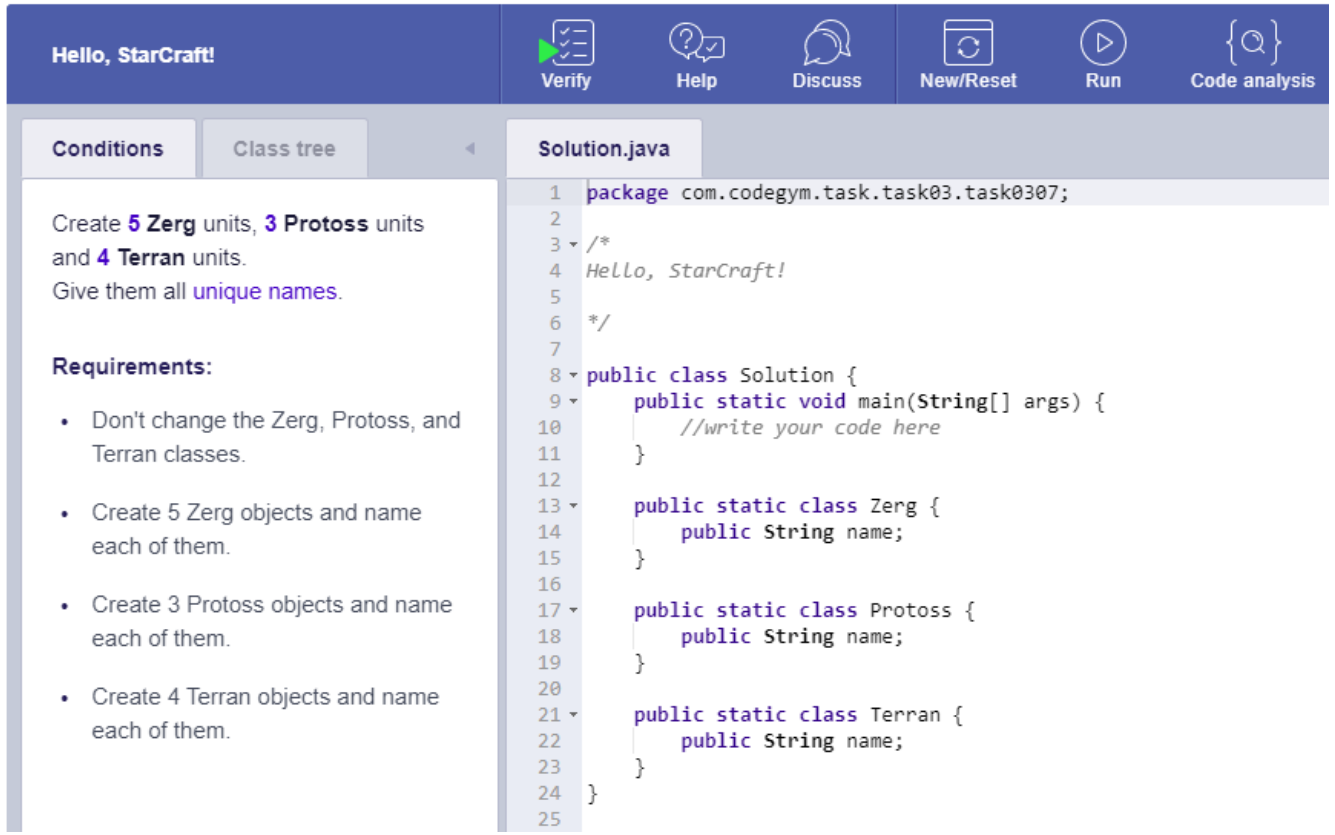


¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 3 - Lesson 4 / Nivel 3 – Lección 4

Excercise: **N3_L4_E3**

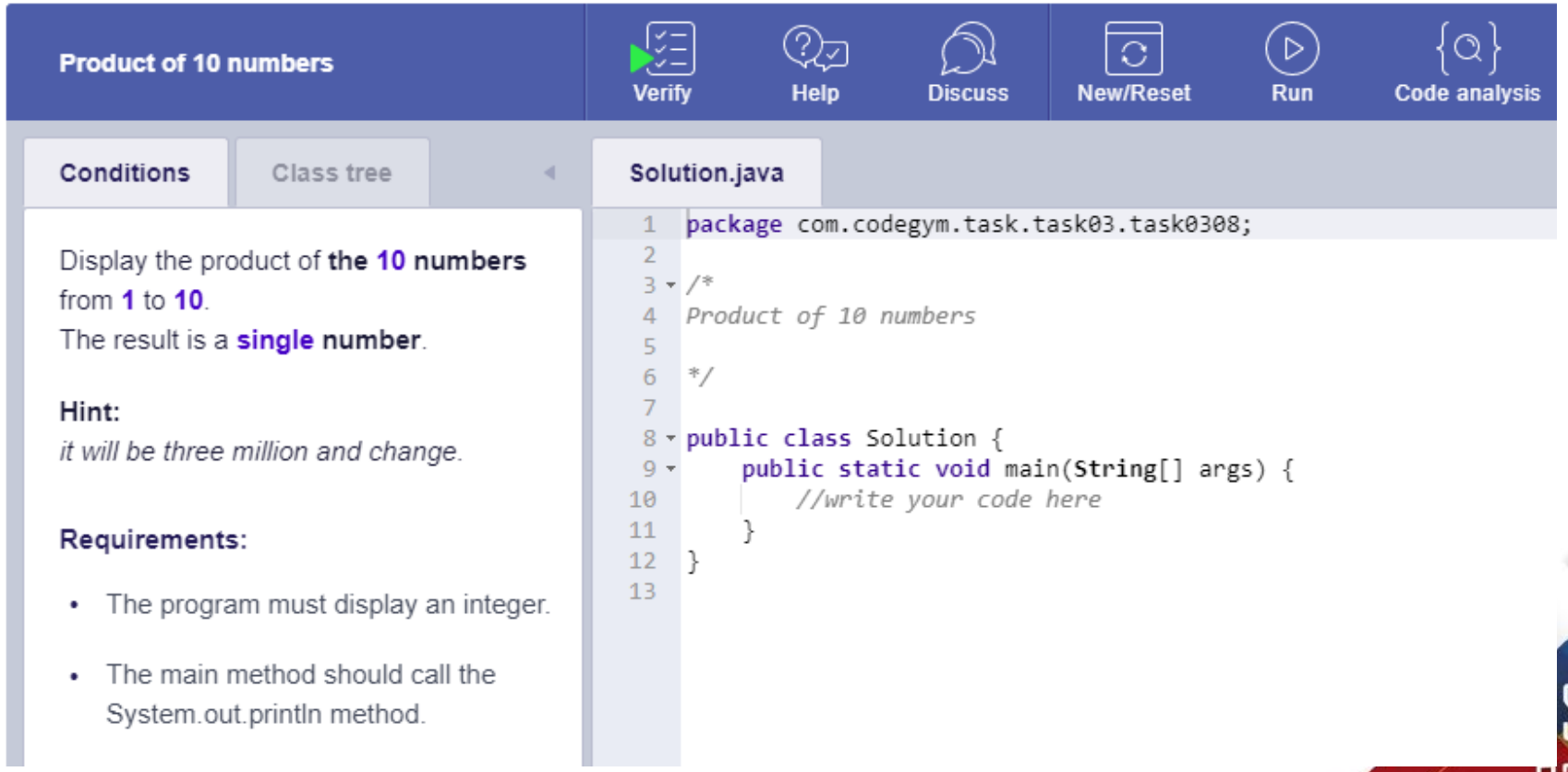


¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 3 - Lesson 4 / Nivel 3 – Lección 4

Excercise: **N3_L4_E4**

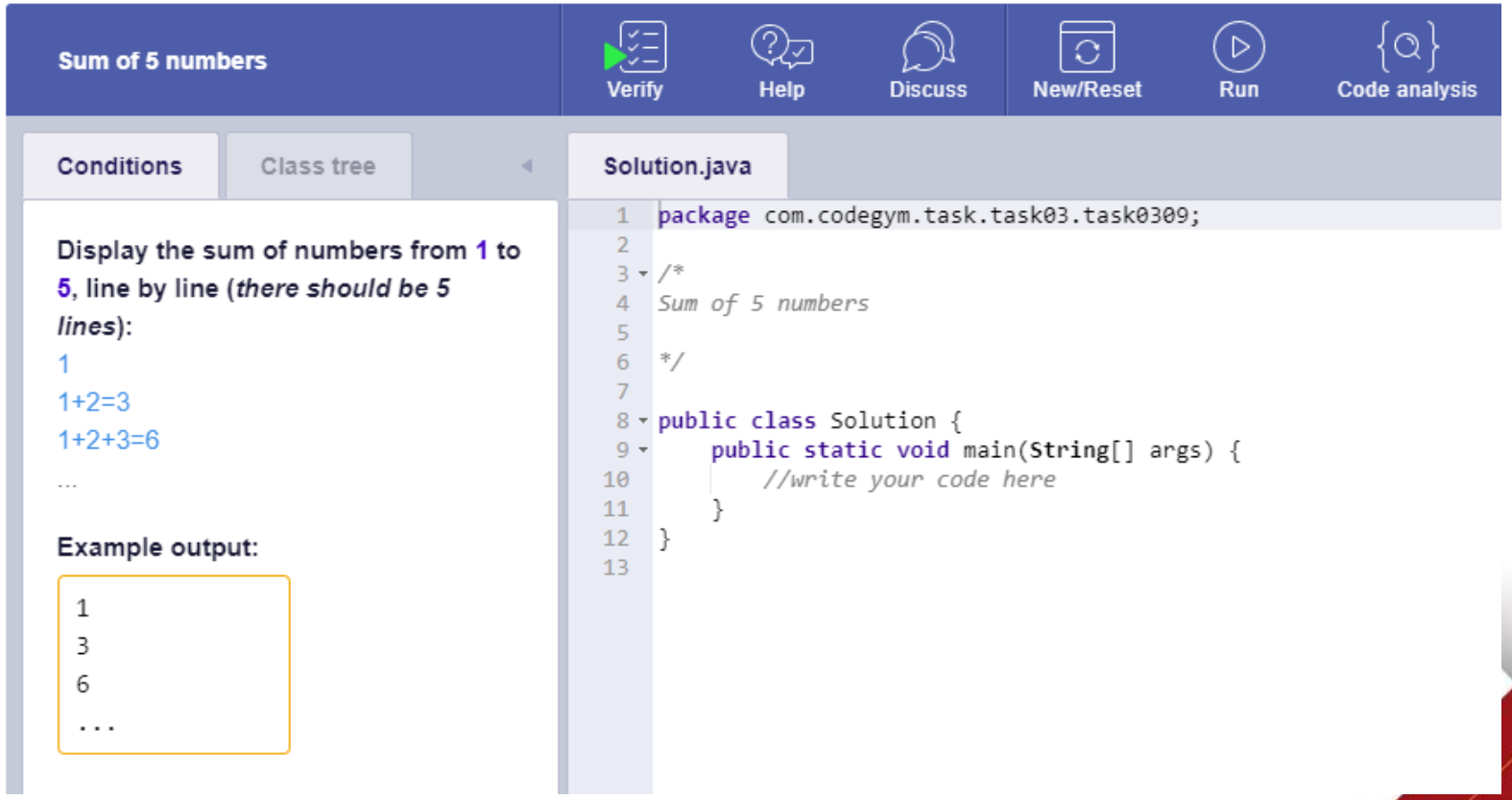


Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 3 - Lesson 4 / Nivel 3 – Lección 4

Excercise: **N3_L4_E5**



¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 3 - Lesson 5 / Nivel 3 – Lección 5

Concatenation

Java Syntax
Level 3, Lesson 5

AVAILABLE

"I'd like to tell you how to merge strings. The process of merging or joining strings is often referred to using the short word 'concatenation'. Cat lovers will find it easy to remember: con-Cat-en-Nation. **I'm joking**."

"The rules for merging strings are simple. If we 'add' (+) a string and something else, then the 'something else' is implicitly converted to a string via the toString() method."

"Were you talking to me just now?"

"Okay, I'll explain it in an easier way. If we add a string, a number and a cat, then both the number and the cat will be transformed into strings. Here are some examples:"

Gamified platform work / Trabajo en plataforma gamificada.

Level 3 - Lesson 5 / Nivel 3 – Lección 5

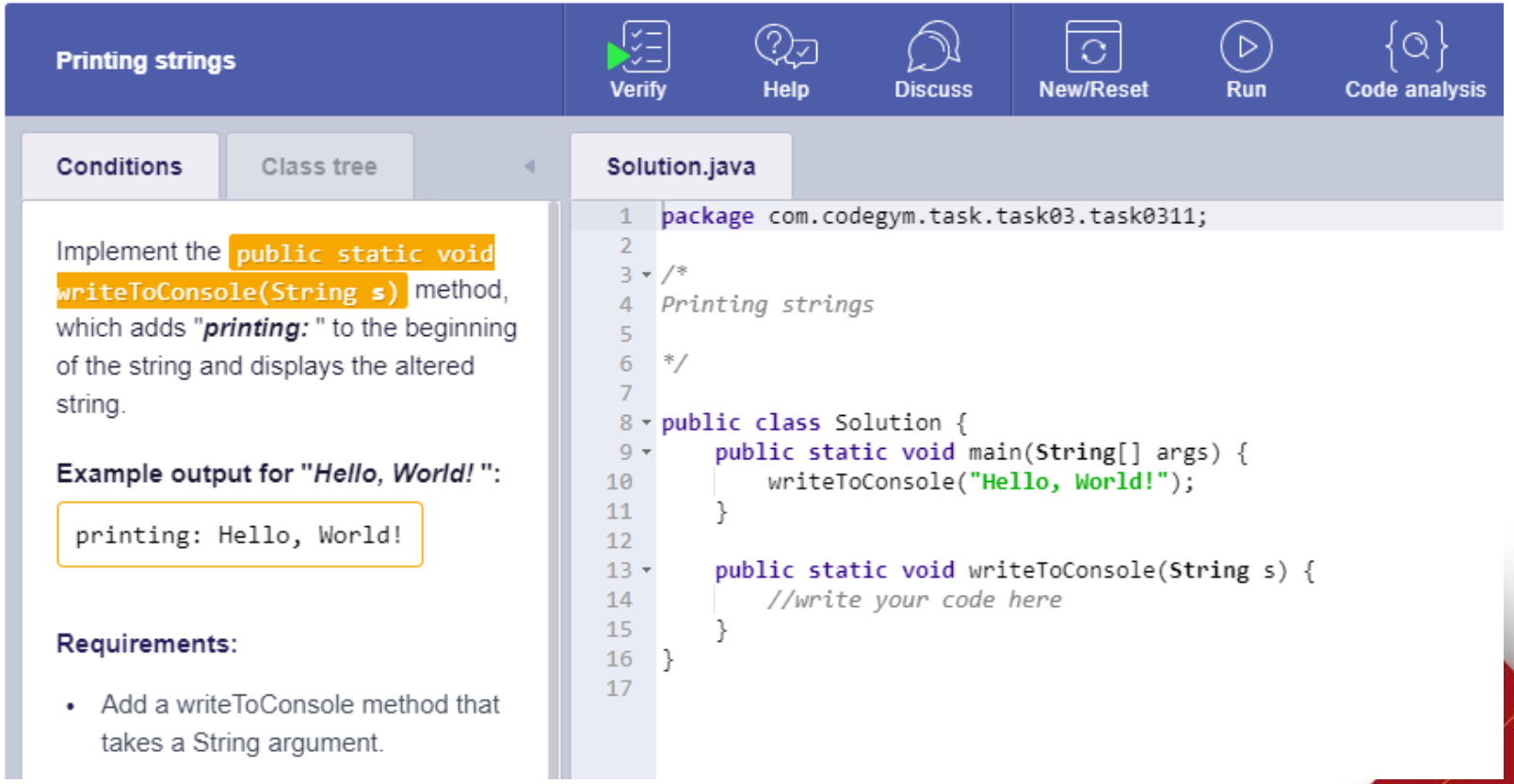
Excercise: **N3_L5_E1**

¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 3 - Lesson 5 / Nivel 3 – Lección 5

Excercise: **N3_L5_E2**

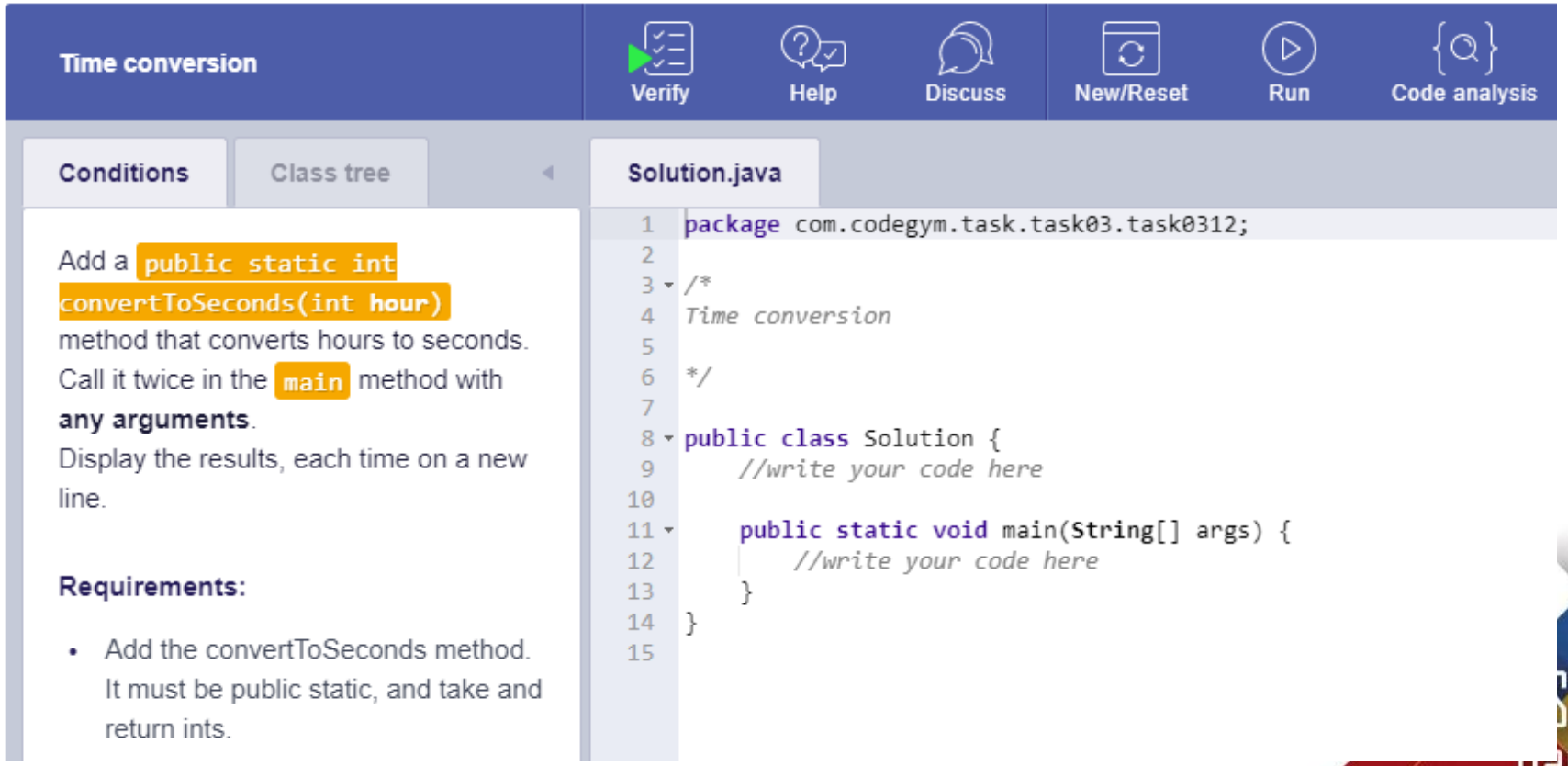


¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 3 - Lesson 5 / Nivel 3 – Lección 5

Excercise: **N3_L5_E3**



Gamified platform work / Trabajo en plataforma gamificada.

Level 3 - Lesson 6 / Nivel 3 – Lección 6

Tasks

Java Syntax
Level 3, Lesson 6

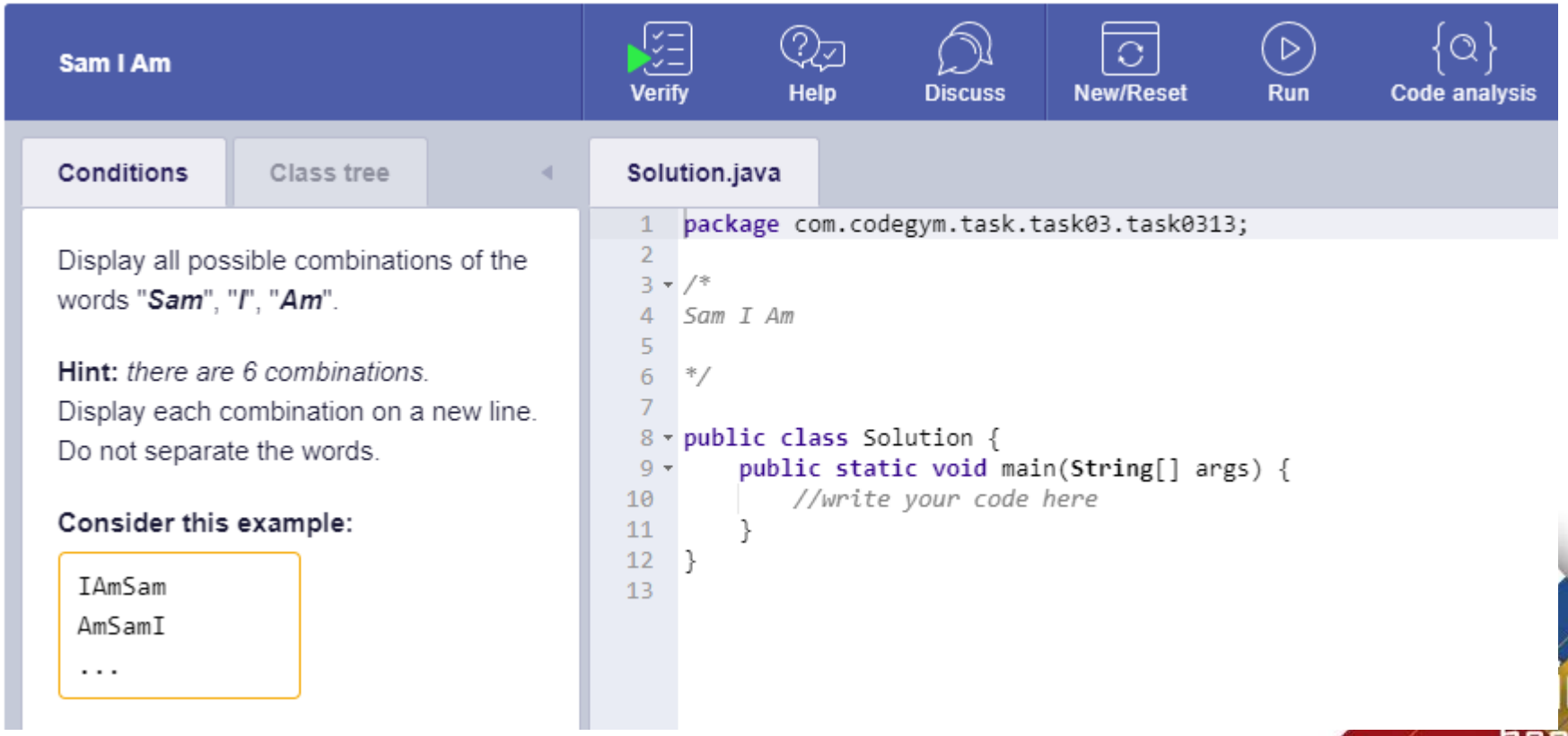
AVAILABLE

"Hi, buddy. Here are a couple more tasks for today. To make them more challenging, I want you to only use variables inside the print/println method."

Gamified platform work / Trabajo en plataforma gamificada.

Level 3 - Lesson 6 / Nivel 3 – Lección 6

Excercise: **N3_L6_E1**

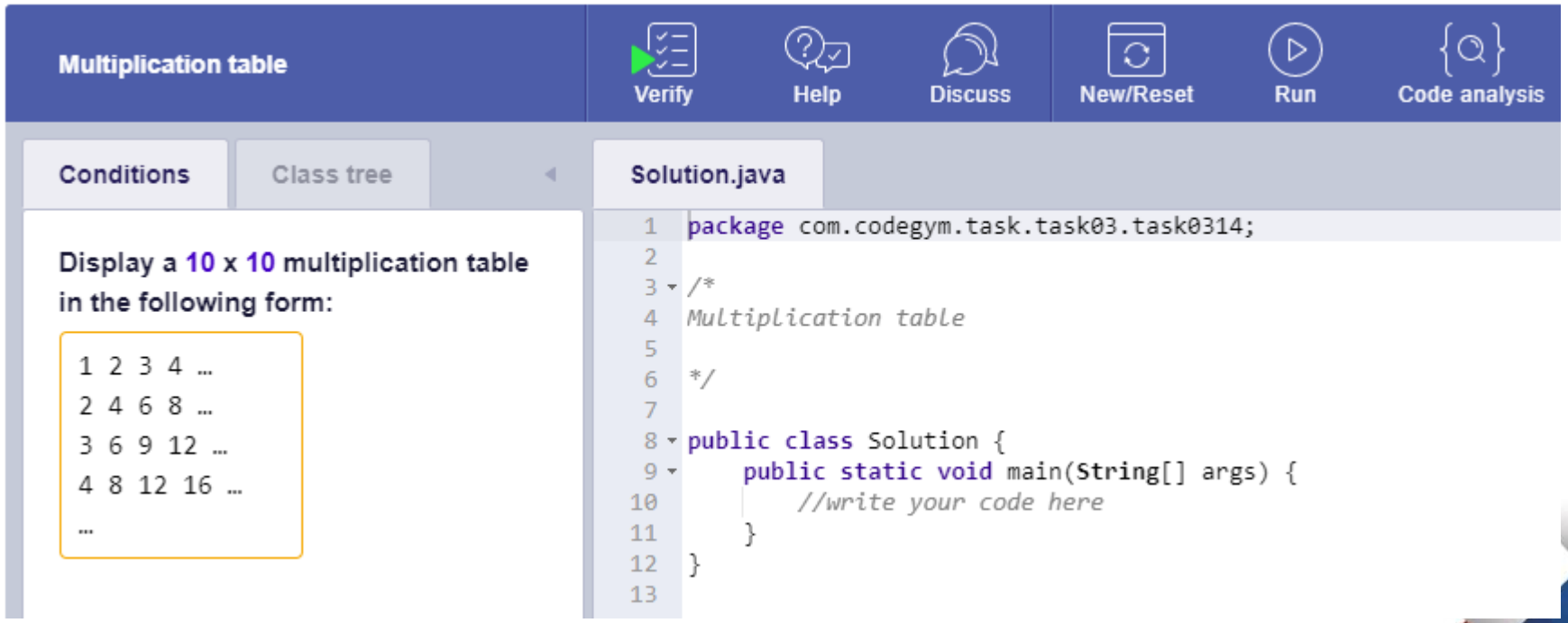


¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 3 - Lesson 6 / Nivel 3 – Lección 6

Excercise: **N3_L6_E2**



¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 3 - Lesson 6 / Nivel 3 – Lección 6

Excercise: **N3_L6_E3**

Roy G. Biv...

Verify | Help | Discuss | New/Reset | Run | Code analysis

Conditions | Class tree

Create **7 objects** to display the **7 colors** of the rainbow on the screen.
Example output:
Red
Orange
Yellow
Green
Blue
Indigo
Violet
Each object displays a specific color when it is created.

Requirements:

- The program should output text.
- You need to create 7 different objects responsible for the colors.
- The order in which the objects are created must correspond to the order of the colors in the rainbow.
- Don't change the classes responsible for the colors.
- The displayed text must match the task conditions.

Solution.java

```java
package com.codegym.task.task03.task0315;

/*⇔*/

public class Solution {
    public static void main(String[] args) {
        //write your code here
    }

    public static class Red {
        public Red() {
            System.out.println("Red");
        }
    }

    public static class Orange {
        public Orange() {
            System.out.println("Orange");
        }
    }

    public static class Yellow {
        public Yellow() {
            System.out.println("Yellow");
        }
    }

    public static class Green {
        public Green() {
            System.out.println("Green");
        }
    }

    public static class Blue {
        public Blue() {
            System.out.println("Blue");
        }
    }

    public static class Indigo {
        public Indigo() {
            System.out.println("Indigo");
        }
    }

    public static class Violet {
        public Violet() {
            System.out.println("Violet");
        }
    }
}
```

Blue
Indigo
Violet
Each object displays a specific color when it is created.

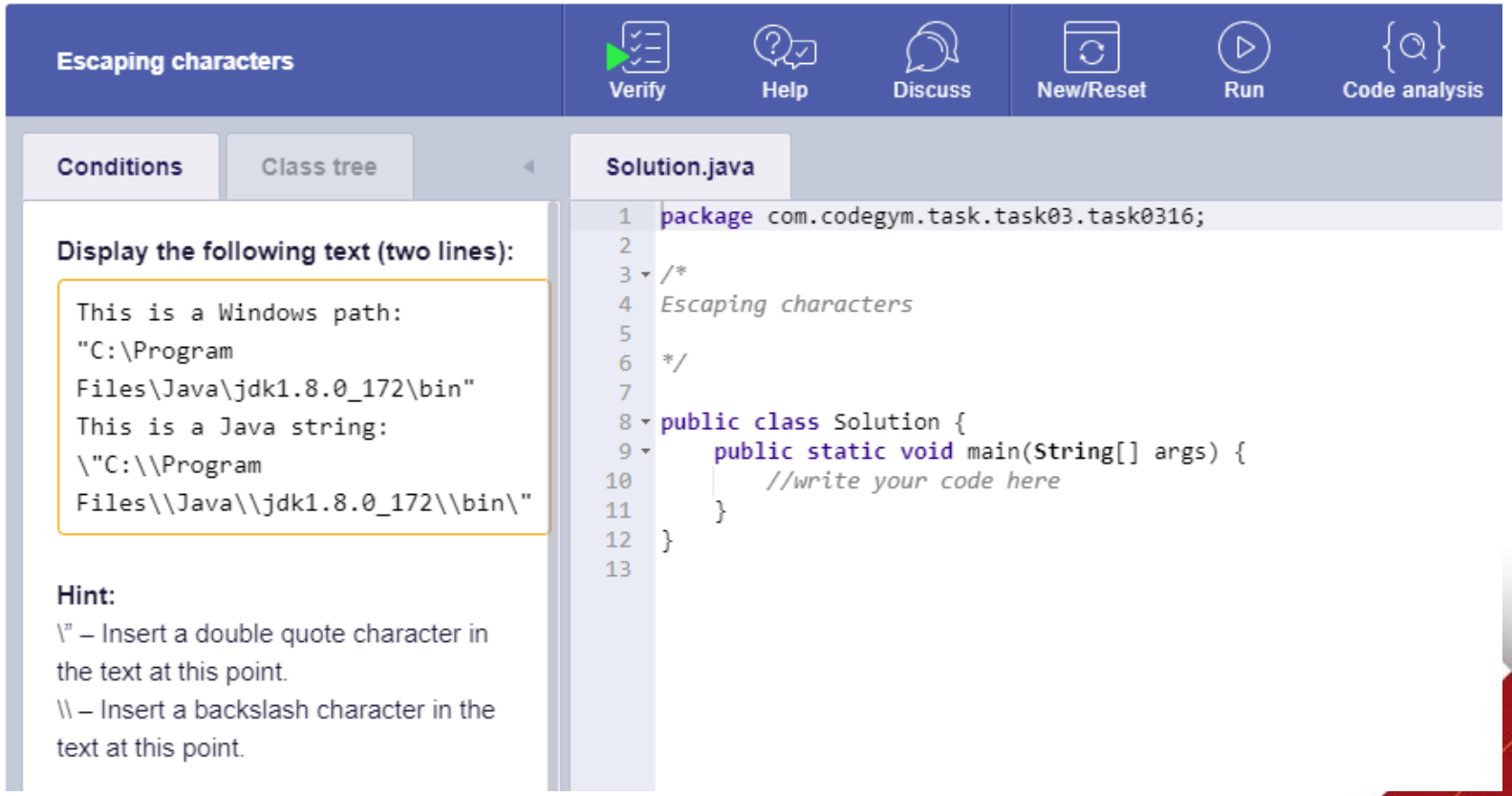
Requirements:

- The program should output text.
- You need to create 7 different objects responsible for the colors.
- The order in which the objects are created must correspond to the order of the colors in the rainbow.
- Don't change the classes responsible for the colors.
- The displayed text must match the task conditions.

Gamified platform work / Trabajo en plataforma gamificada.

Level 3 - Lesson 6 / Nivel 3 – Lección 6

Excercise: **N3_L6_E4**

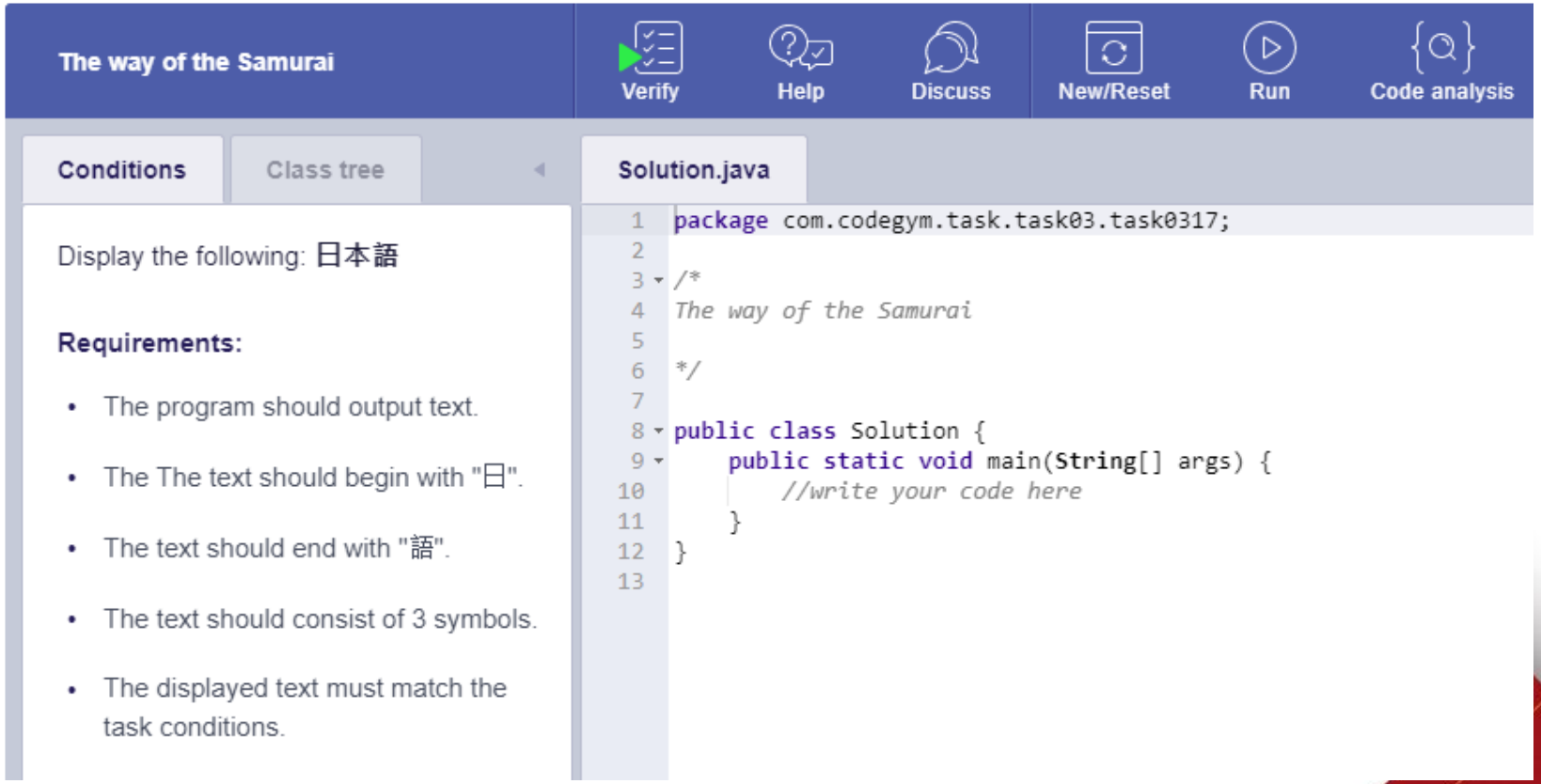


¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 3 - Lesson 6 / Nivel 3 – Lección 6

Excercise: **N3_L6_E5**



¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 3 - Lesson 7 / Nivel 3 – Lección 7

Keyboard input

Java Syntax
Level 3, Lesson 7

AVAILABLE

"Amigo, your time has come. I'm now going to tell you about keyboard input."

"We've used **System.out** to display data on the screen. To receive input, we'll use **System.in**."

"Sounds easy."

"But **System.in** has one shortcoming – it only lets us read character codes from the keyboard. To get around this problem and read big chunks of data all at once, we'll use a more complex construct:"

¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 3 - Lesson 7 / Nivel 3 – Lección 7

Example 1
Input a string and number from the keyboard

```java
InputStream inputStream = System.in;
Reader inputStreamReader = new InputStreamReader(inputStream);
BufferedReader bufferedReader = new BufferedReader(inputStreamReader);

String name = bufferedReader.readLine(); //Read a string from the keyboard
String sAge = bufferedReader.readLine(); //Read a string from the keyboard
int nAge = Integer.parseInt(sAge); //Convert the string to a number.
```

Example 2
A more compact version of the previous example:

```java
BufferedReader reader = new BufferedReader(new InputStreamReader(System.in));

String name = reader.readLine();
String sAge = reader.readLine();
int nAge = Integer.parseInt(sAge);
```

Gamified platform work / Trabajo en plataforma gamificada.

Level 3 - Lesson 7 / Nivel 3 – Lección 7

Example 3
Even more compact

```
1  Scanner scanner = new Scanner(System.in);
2  String name = scanner.nextLine();
3  int age = scanner.nextInt();
```

Gamified platform work / Trabajo en plataforma gamificada.

Level 3 - Lesson 8 / Nivel 3 – Lección 8

Enter a number

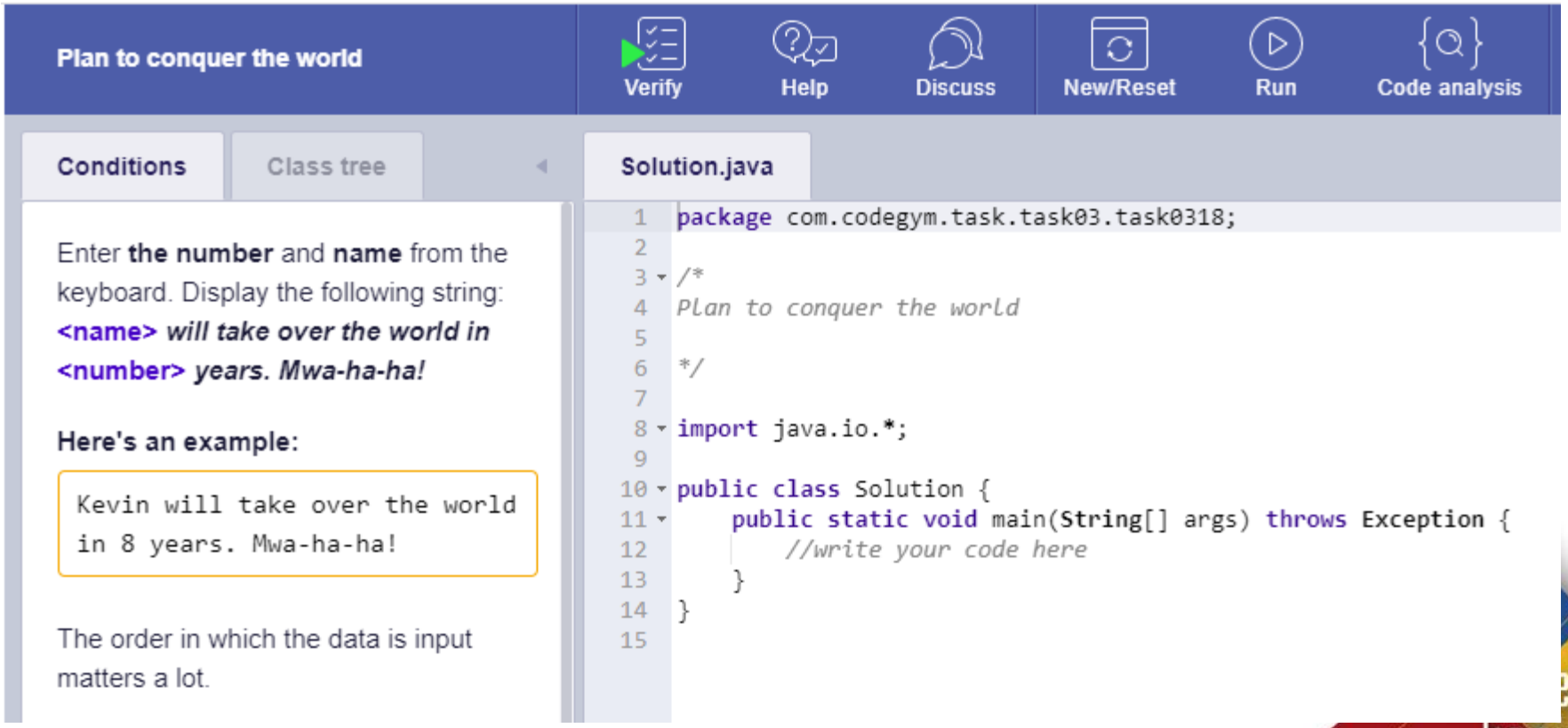
Java Syntax
Level 3, Lesson 8

AVAILABLE

Gamified platform work / Trabajo en plataforma gamificada.

Level 3 - Lesson 8 / Nivel 3 – Lección 8

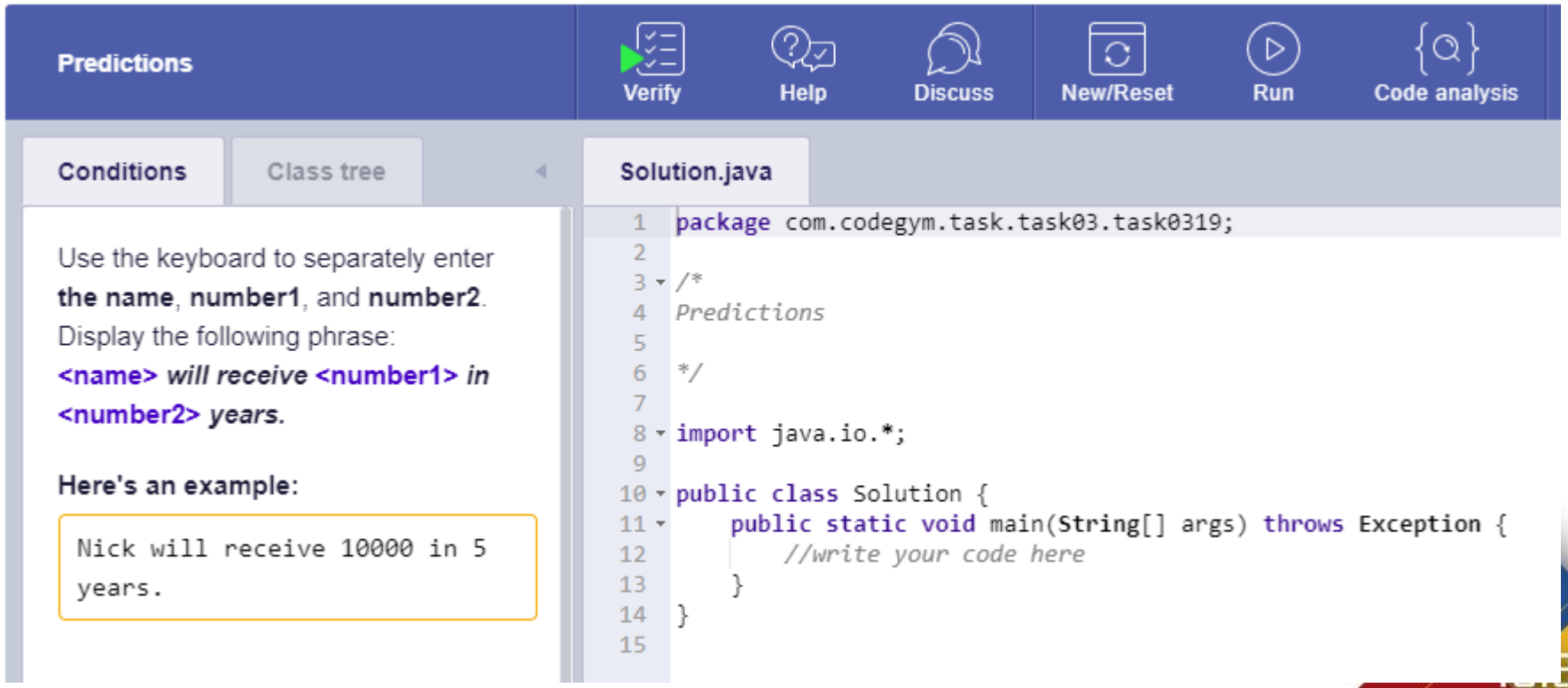
Excercise: **N3_L8_E1**



Gamified platform work / Trabajo en plataforma gamificada.

Level 3 - Lesson 8 / Nivel 3 – Lección 8

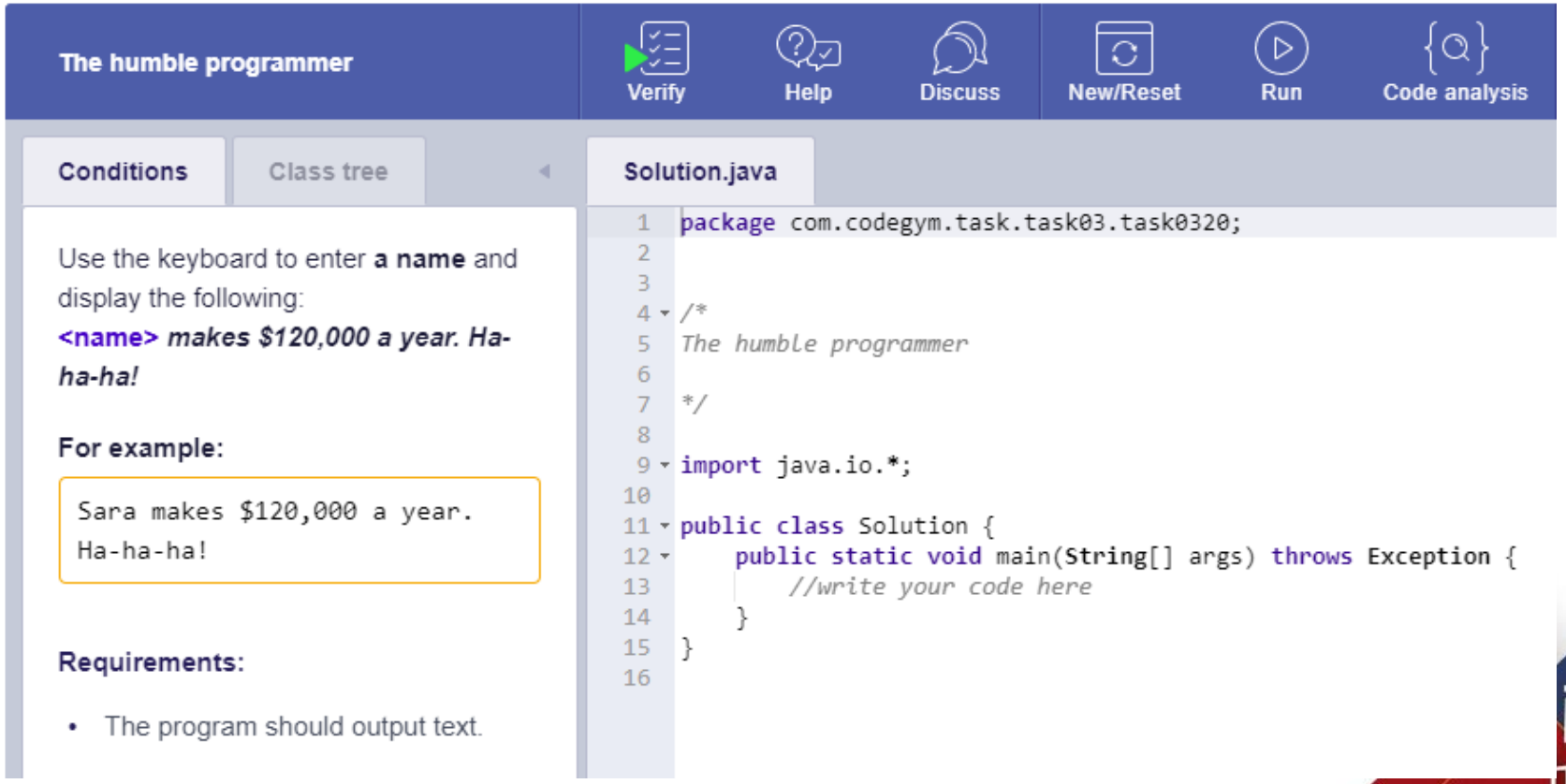
Excercise: **N3_L8_E2**



Gamified platform work / Trabajo en plataforma gamificada.

Level 3 - Lesson 8 / Nivel 3 – Lección 8

Excercise: **N3_L8_E3**

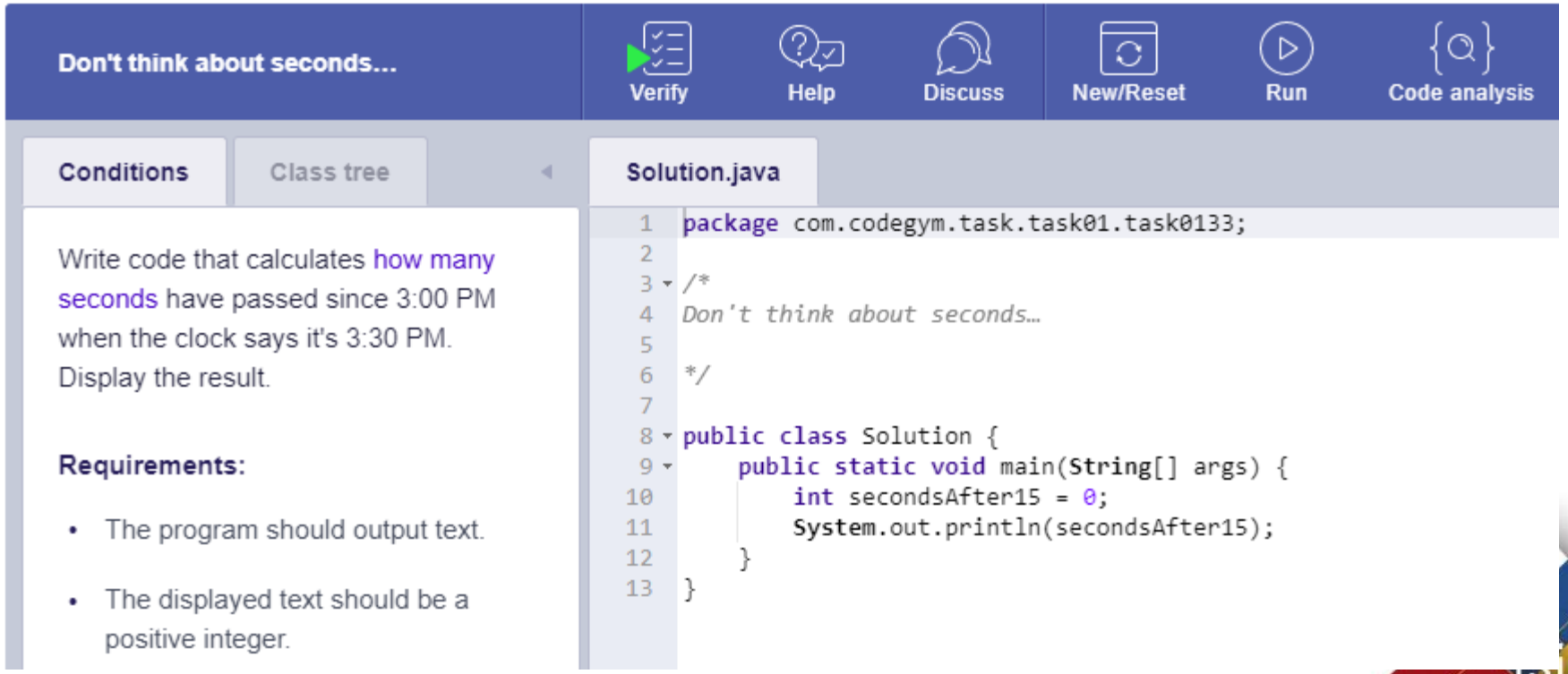


¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 3 - Lesson 8 / Nivel 3 – Lección 8

Excercise: **N3_L8_E4**

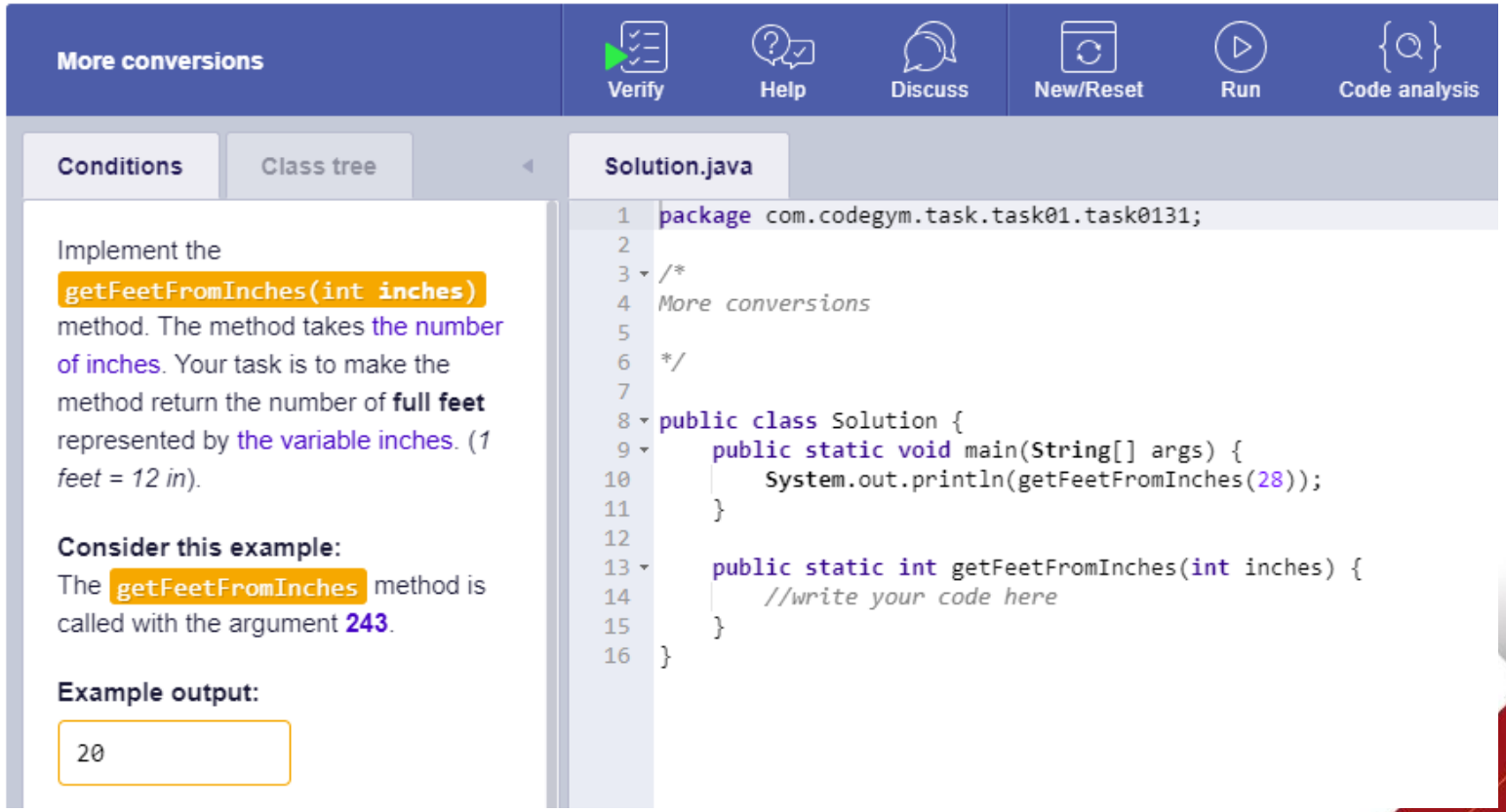


siempre
hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 3 - Lesson 8 / Nivel 3 – Lección 8

Excercise: **N3_L8_E5**



¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 3 - Lesson 8 / Nivel 3 – Lección 8

Excercise: **N3_L8_E6**

Deep and pure love

Verify | Help | Discuss | New/Reset | Run | Code analysis

Conditions | Class tree

Use the keyboard to enter **three names**, then display:
name1 + **name2** + **name3** = ***Pure love. Ooo la-la!***

For example:

```
Kevin + Eva + Angelica = Pure love. Ooo la-la!
```

Requirements:

- The program should output text.

Solution.java

```
1  package com.codegym.task.task03.task0322;
2
3
4  /*
5  Deep and pure love
6
7  */
8
9  import java.io.*;
10
11 public class Solution {
12     public static void main(String[] args) throws Exception {
13         //write your code here
14     }
15 }
```

Gamified platform work / Trabajo en plataforma gamificada.

Level 3 - Lesson 9 / Nivel 3 – Lección 9

IntelliJ and creating your own project

Java Syntax
Level 3, Lesson 9

AVAILABLE

"Hi, Amigo! Are you making progress?"

"Hello, Professor."

"Everything's great. Today I installed the JDK and IntelliJ IDEA on Diego's advice. Then, on Kim's advice, I downloaded a project and plugin for doing tasks. I'm now trying to sort out how to use it all."

"I'll help you. I believe that I know the best way to teach you how to create applications. Let's create our own project, separate from the project for CodeGym tasks. Seeing once is better than hearing a hundred times. Here's a video for you:"

Gamified platform work / Trabajo en plataforma gamificada.

Level 3 - Lesson 10 / Nivel 3 – Lección 10

Useful Links from the Professor – 3

Java Syntax
Level 3, Lesson 10

AVAILABLE

"Well, hello, Amigo! I trust you understood everything that Ellie and Rishi told you today."

"But even if you do, repeating and reinforcing class material never hurt anybody. These activities usually not only solidify your knowledge, but also expand your awareness!"

"But I'm getting carried away. Forget about what I just said. Instead, here are a few useful links that will help you to dive deeper into and review the material from the third level. They may even help you understand something that you didn't before."

Reading from the keyboard with readers

"This topic isn't that complicated, but it can be muddy. Beginners often find it confusing due to the abundance of incomprehensible words. If you misunderstood something or want to reinforce your knowledge, read this article. It'll give a little boost to your understanding of reading from the keyboard. For example, by improving your understanding of what a stream is."

Gamified platform work / Trabajo en plataforma gamificada.

Level 3 - Lesson 11 / Nivel 3 – Lección 11

How do hard drives work?

Java Syntax
Level 3, Lesson 11

AVAILABLE

"Hi, Amigo! The time has come to relax a bit. How about watching a good video?"

Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 3 - Lesson 12 / Nivel 3 – Lección 12

Additional tasks

Java Syntax
Level 3, Lesson 12

AVAILABLE

¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 3 - Lesson 12 / Nivel 3 – Lección 12

Excercise: **N3_L12_E1**

Sum of the digits of a three-digit number

Verify | Help | Discuss | New/Reset | Run | Code analysis

Conditions | Class tree

Write the code for `sumDigitsInNumber(int number)`. The method **takes a three-digit whole number**. You need to calculate **the sum of the digits of this number**, and then return the result.

Consider this example:

The `sumDigitsInNumber` method is called with the argument **546**.

Example output:

```
15
```

Solution.java

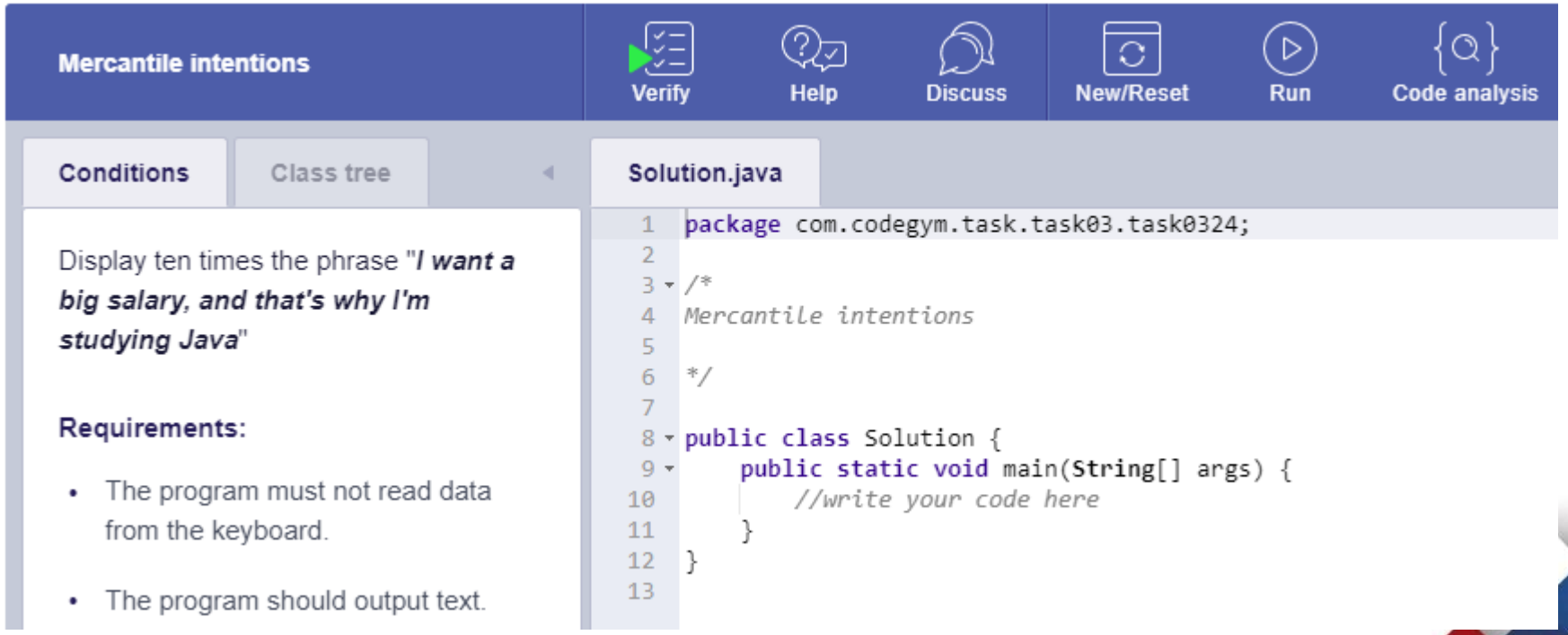
```java
package com.codegym.task.task01.task0132;

/*
Sum of the digits of a three-digit number

*/

public class Solution {
    public static void main(String[] args) {
        System.out.println(sumDigitsInNumber(546));
    }

    public static int sumDigitsInNumber(int number) {
        //write your code here
    }
}
```

¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 3 - Lesson 12 / Nivel 3 – Lección 12

Excercise: **N3_L12_E2**

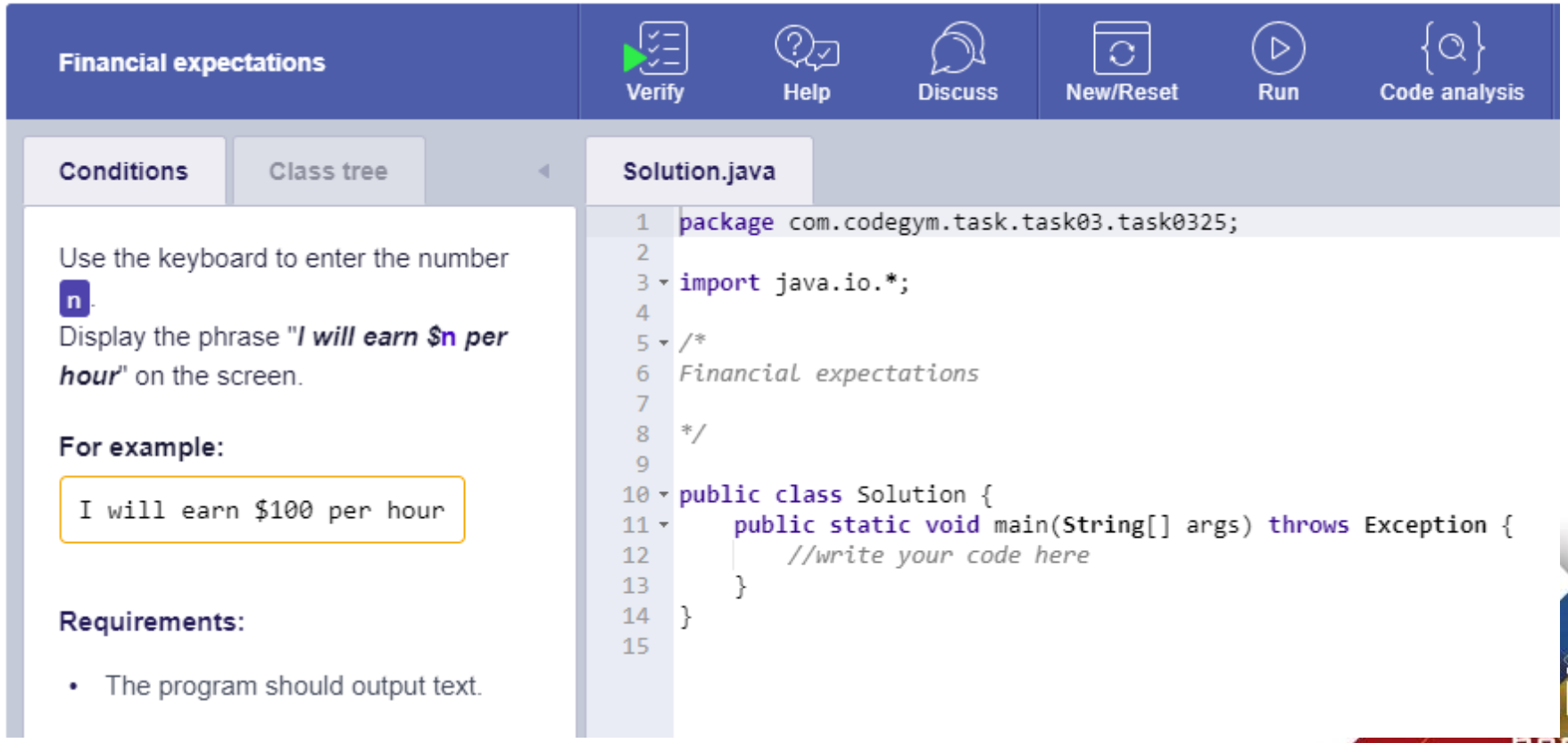


¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

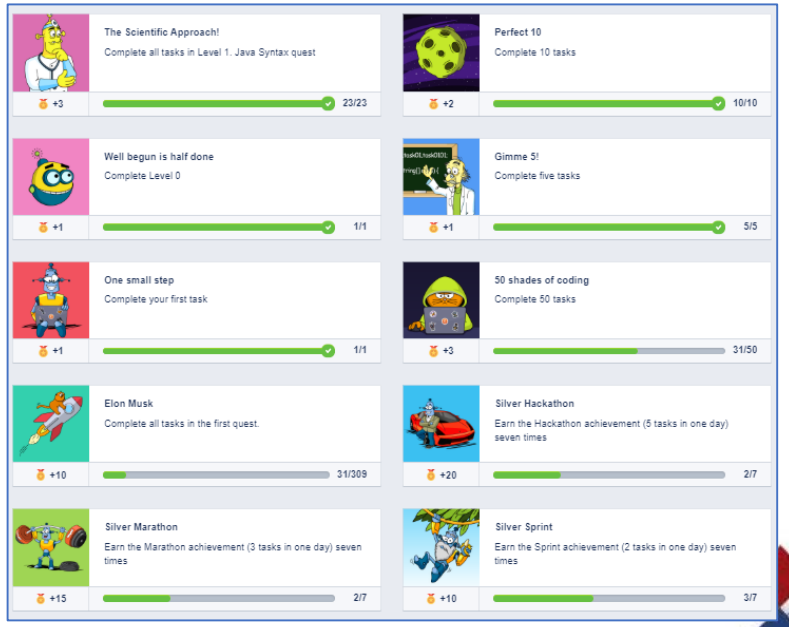
Level 3 - Lesson 12 / Nivel 3 – Lección 12

Excercise: **N3_L12_E3**



Working document with full level 3! / Documento de trabajo con nivel 3 completo!

Do not forget screenshot of the achievement board / No olvides pantallazo del tablero de logros



Upload it to your drive folder: /

Súbelo a tu carpeta en drive.

¡Siempre hacia lo alto!

Feedback

¡Siempre hacia lo alto!

Survey

¡Siempre hacia lo alto!

Clarification of doubts

¡Siempre hacia lo alto!

To practice!

¡Siempre hacia lo alto!

P1TX: Exercise

Create ...

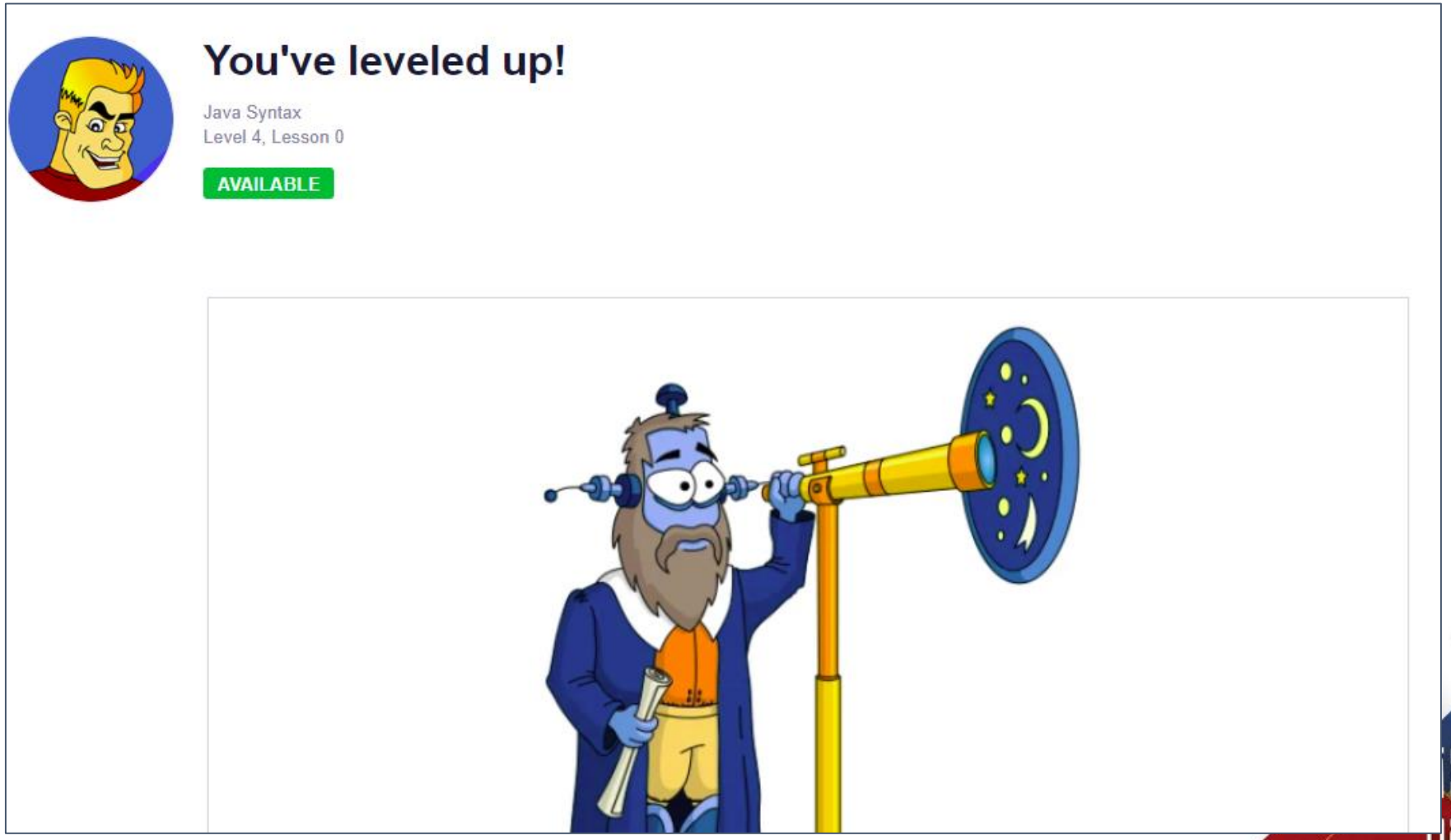
Crear...

7° Session CodeGym

¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 0 / Nivel 4 – Lección 0



Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 1 / Nivel 4 – Lección 1

More about variable scope

Java Syntax
Level 4, Lesson 1

AVAILABLE

"The Professor just can't get out of his rut. Old teachers who are used to lecturing are always like this. There's nothing he can tell you that you can't find in books. You don't learn how to swim by listening to swimming lectures. Lectures are only useful when you're familiar with the subject and know almost as much as your professor."

"Still, his lessons are useful."

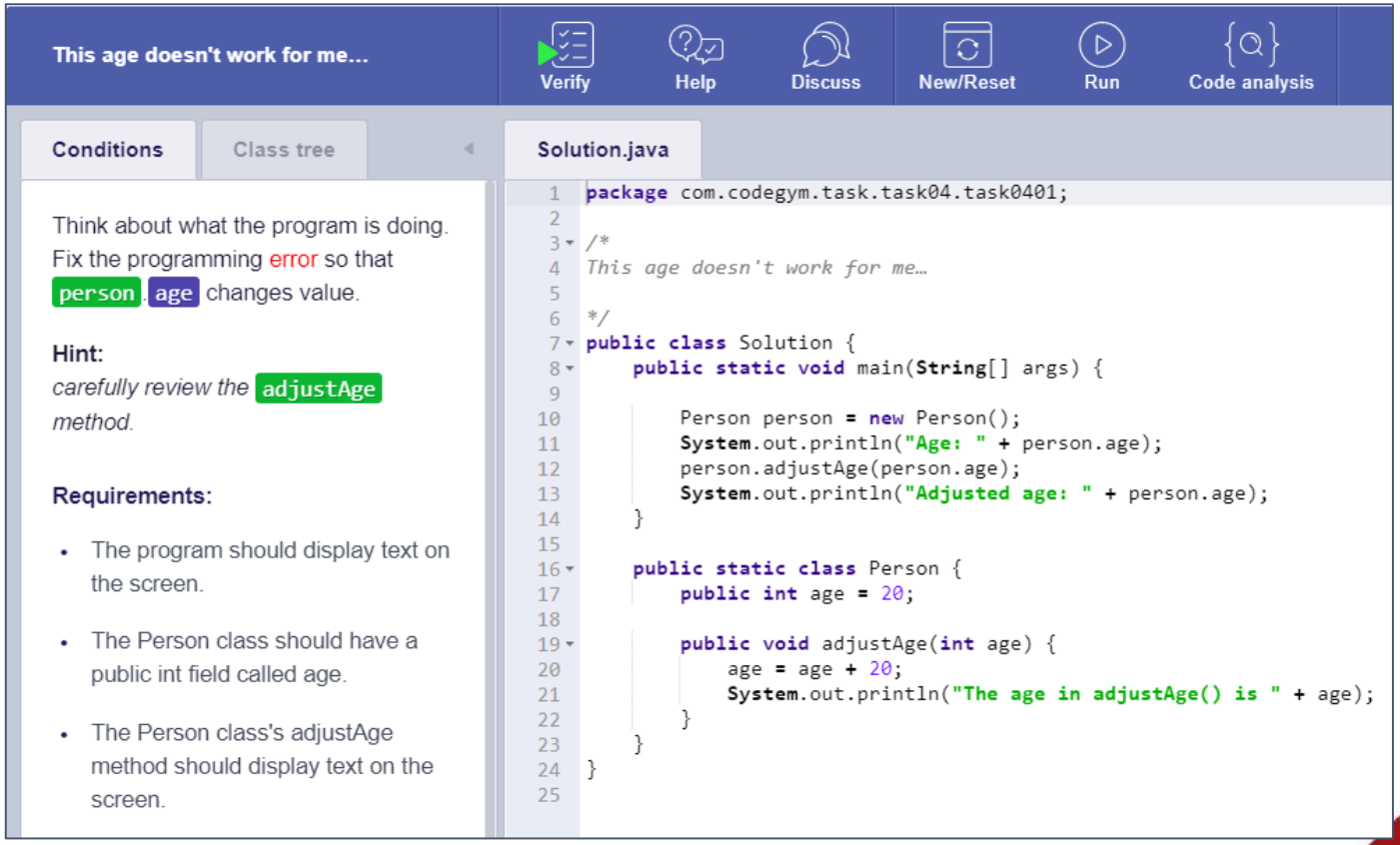
"Yep. I mean, we hope they are. The more perspectives on the subject you hear, the closer you'll get to the truth. When you hear just one, all you can do is believe it or disbelieve it. OK, let's get back to business."

"Let's look at a picture I've shown you before."

Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 1 / Nivel 4 – Lección 1

Excercise: **N4_L1_E1**



¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 1 / Nivel 4 – Lección 1

Excercise: **N4_L1_E2**

Price of apples

Verify | Help | Discuss | New/Reset | Run | Code analysis

Conditions | Class tree

Calculate the total cost of apples.
The total cost of apples corresponds to
public static int `applePrice`.

Requirements:

- The program should display text on the screen.
- The Apple class's addPrice method should not display text on the screen.
- The Apple class's applePrice variable must be a static int initialized to zero.
- The main method should call the addPrice method only twice.
- The Apple class's addPrice method should increase the cost of apples by the passed-in value.

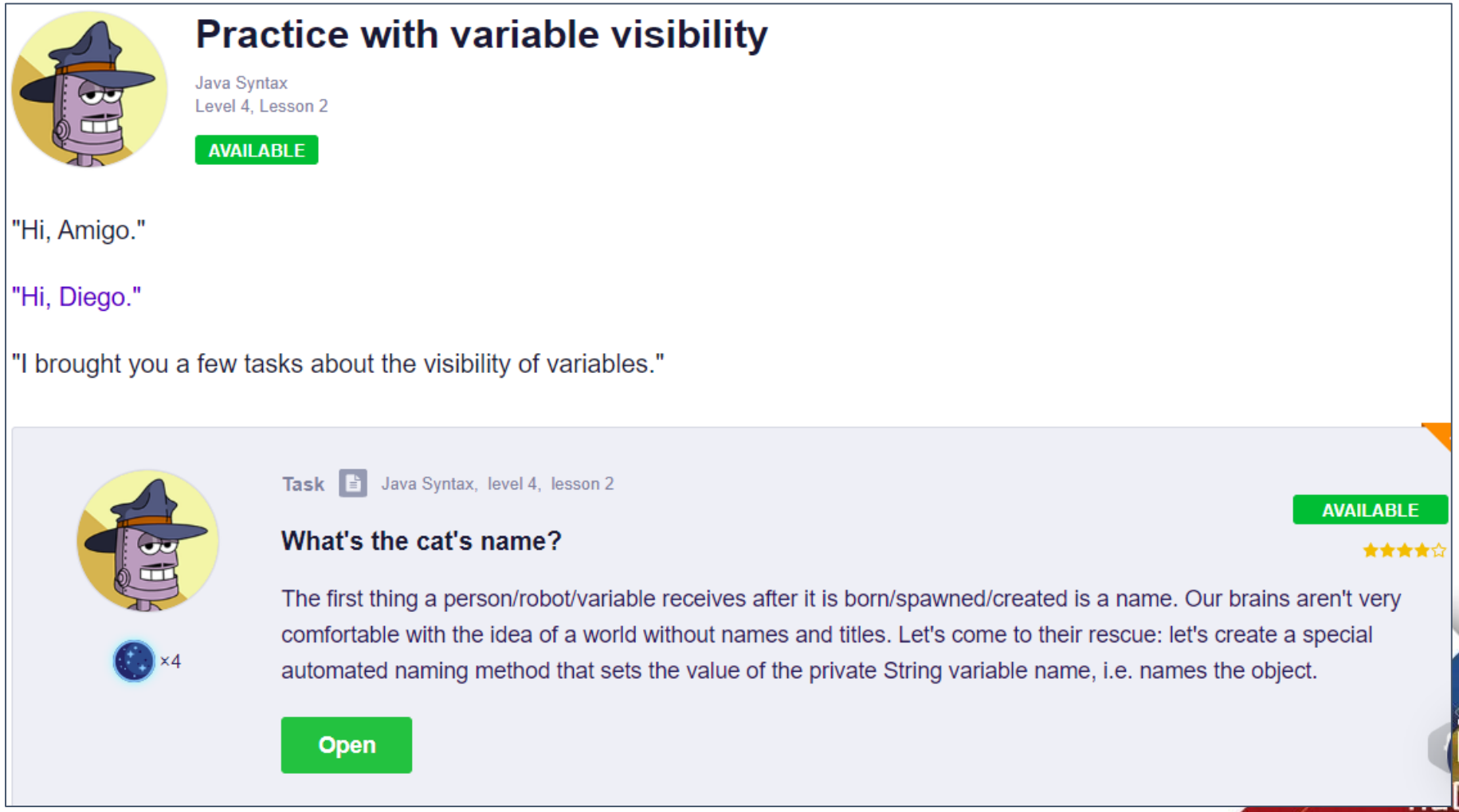
Solution.java

```
1  package com.codegym.task.task04.task0402;
2
3  /*
4  Price of apples
5
6  */
7  public class Solution {
8      public static void main(String[] args) {
9          Apple apple = new Apple();
10         apple.addPrice(50);
11         Apple apple2 = new Apple();
12         apple2.addPrice(100);
13         System.out.println("The cost of apples is " + Apple.applePrice);
14     }
15
16     public static class Apple {
17         public static int applePrice = 0;
18
19         public static void addPrice(int applePrice) {
20             //write your code here
21         }
22     }
23 }
24
```

¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 2 / Nivel 4 – Lección 2



Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 2 / Nivel 4 – Lección 2

Excercise: **N4_L2_E1**

What's the cat's name?

Verify | Help | Discuss | New/Reset | Run | Code analysis

Conditions | Class tree

Help the cat get a name using the `setName` method.

Requirements:

- The Cat class must contain only one variable called name.
- The variable name must be a String and have a private access modifier.
- The Cat class must have only two methods: setName and main.
- The Cat class's setName method must set the value of the private String variable name equal to the passed String argument name.

Cat.java

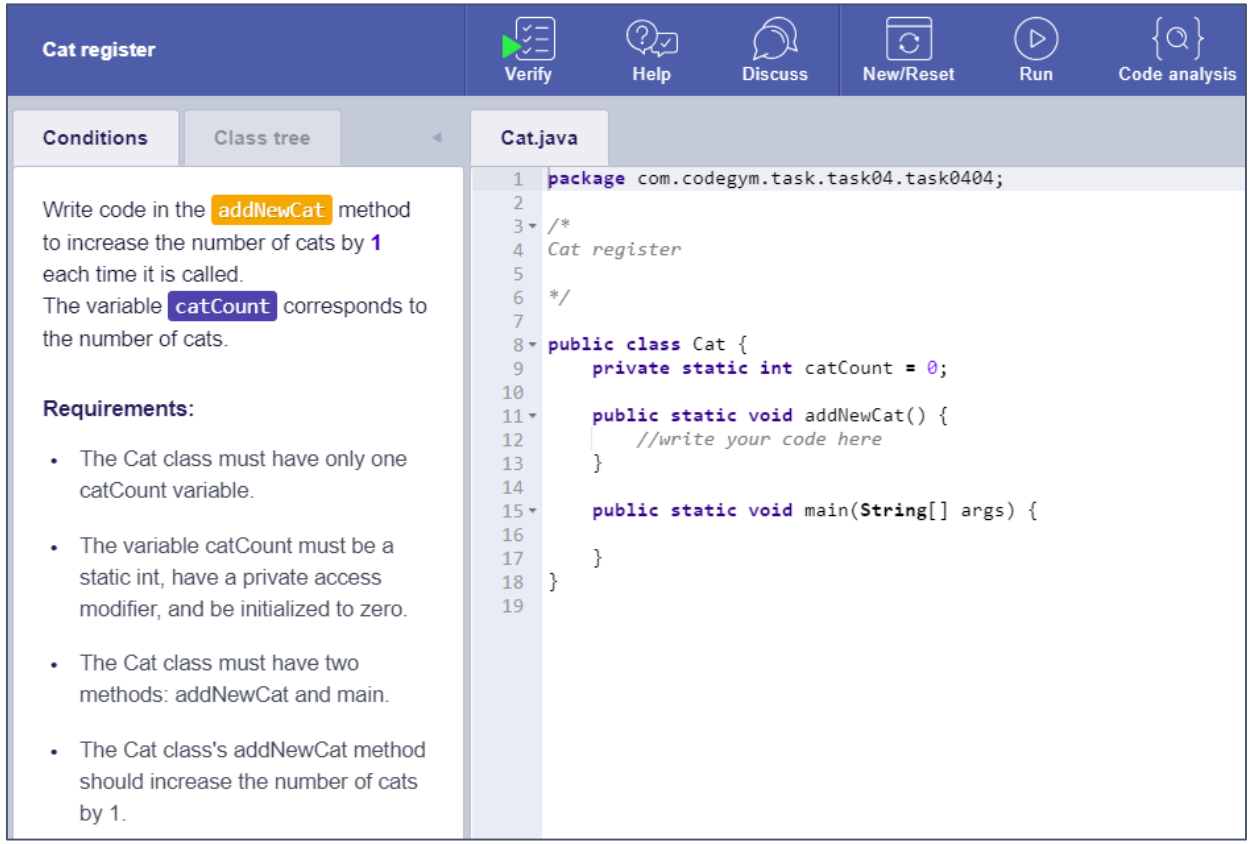
```
1  package com.codegym.task.task04.task0403;
2
3  /*
4  What's the cat's name?
5
6  */
7
8  public class Cat {
9      private String name = "nameless cat";
10
11     public void setName(String name) {
12         //write your code here
13     }
14
15     public static void main(String[] args) {
16         Cat cat = new Cat();
17         cat.setName("Simba");
18         System.out.println(cat.name);
19     }
20 }
21
```

¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 2 / Nivel 4 – Lección 2

Excercise: **N4_L2_E2**



¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 2 / Nivel 4 – Lección 2

Excercise: **N4_L2_E3**

Setting the number of cats

Verify | Help | Discuss | New/Reset | Run | Code analysis

Conditions | Class tree

Write the `setCatCount` method. The method must set the number of cats (`catCount`).

Requirements:

- The Cat class must have only one catCount variable.
- The variable catCount must be a static int, have a private access modifier, and be initialized to zero.
- The Cat class must have two methods: setCatCount and main.
- The Cat class's setCatCount method must set the value of the catCount variable equal to the passed argument.

Cat.java

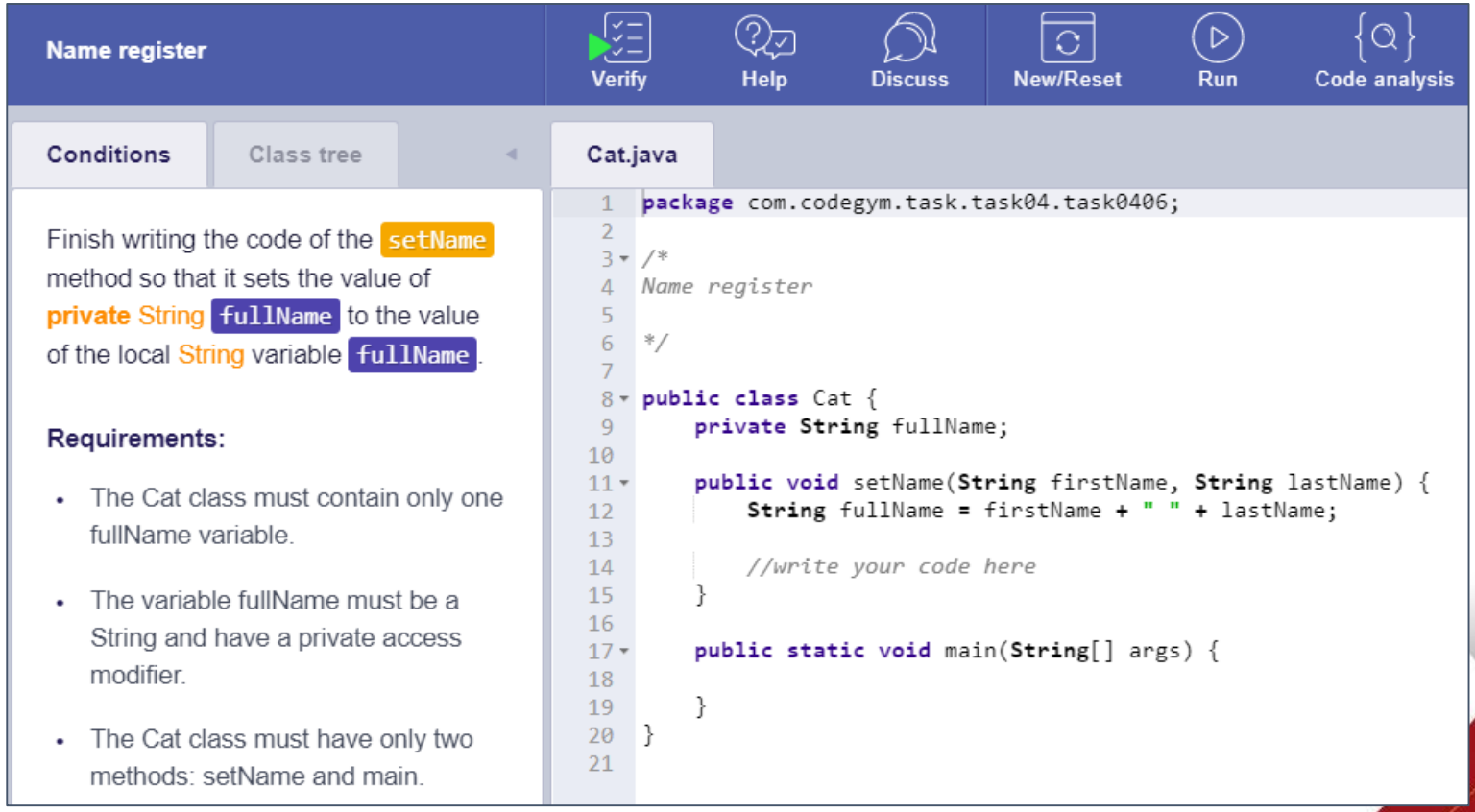
```
1  package com.codegym.task.task04.task0405;
2
3  /*
4  Setting the number of cats
5
6  */
7
8  public class Cat {
9      private static int catCount = 0;
10
11     public static void setCatCount(int catCount) {
12         //write your code here
13     }
14
15     public static void main(String[] args) {
16
17     }
18 }
19
```

¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 2 / Nivel 4 – Lección 2

Excercise: **N4_L2_E4**



¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 2 / Nivel 4 – Lección 2

Excercise: **N4_L2_E5**

Cats in the Universe

Verify | Help | Discuss | New/Reset | Run | Code analysis

Conditions | Class tree

Write code to correctly count **the number** of cats created (*count*) and **display** the correct number of cats on the screen.

Requirements:

- The program should display text on the screen.
- Don't change the line responsible for screen output.
- The Cat class must have only one count variable.
- The Cat class's variable count must be a static int, have a public access modifier, and be initialized to zero.

Solution.java

```
1  package com.codegym.task.task04.task0407;
2
3  /*
4  Cats in the Universe
5
6  */
7
8  public class Solution {
9      public static void main(String[] args) {
10         Cat cat1 = new Cat();
11         //write your code here
12
13         Cat cat2 = new Cat();
14         //write your code here
15
16         System.out.println("The cat count is " + Cat.count);
17     }
18
19     public static class Cat {
20         public static int count = 0;
21     }
22 }
```

¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 3 / Nivel 4 – Lección 3

Commands and code blocks

Java Syntax
Level 4, Lesson 3

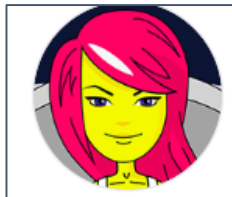
AVAILABLE

"Let me tell you about commands (statements) and code blocks. This is really simple stuff. A method body consists of commands, or statements. Each command ends in a semicolon."

	Examples of commands:
1	`String s = "Name";`
2	`System.out.println(1234);`
3	`return a + b * c;`
4	`throw new RuntimeException();`
5	`;`

Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 4 / Nivel 4 – Lección 4



Conditional operators

Java Syntax
Level 4, Lesson 4

AVAILABLE

"Hi, Amigo. Today we'll talk about **if/else statements**."

"Programs would be of little use if they didn't respond to changing external circumstances. A program needs to know how to adapt to circumstances and perform one action in one case and other actions in other cases. In Java, this is achieved using the 'if/else statement' – a special construct that makes it possible to perform different code blocks if a condition is satisfied."

"It consists of three parts: 'condition', 'command 1' and 'command 2'. If the condition is true, then 'command 1' is executed, otherwise 'command 2' is executed. These commands are never both executed. The statement looks more or less like this:"

Code for an if/else statement

```
if (condition)
    command_1;
else
    command_2;
```

Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 4 / Nivel 4 – Lección 4

Excercise: **N4_L4_E1**

Good or bad?

Verify | Help | Discuss | New/Reset | Run | Code analysis

Conditions | Class tree

Write the `compare(int a)` method so that it:
- displays "***The number is less than 5***" if the method argument is less than **5**,
- displays "***The number is greater than 5***" if the method argument is greater than **5**,
- displays "***The number is equal to 5***" if the method argument is equal to **5**.

Requirements:

- The program should display text on the screen.

Solution.java

```
1  package com.codegym.task.task04.task0408;
2
3  /*
4  Good or bad?
5
6  */
7
8  public class Solution {
9      public static void main(String[] args) {
10         compare(3);
11         compare(6);
12         compare(5);
13     }
14
15     public static void compare(int a) {
16         //write your code here
17     }
18 }
```

¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 4 / Nivel 4 – Lección 4

Excercise: **N4_L4_E2**

Closest to 10

Verify | Help | Discuss | New/Reset | Run | Code analysis

Conditions | Class tree

Write the `displayClosestToTen` method. The method should display the argument that is nearest to **10**.
For example, given the numbers **8** and **11**, **11 is closest to ten**. If both numbers are equally close to **10**, then display **either** of them.

Hint:
use the `public static int abs(int a)` *method, which returns the absolute value of a number.*

Requirements:

- The program should display text on the screen.
- The main method should not call System.out.println or System.out.print().

Solution.java

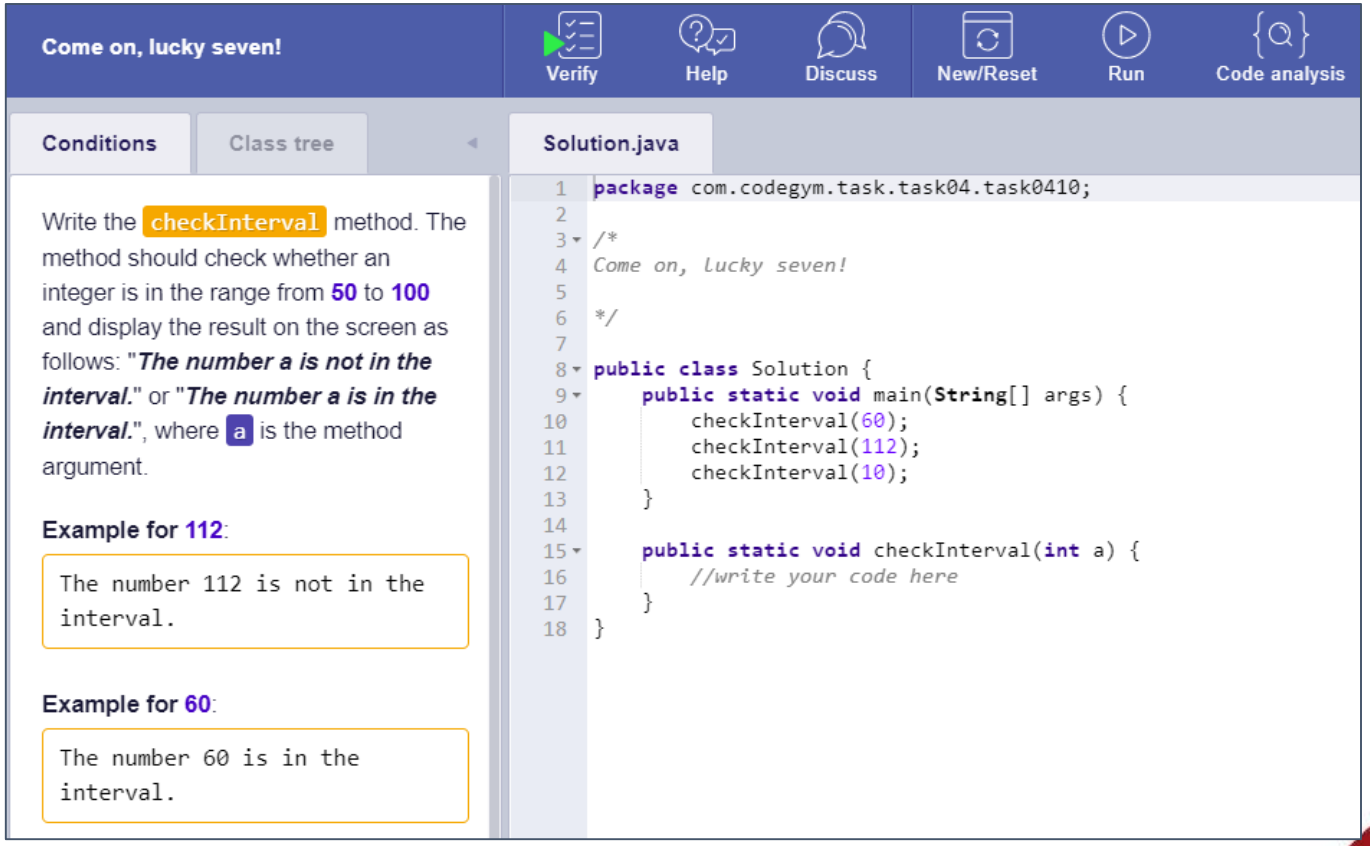
```
1  package com.codegym.task.task04.task0409;
2
3  /*
4  Closest to 10
5
6  */
7
8  public class Solution {
9      public static void main(String[] args) {
10         displayClosestToTen(8, 11);
11         displayClosestToTen(7, 14);
12     }
13
14     public static void displayClosestToTen(int a, int b) {
15         // write your code here
16
17     }
18
19     public static int abs(int a) {
20         if (a < 0) {
21             return -a;
22         } else {
23             return a;
24         }
25     }
26 }
```

¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 4 / Nivel 4 – Lección 4

Excercise: **N4_L4_E3**

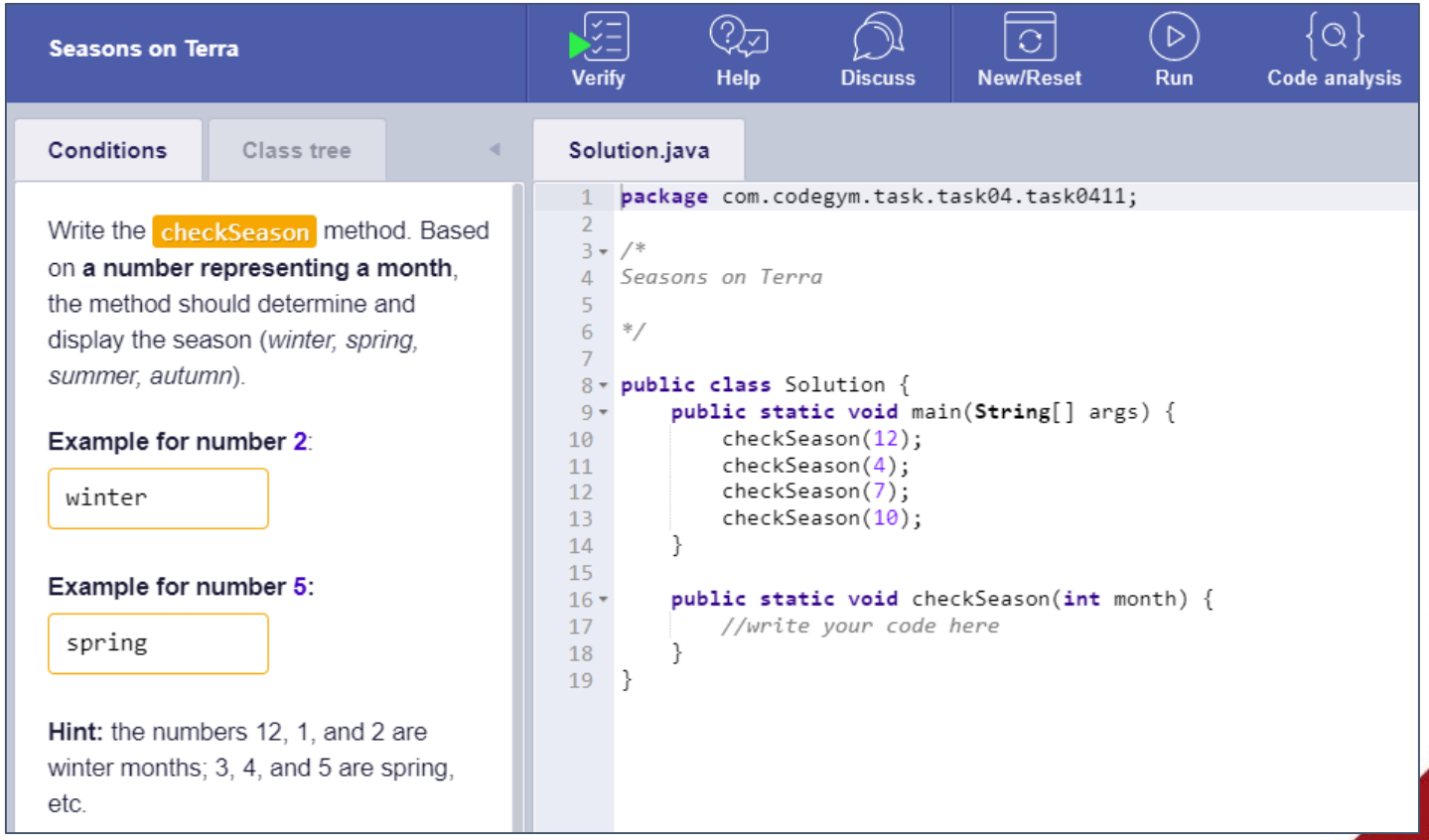


¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 4 / Nivel 4 – Lección 4

Excercise: **N4_L4_E4**

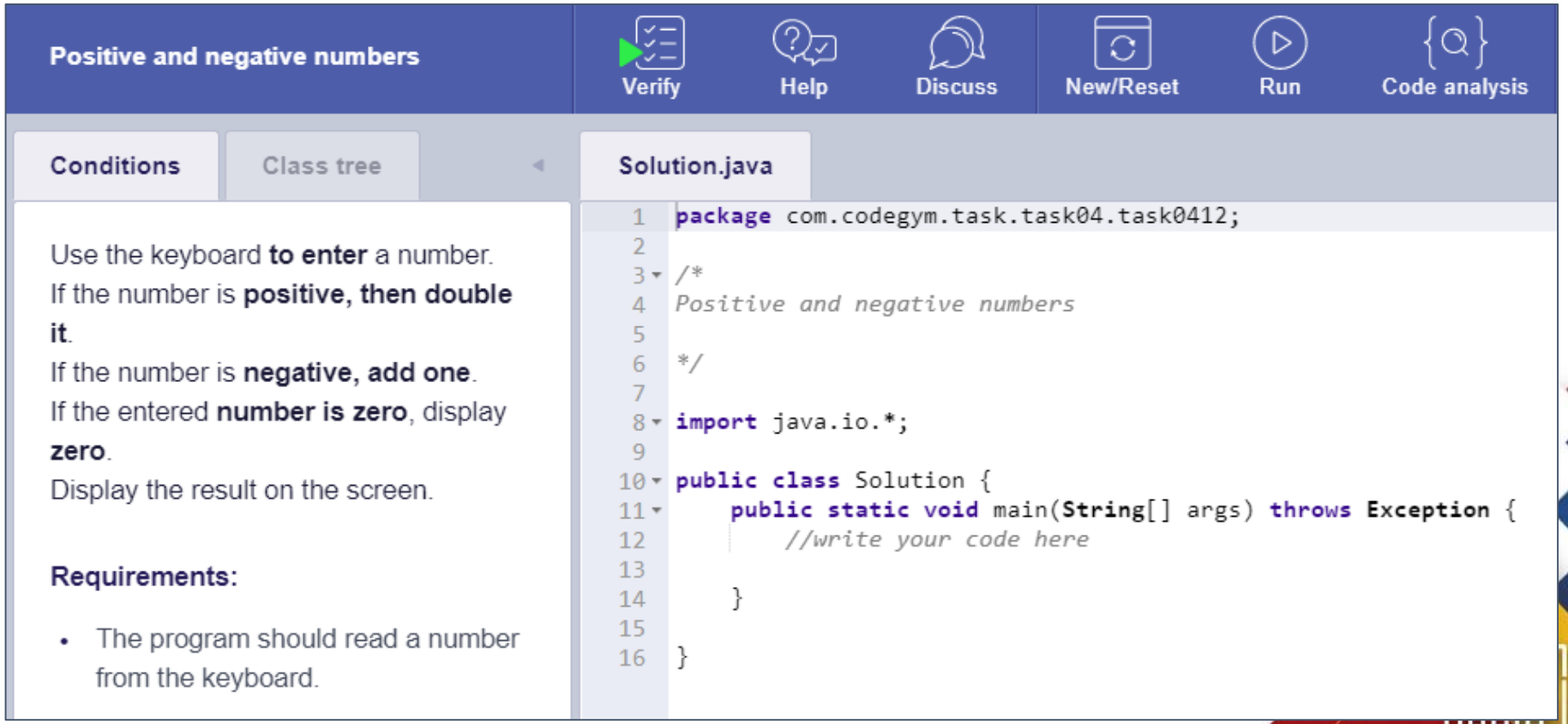


¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 4 / Nivel 4 – Lección 4

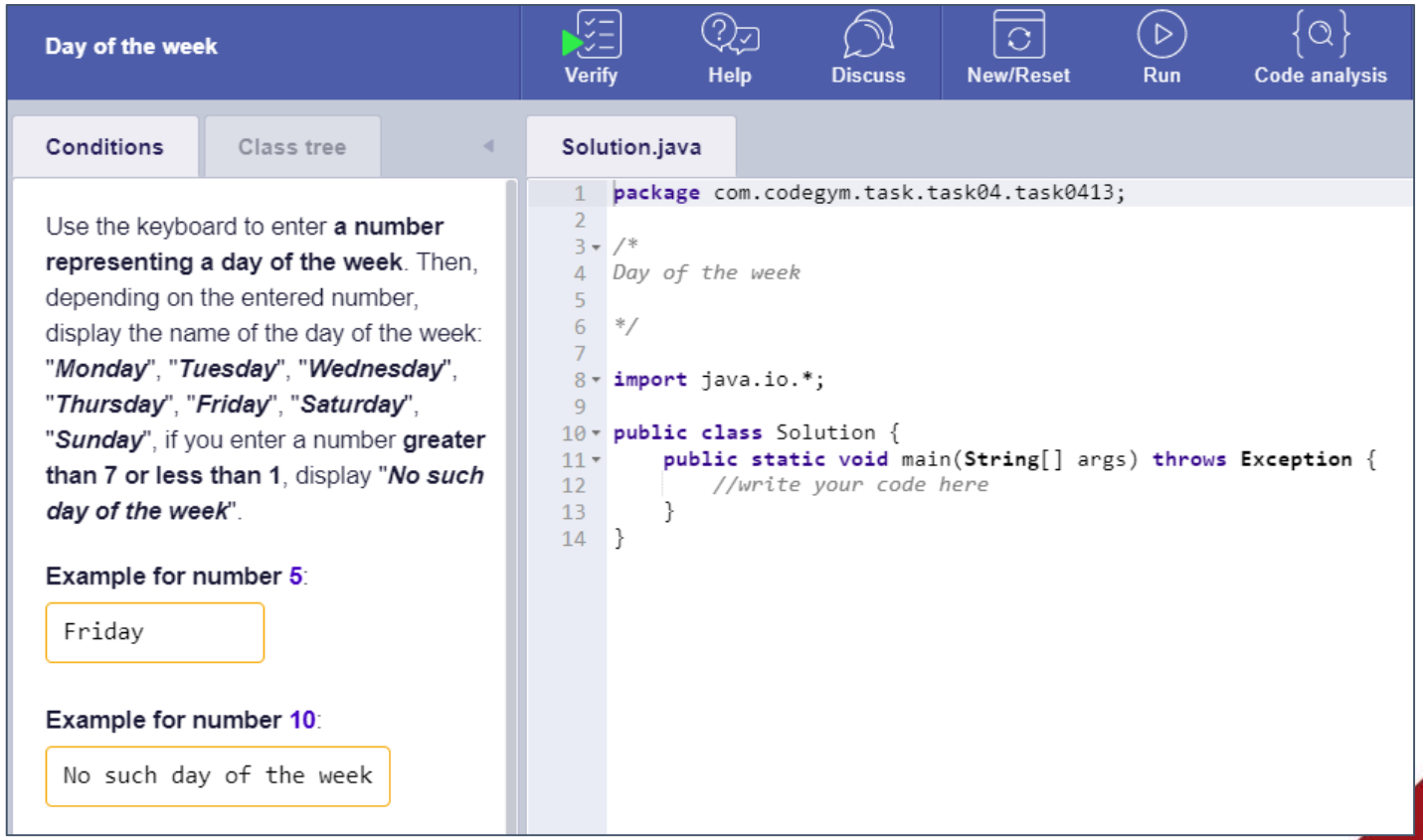
Excercise: **N4_L4_E5**



Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 4 / Nivel 4 – Lección 4

Excercise: **N4_L4_E6**

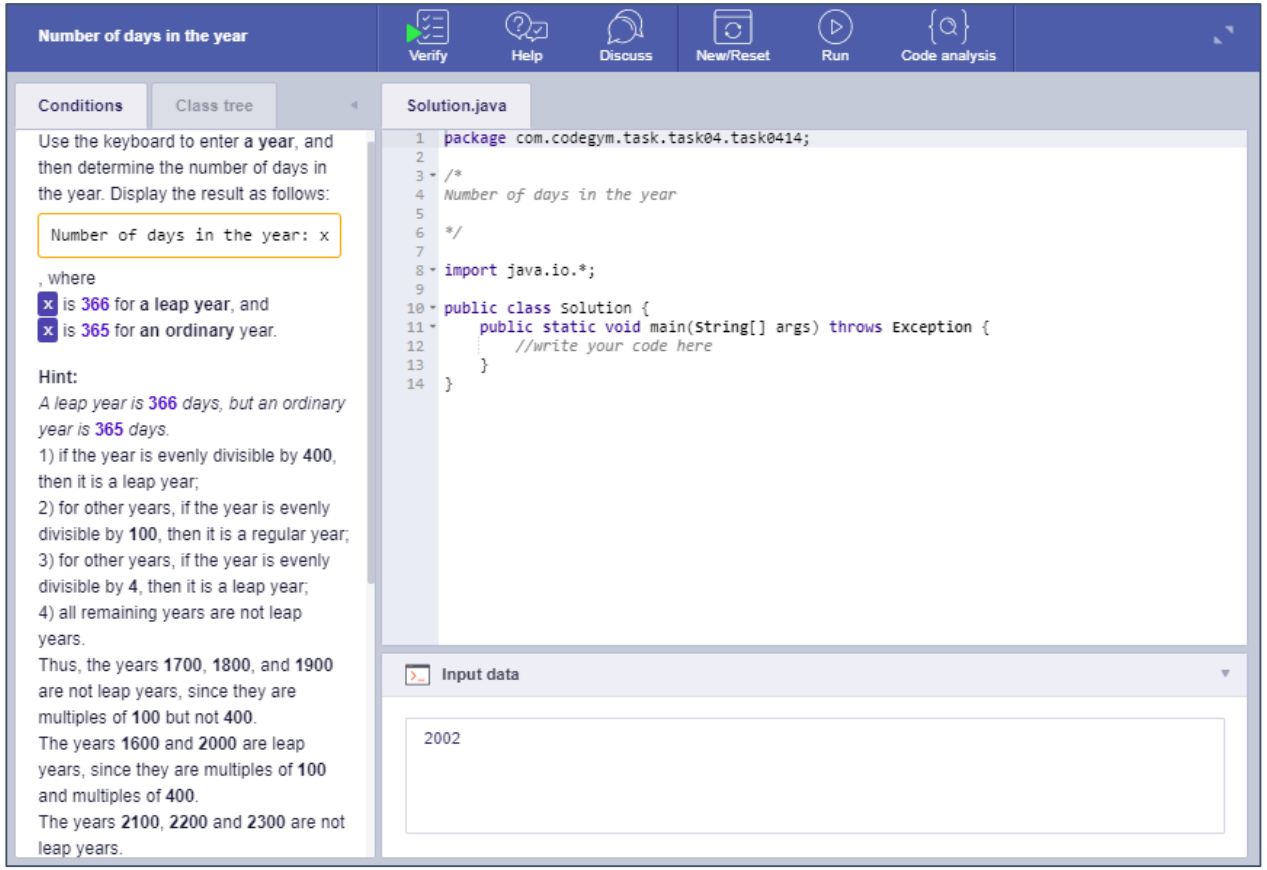


¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 4 / Nivel 4 – Lección 4

Excercise: **N4_L4_E7**

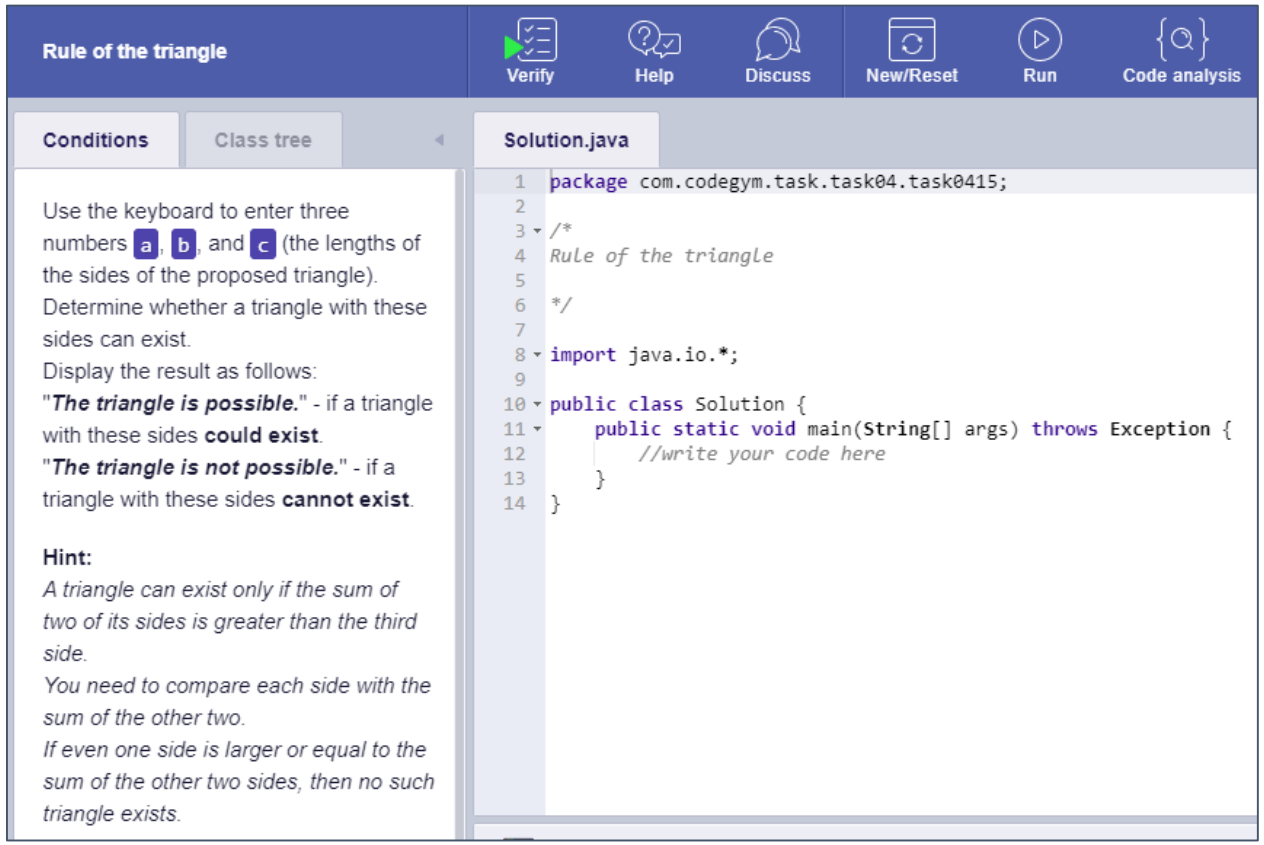


¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 4 / Nivel 4 – Lección 4

Excercise: **N4_L4_E8**

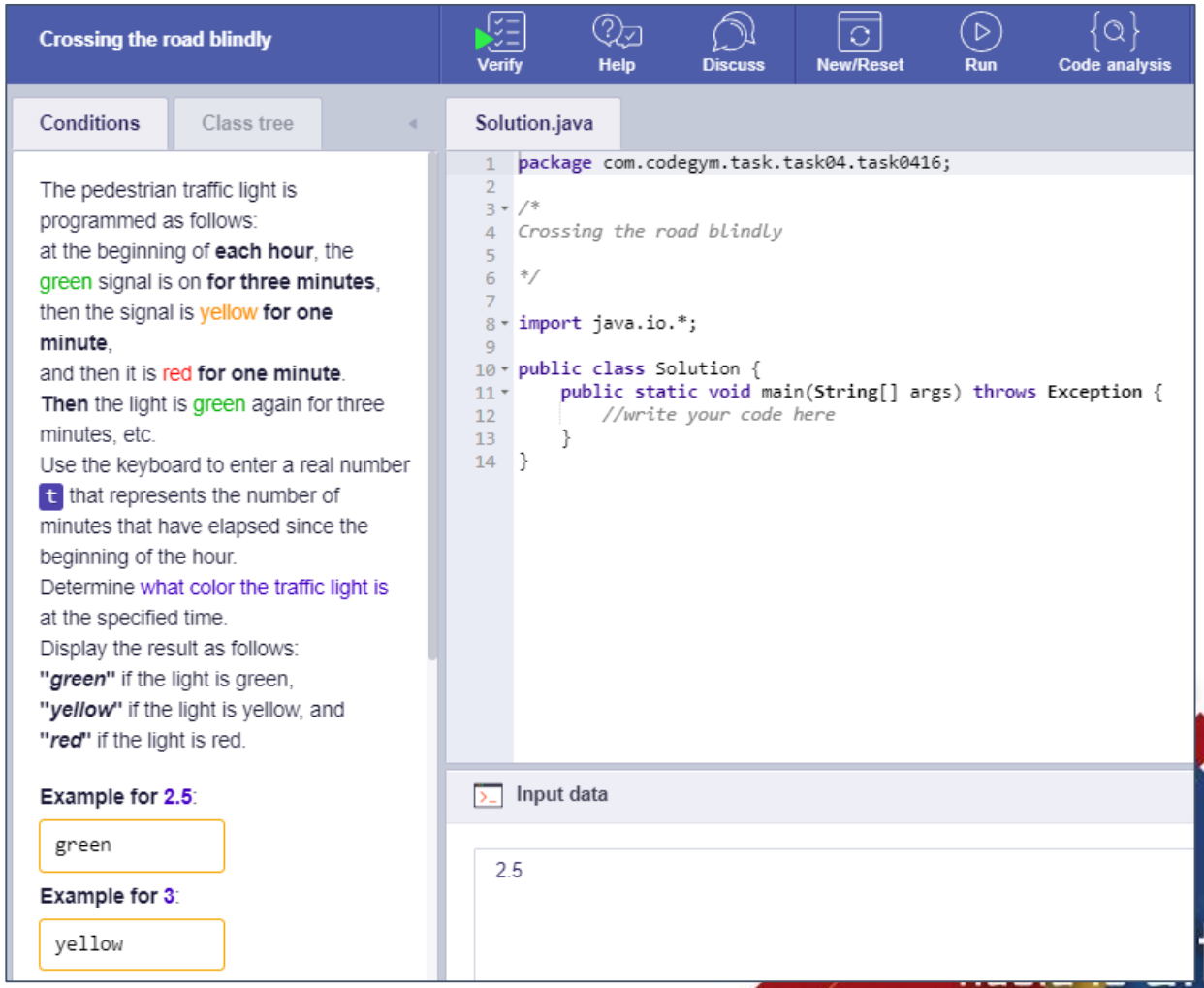


¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 4 / Nivel 4 – Lección 4

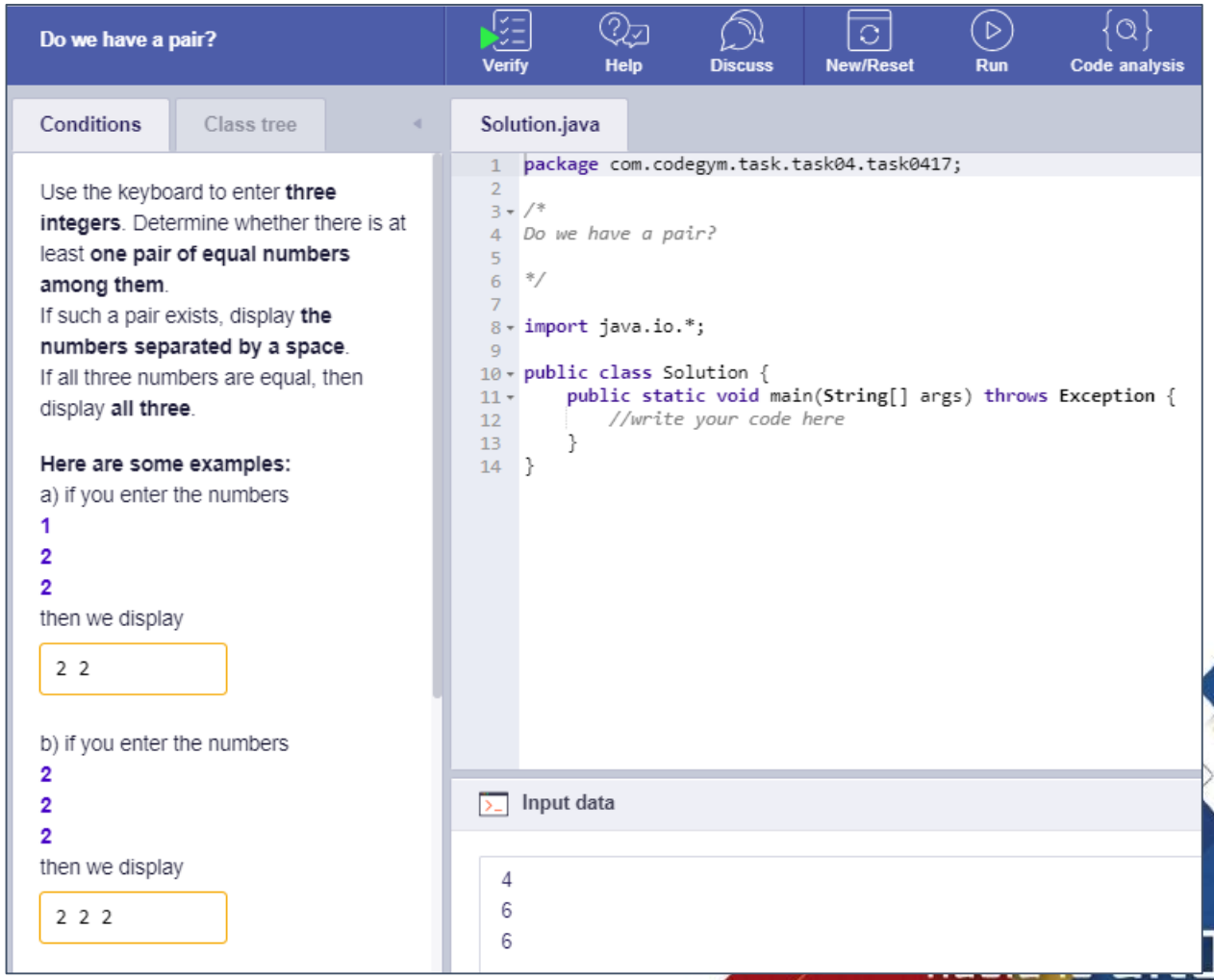
Excercise: **N4_L4_E9**



Gamified platform work / Trabajo en plataforma gamificada.

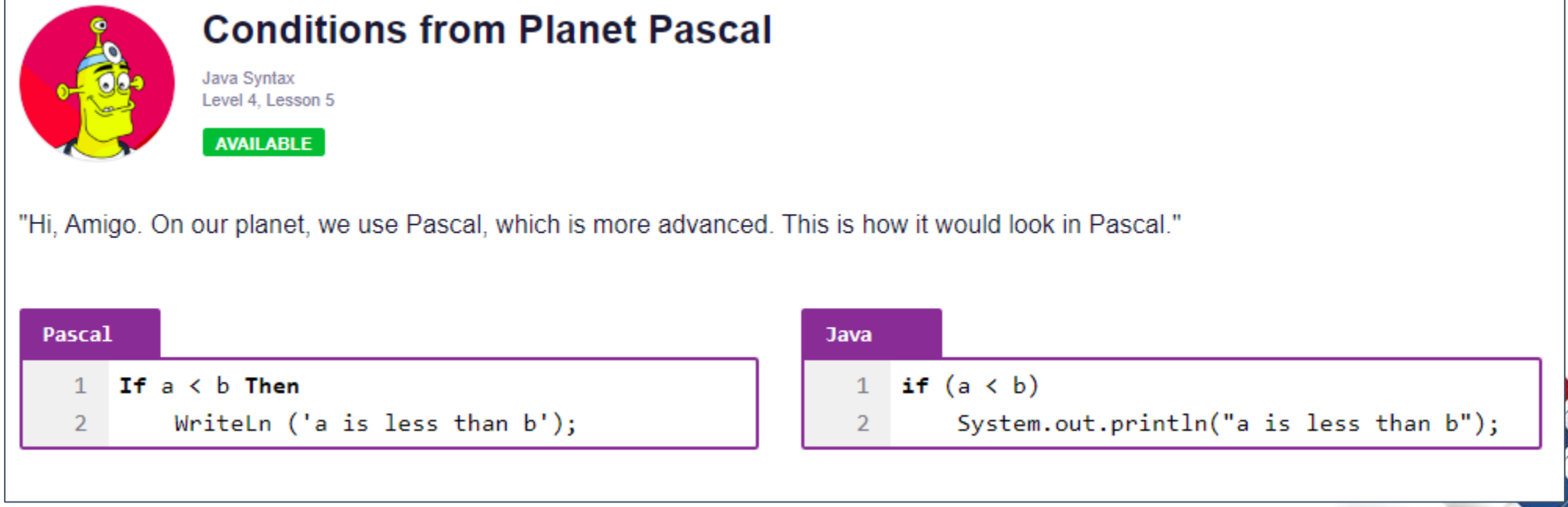
Level 4 - Lesson 4 / Nivel 4 – Lección 4

Excercise: **N4_L4_E10**



Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 5 / Nivel 4 – Lección 5



Conditions from Planet Pascal

Java Syntax
Level 4, Lesson 5

AVAILABLE

"Hi, Amigo. On our planet, we use Pascal, which is more advanced. This is how it would look in Pascal."

Pascal

```
1  If a < b Then
2      WriteLn ('a is less than b');
```

Java

```
1  if (a < b)
2      System.out.println("a is less than b");
```

Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 6 / Nivel 4 – Lección 6

Comparing and setting conditions

Java Syntax
Level 4, Lesson 6

AVAILABLE

"I'd like to tell you a bit about comparing variables in Java."

"You already know the simplest comparison operators – less than (<) and greater than (>)."

"Yep."

"There are also operators like equal to (==) and not equal to (!=). As well as, less than or equal to (<=) and greater than or equal to (>=)."

"Now this is getting interesting."

"Note that there are no =< or => operators in Java!"

"The = sign is used for assignment operations. That's why two equal signs (==) are used to test equality. To check that variables aren't equal, use the != operator."

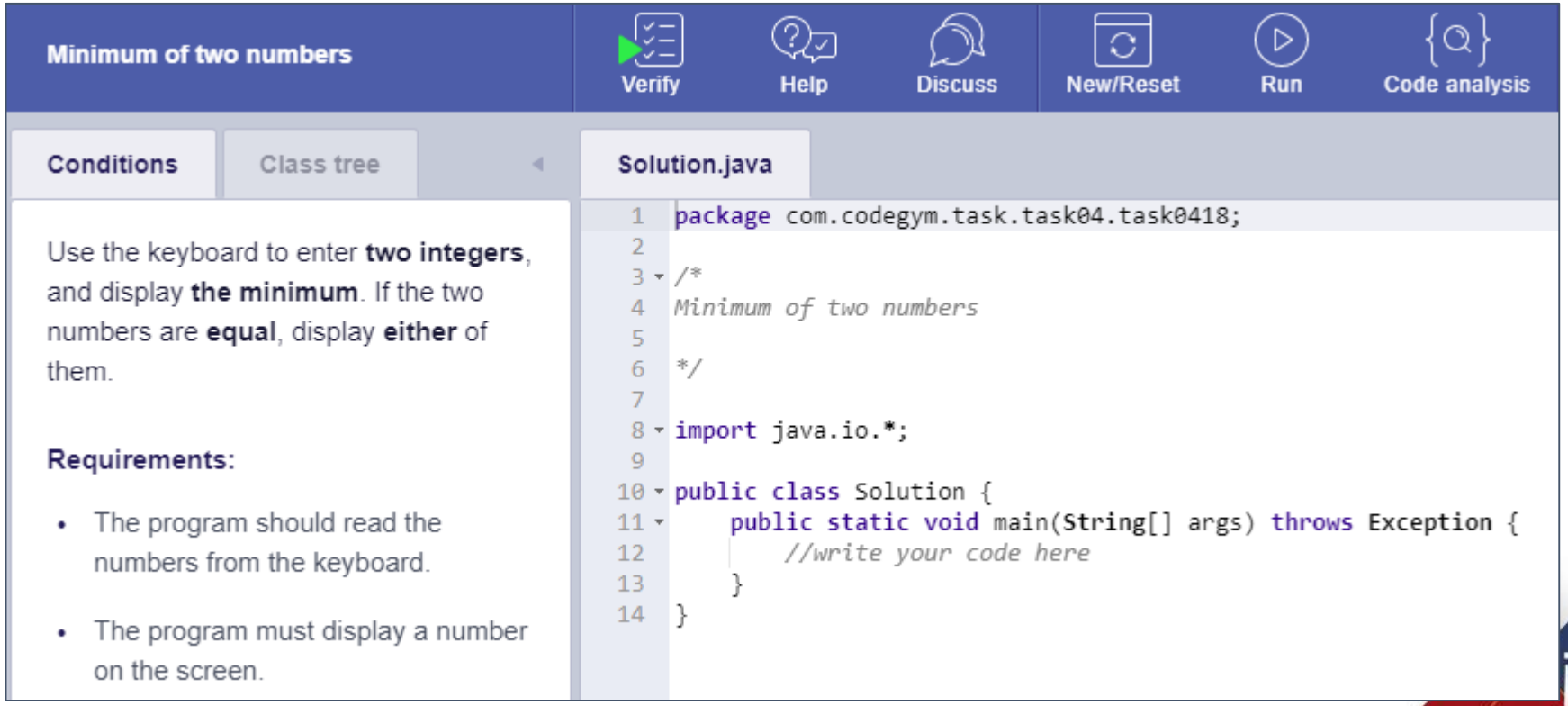
"I see."

"When comparing two variables in Java using the == operator, we are comparing the contents of the variables."

Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 6 / Nivel 4 – Lección 6

Excercise: **N4_L6_E1**

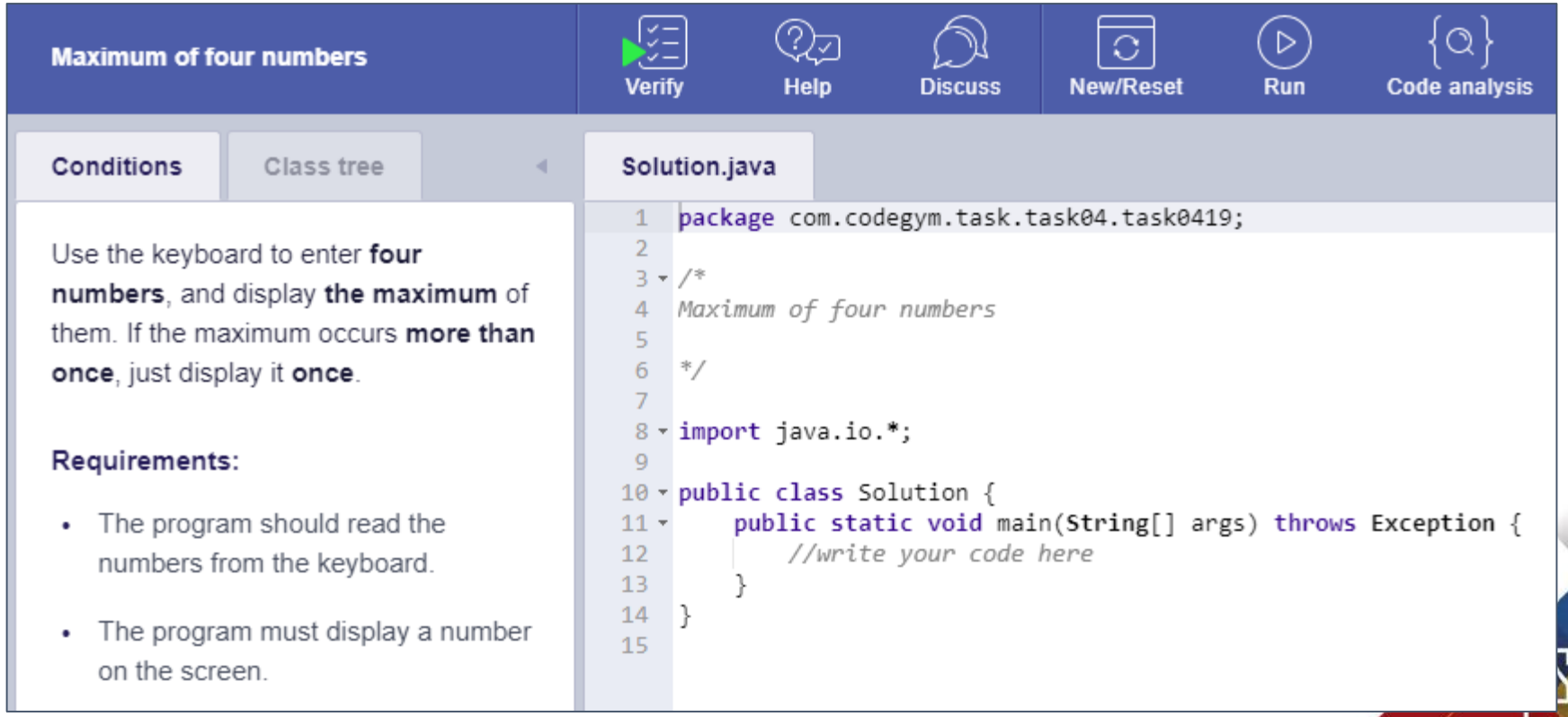


¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 6 / Nivel 4 – Lección 6

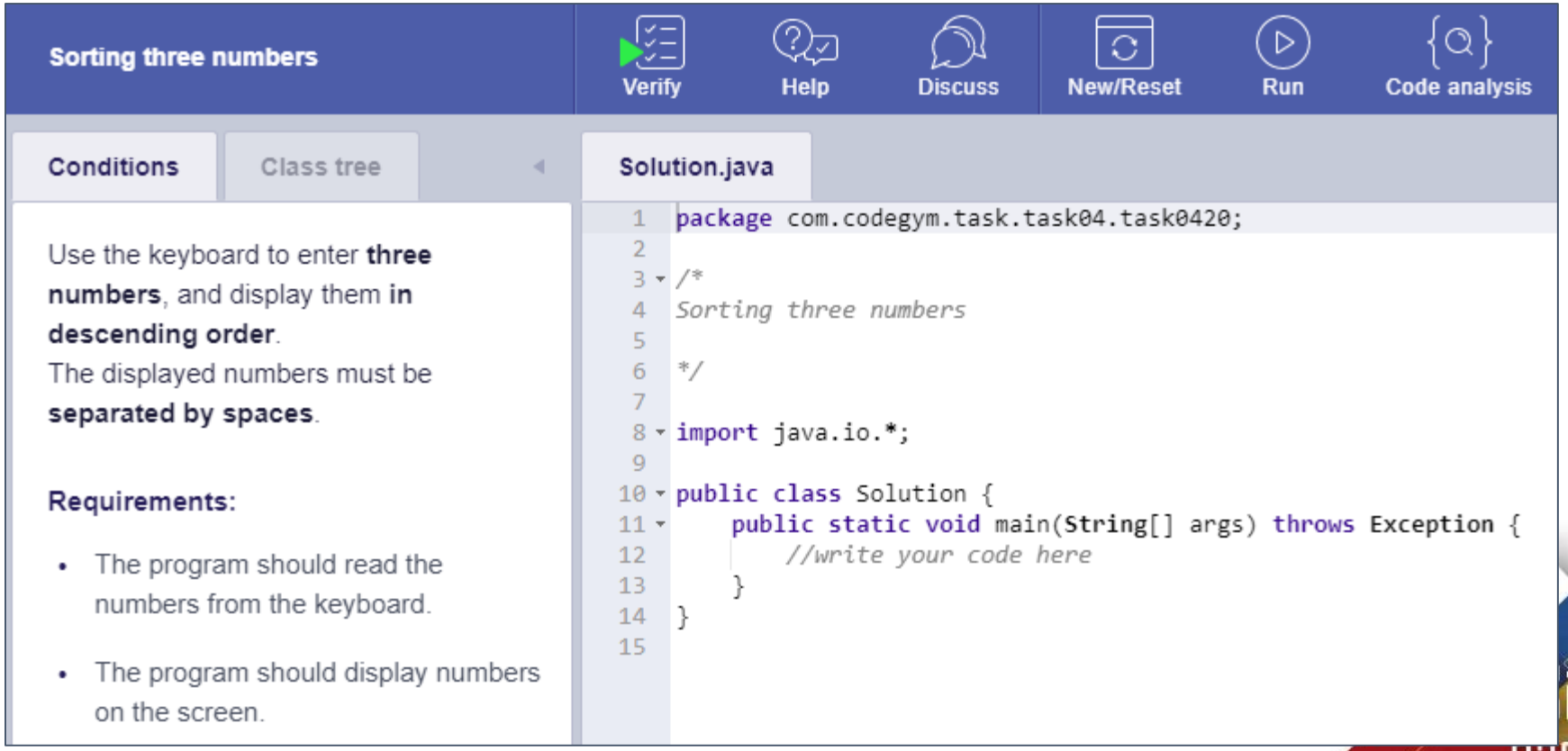
Excercise: **N4_L6_E2**



Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 6 / Nivel 4 – Lección 6

Excercise: **N4_L6_E3**



Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 6 / Nivel 4 – Lección 6

Excercise: **N4_L6_E4**

Jen or Jen?

Verify | Help | Discuss | New/Reset | Run | Code analysis

Conditions | Class tree

Use the keyboard to enter **two names**.
If the names are **identical**, display "***The names are identical***".
If the names are **different**, but they are the same length, display "***The names are the same length***".
If **the names and name lengths are different**, **don't display** anything.

Requirements:

- The program should read two lines from the keyboard.

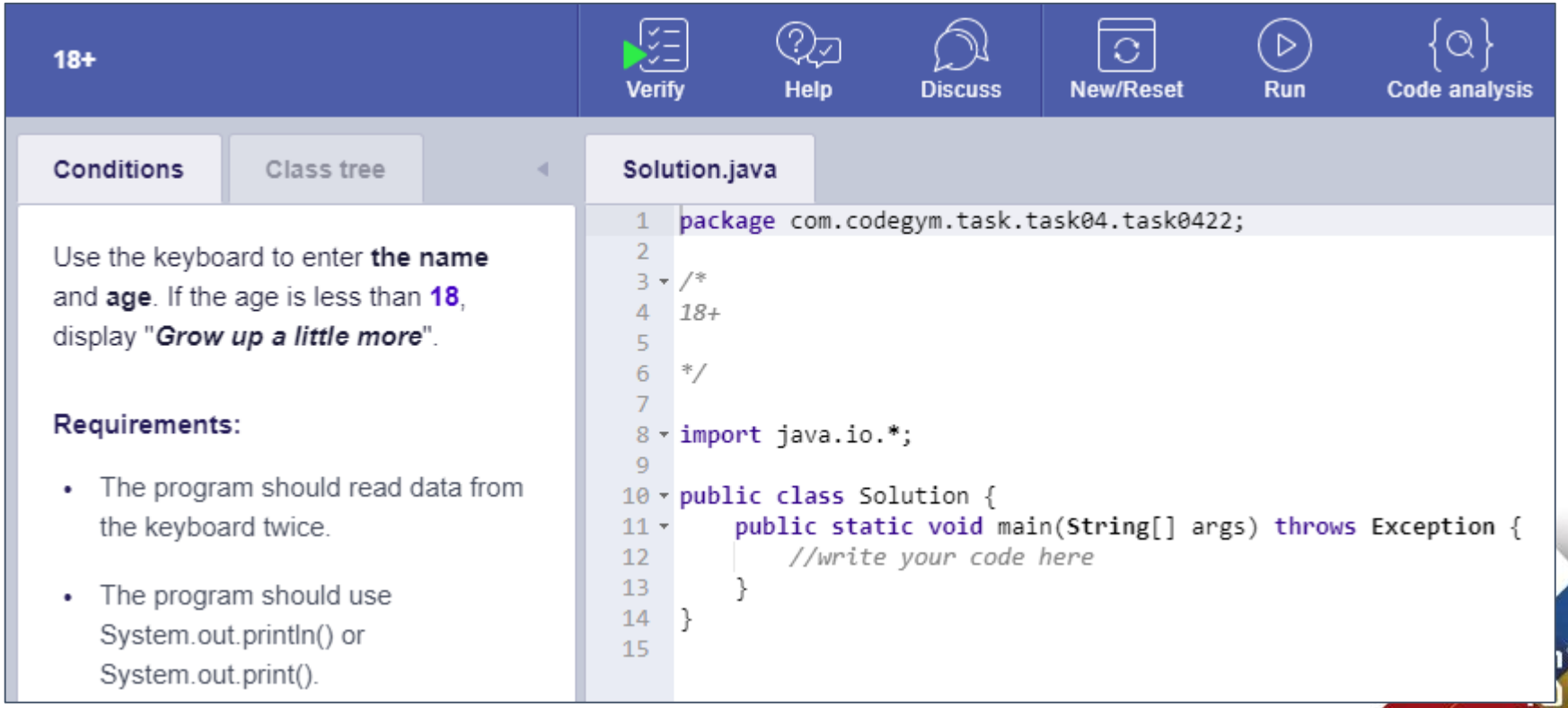
Solution.java

```
1  package com.codegym.task.task04.task0421;
2
3  /*
4  Jen or Jen?
5
6  */
7
8  import java.io.*;
9
10 public class Solution {
11     public static void main(String[] args) throws Exception {
12         //write your code here
13     }
14 }
15
```

Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 6 / Nivel 4 – Lección 6

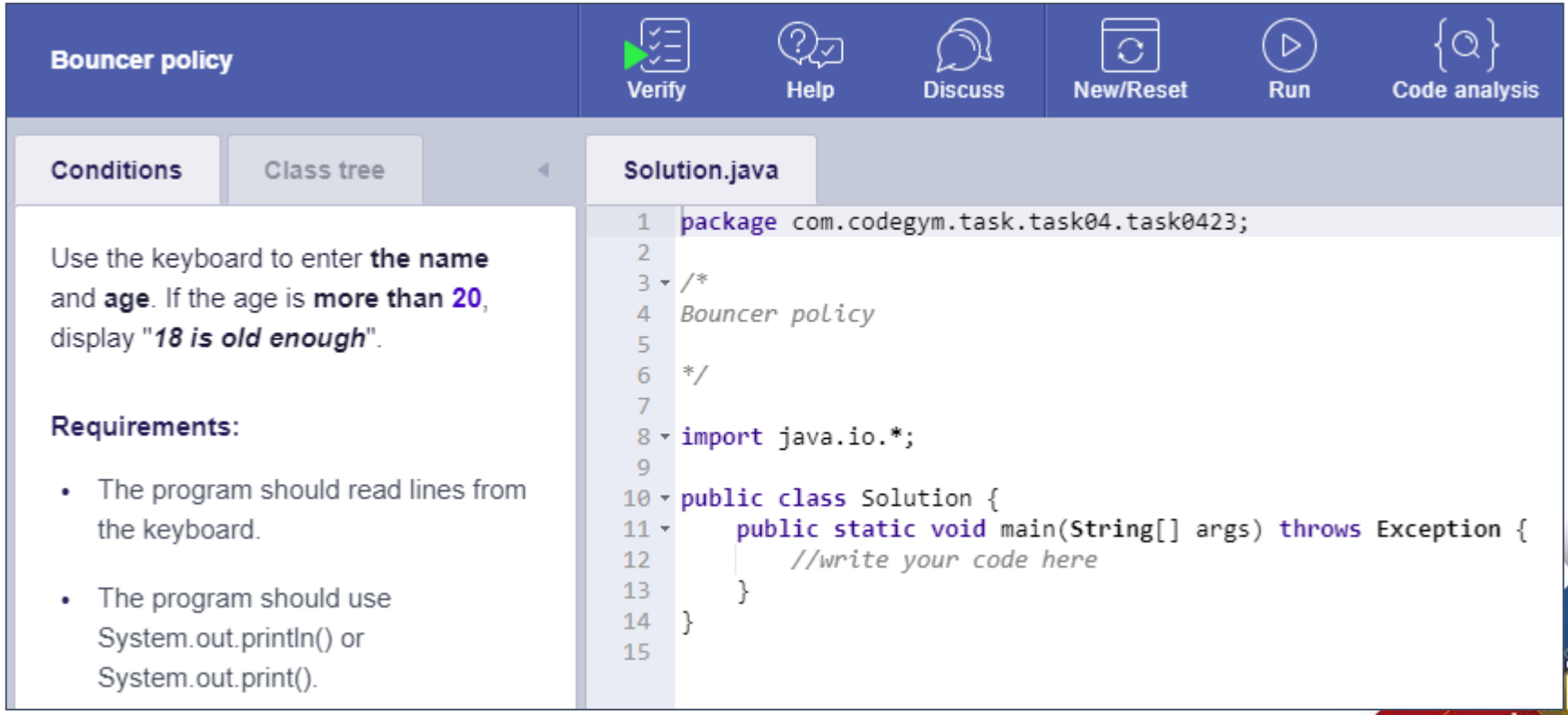
Excercise: **N4_L6_E5**



Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 6 / Nivel 4 – Lección 6

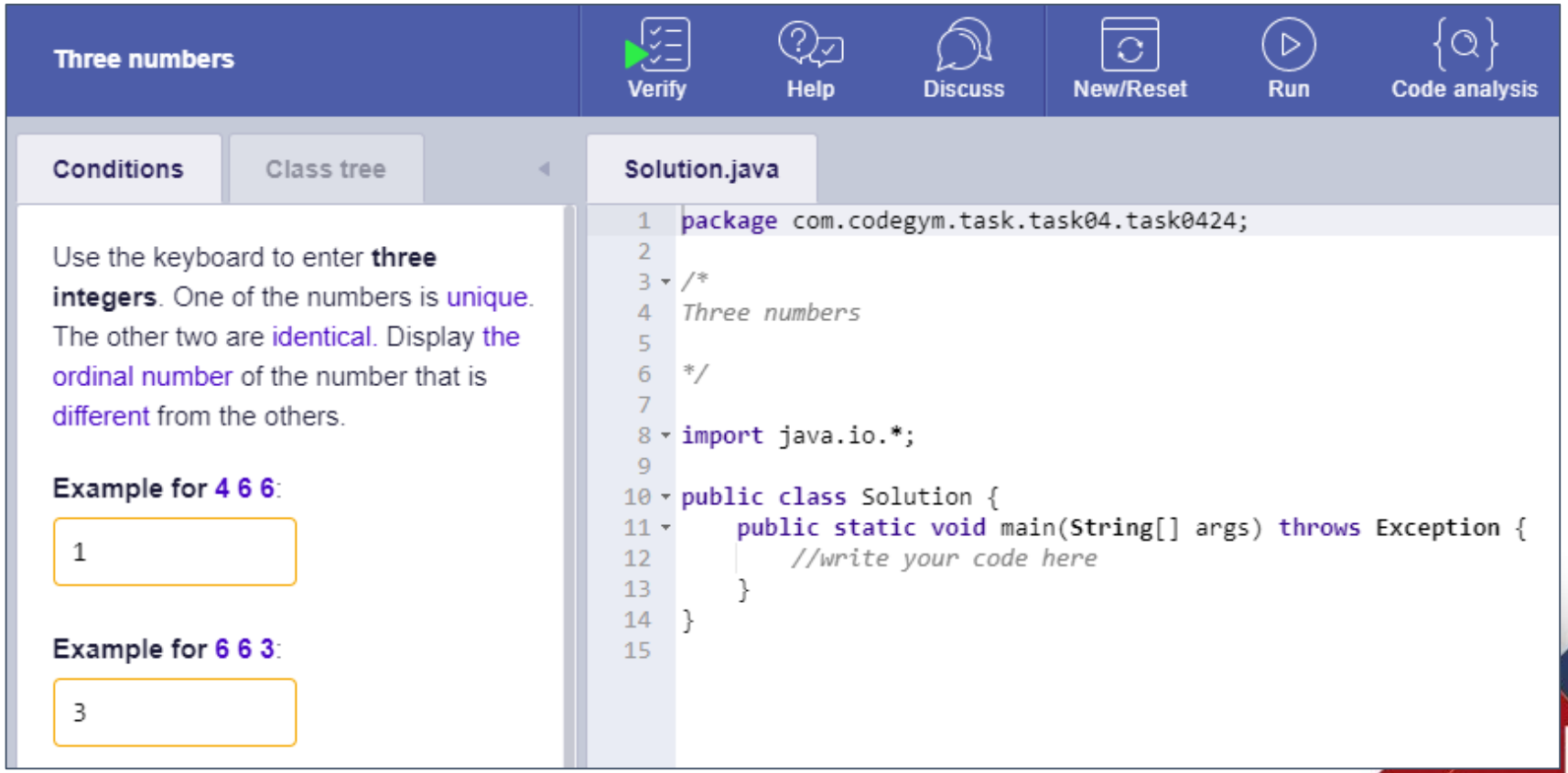
Excercise: **N4_L6_E6**



Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 6 / Nivel 4 – Lección 6

Excercise: **N4_L6_E7**

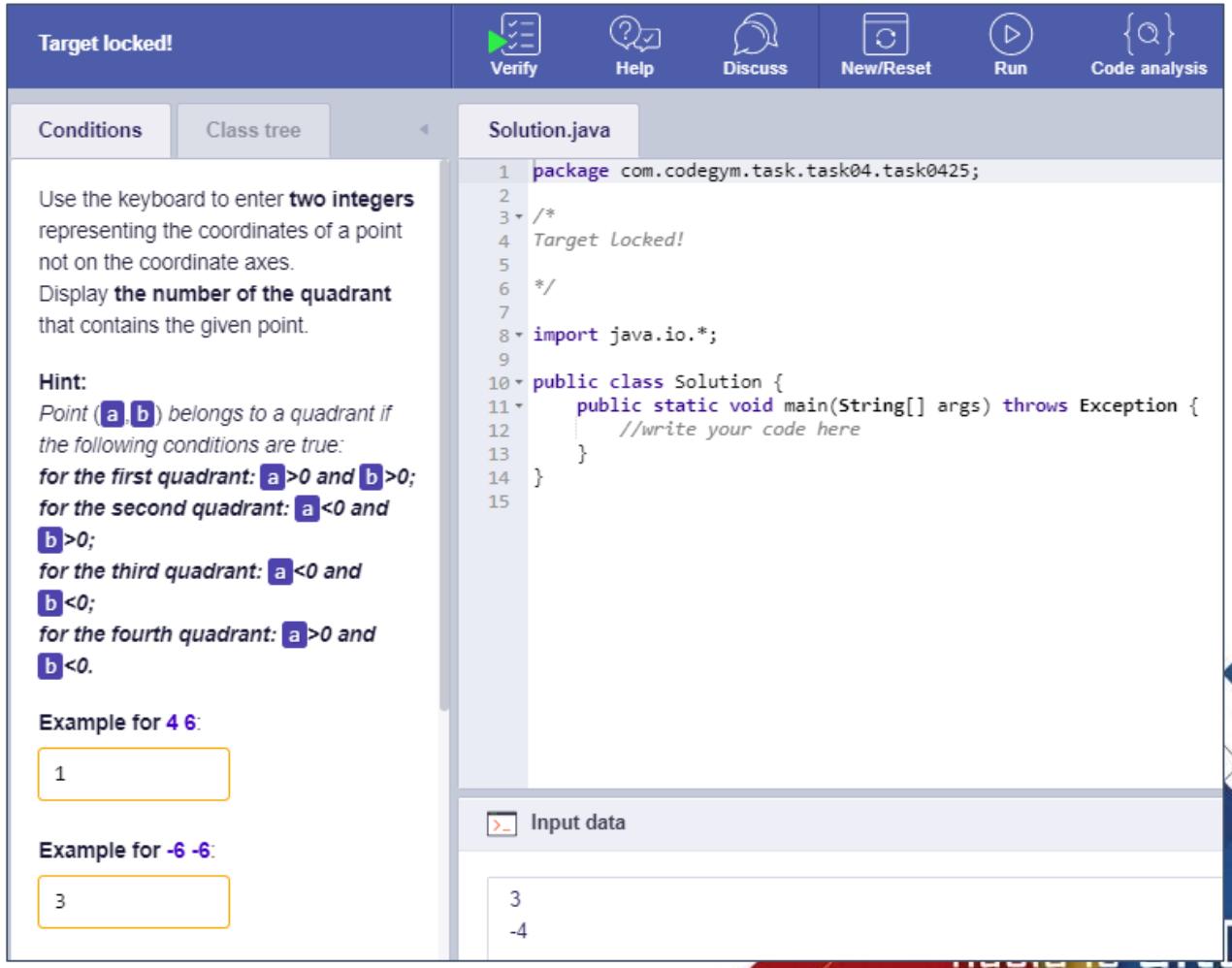


¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 6 / Nivel 4 – Lección 6

Excercise: **N4_L6_E8**



Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 7 / Nivel 4 – Lección 7

Boolean type

Java Syntax
Level 4, Lesson 7

AVAILABLE

"Hi, Amigo. Let me tell you about a new data type. The boolean. Variables of this type can take only two values: **true** and **false**."

"How do we use it?"

"This type is implicitly used in many places. Just as any addition operation produces a number, the result of any comparison is a **boolean**. Here are some examples:"

	Code	Explanation
1	`boolean m;`	These two expressions are equivalent. The default value of a boolean variable is **false**.
2	`boolean m = false;`	
3	`if (a > b)` `  System.out.println(a);`	The result of the comparison (either **true** or **false**) will be assigned to the variable `m`. The condition is satisfied if the expression evaluates to **true**.
	`boolean m = (a > b);`	

¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 7 / Nivel 4 – Lección 7

Excercise: **N4_L7_E1**

Labels and numbers

Verify | Help | Discuss | New/Reset | Run | Code analysis

Conditions | Class tree

Use the keyboard to enter an integer.
Display a string description as follows:
"***Negative even number***" - if the number is **negative** and **even**,
"***Negative odd number***" - if the number is **negative** and **odd**,
"***Zero***" - if the number is **0**,
"***Positive even number***" - if the number is **positive** and **even**,
"***Positive odd number***" - if the number is **positive** and **odd**.

Example for 100:

```
Positive even number
```

Example for -51:

```
Negative odd number
```

Solution.java

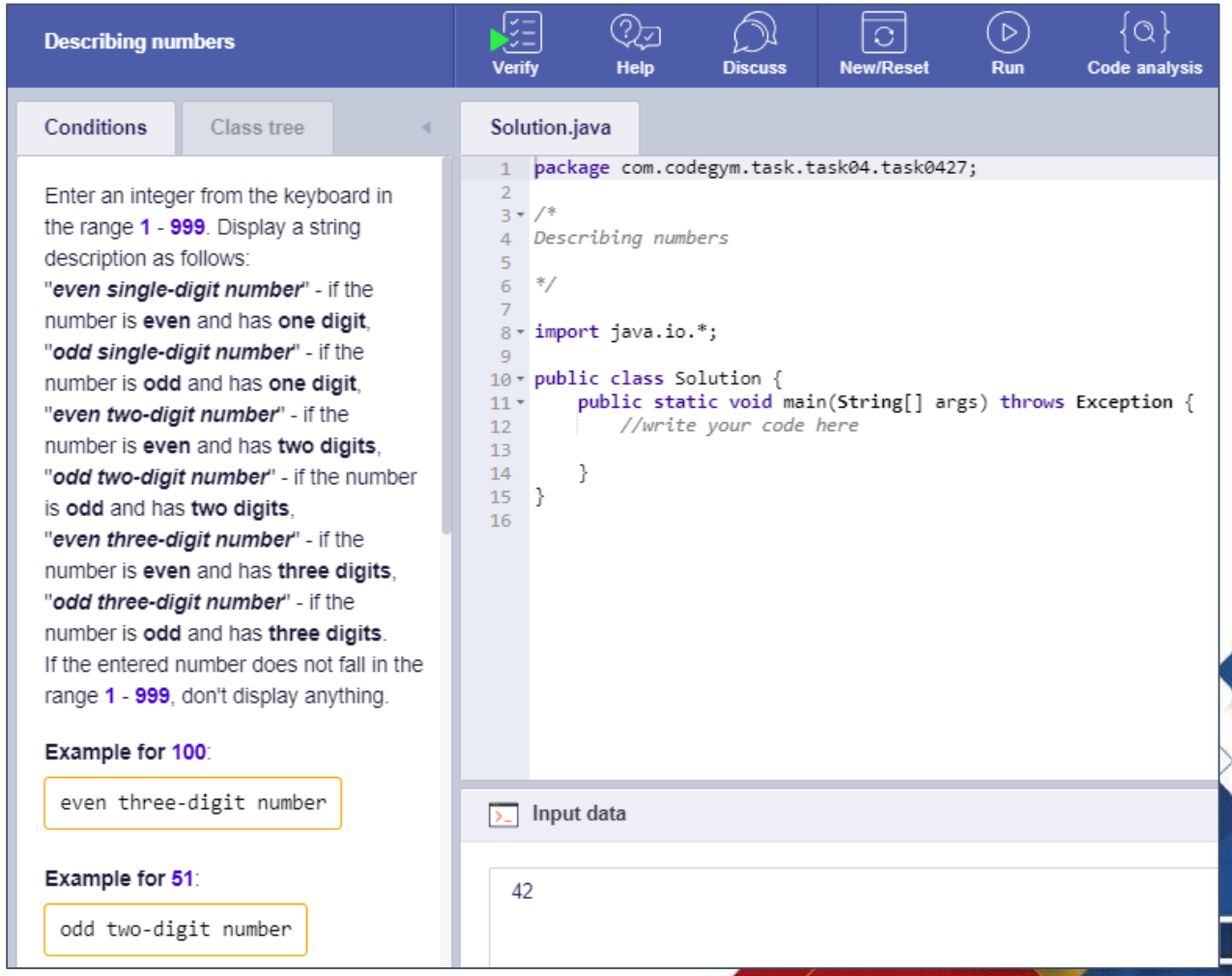
```java
package com.codegym.task.task04.task0426;

/*
Labels and numbers

*/

import java.io.*;

public class Solution {
    public static void main(String[] args) throws Exception {
        //write your code here
    }
}
```

¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 7 / Nivel 4 – Lección 7

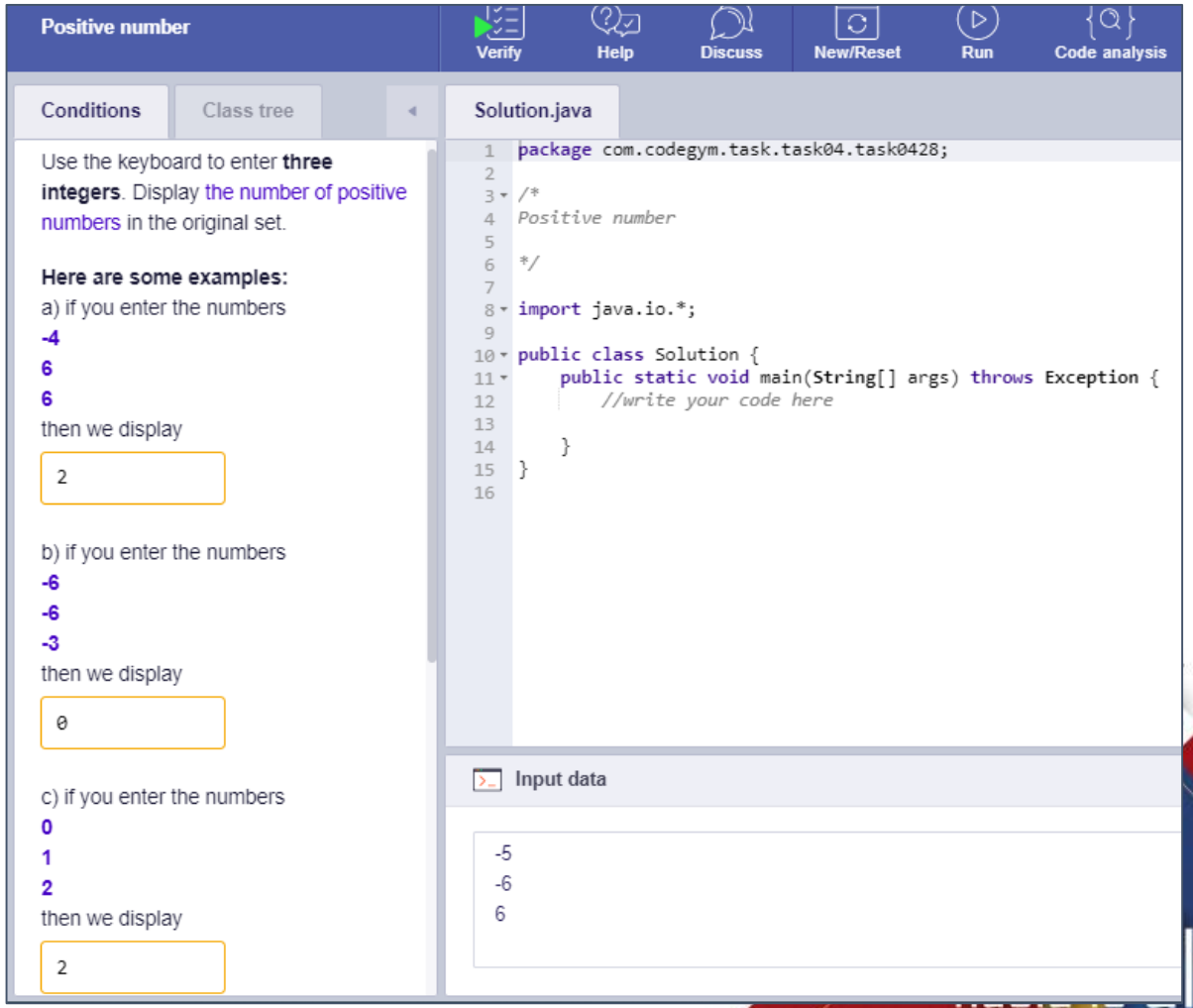
Excercise: **N4_L7_E2**



Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 7 / Nivel 4 – Lección 7

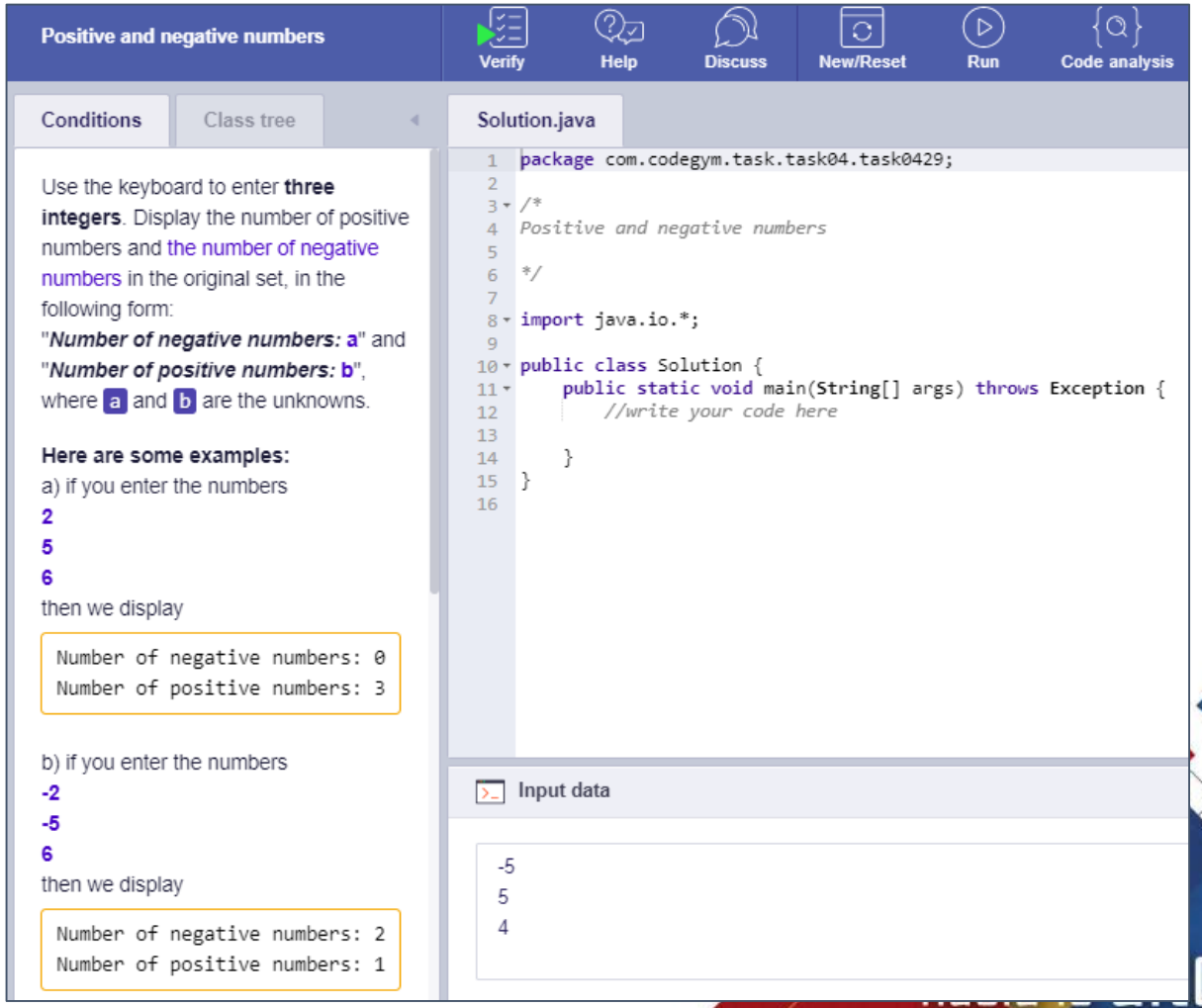
Excercise: **N4_L7_E3**



Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 7 / Nivel 4 – Lección 7

Excercise: **N4_L7_E4**



Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 8 / Nivel 4 – Lección 8

Loops

Java Syntax
Level 4, Lesson 8

AVAILABLE

"Hi."

"Hi, Ellie!"

"It's time to learn about loops. Loops are as simple as if/else statements, but even more interesting. You can use a loop to execute any command or a block of commands multiple times. In general, a loop looks like this:"

Loop (example 1)

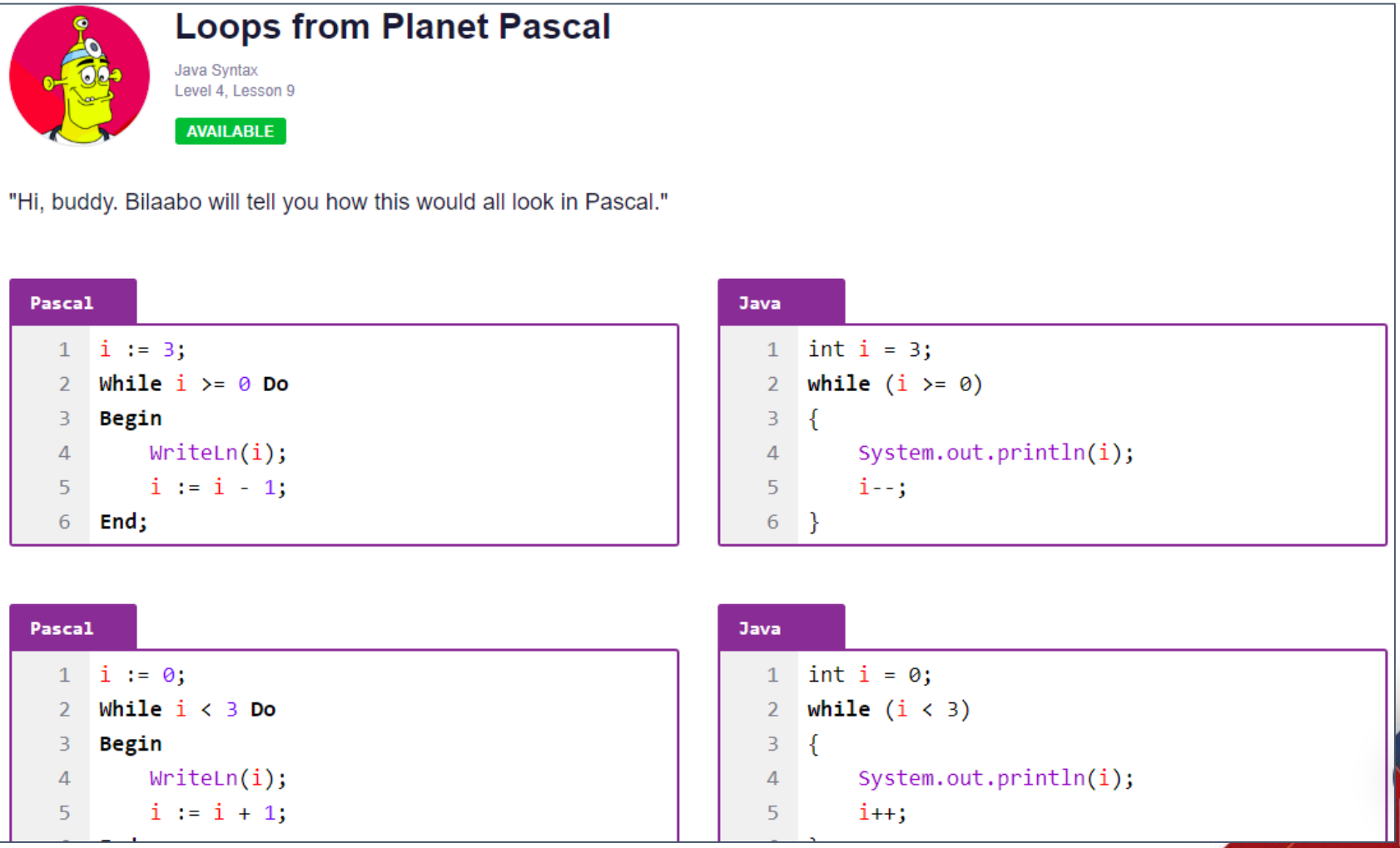
```
while(boolean condition)
    command;
```

Loop (example 2)

```
while(boolean condition)
    block of commands in curly brackets
```

Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 9 / Nivel 4 – Lección 9



Loops from Planet Pascal

Java Syntax
Level 4, Lesson 9

AVAILABLE

"Hi, buddy. Bilaabo will tell you how this would all look in Pascal."

Pascal

```
1  i := 3;
2  While i >= 0 Do
3  Begin
4      WriteLn(i);
5      i := i - 1;
6  End;
```

Java

```
1  int i = 3;
2  while (i >= 0)
3  {
4      System.out.println(i);
5      i--;
6  }
```

Pascal

```
1  i := 0;
2  While i < 3 Do
3  Begin
4      WriteLn(i);
5      i := i + 1;
```

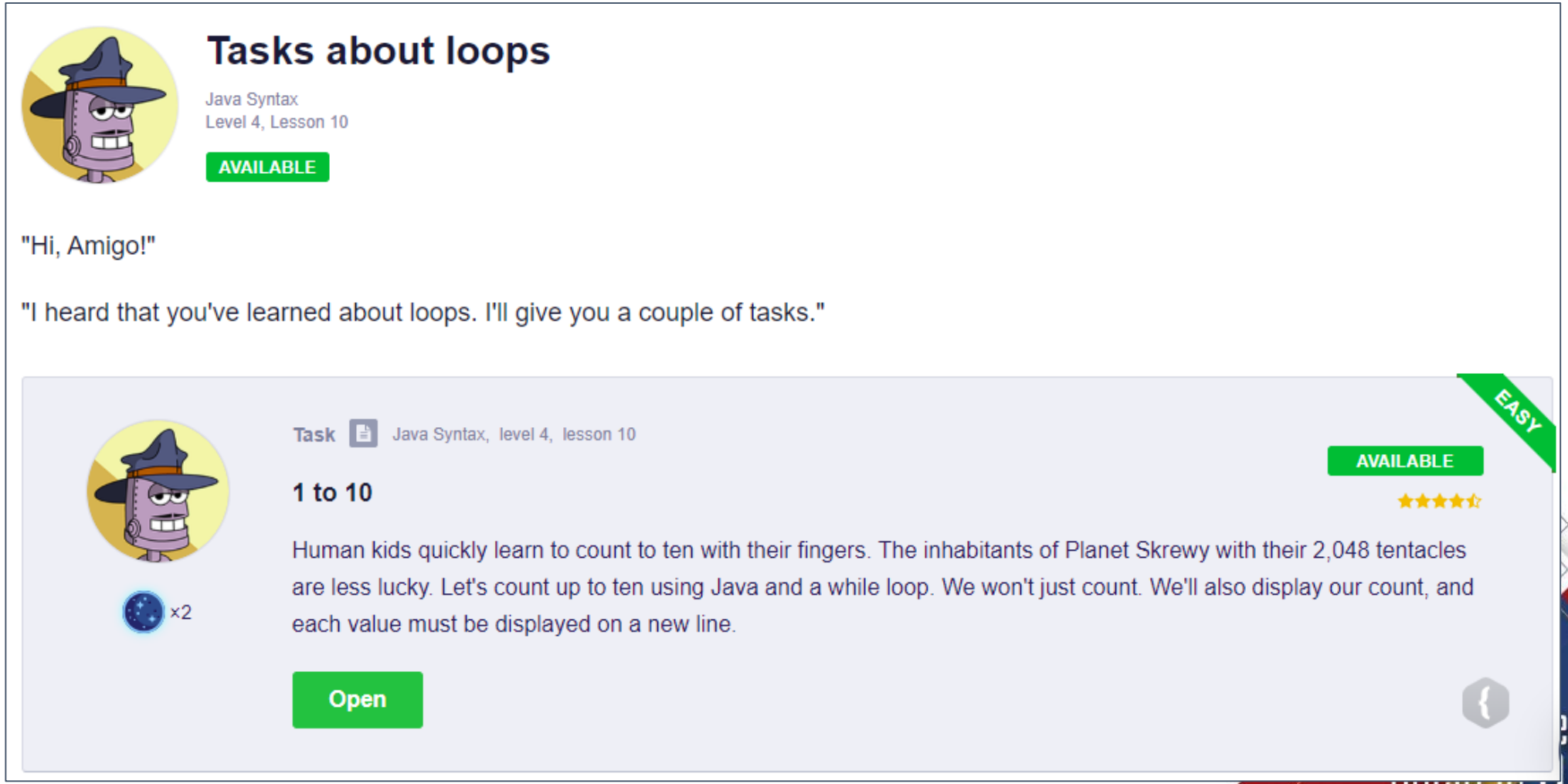
Java

```
1  int i = 0;
2  while (i < 3)
3  {
4      System.out.println(i);
5      i++;
```

¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

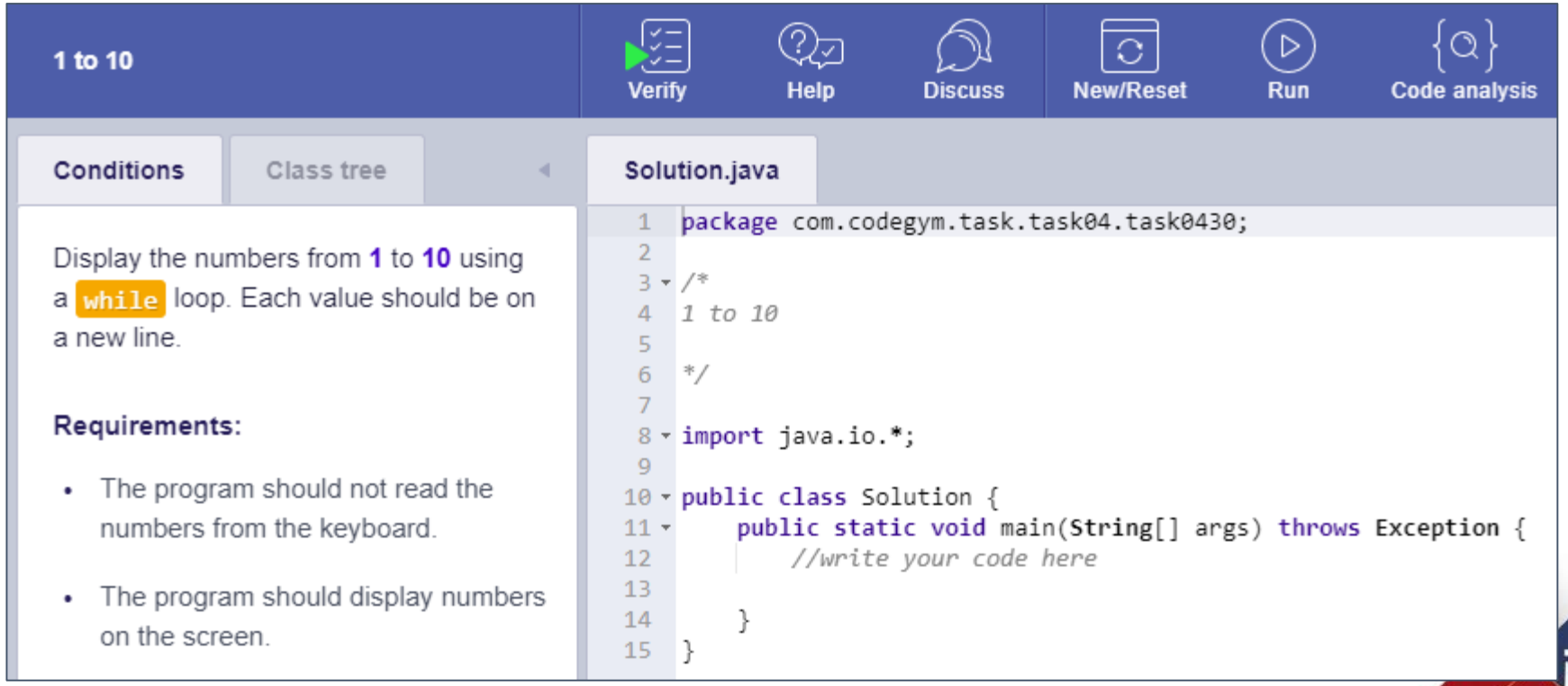
Level 4 - Lesson 10 / Nivel 4 – Lección 10



Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 10 / Nivel 4 – Lección 10

Excercise: **N4_L10_E1**

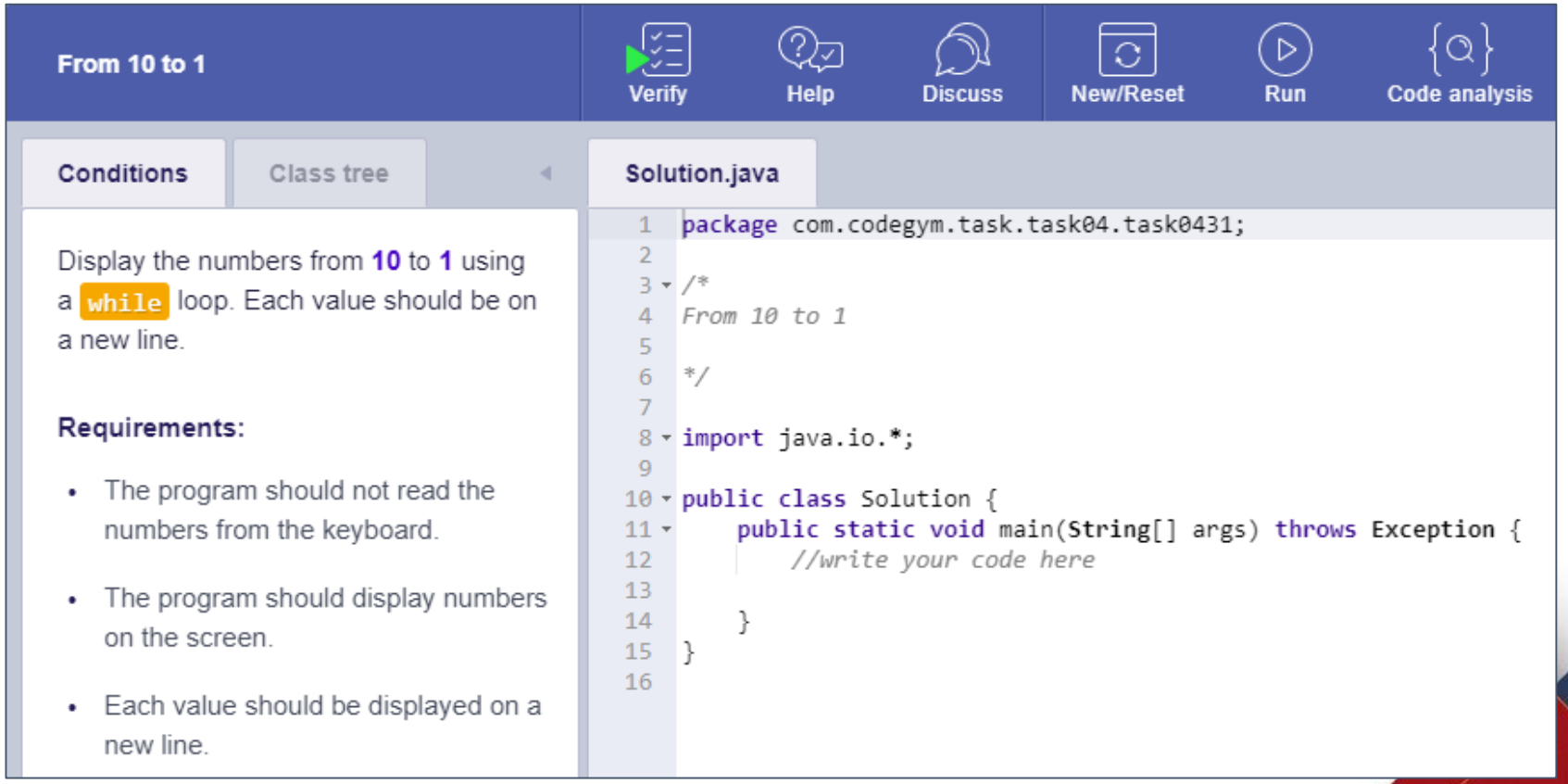


¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 10 / Nivel 4 – Lección 10

Excercise: **N4_L10_E2**

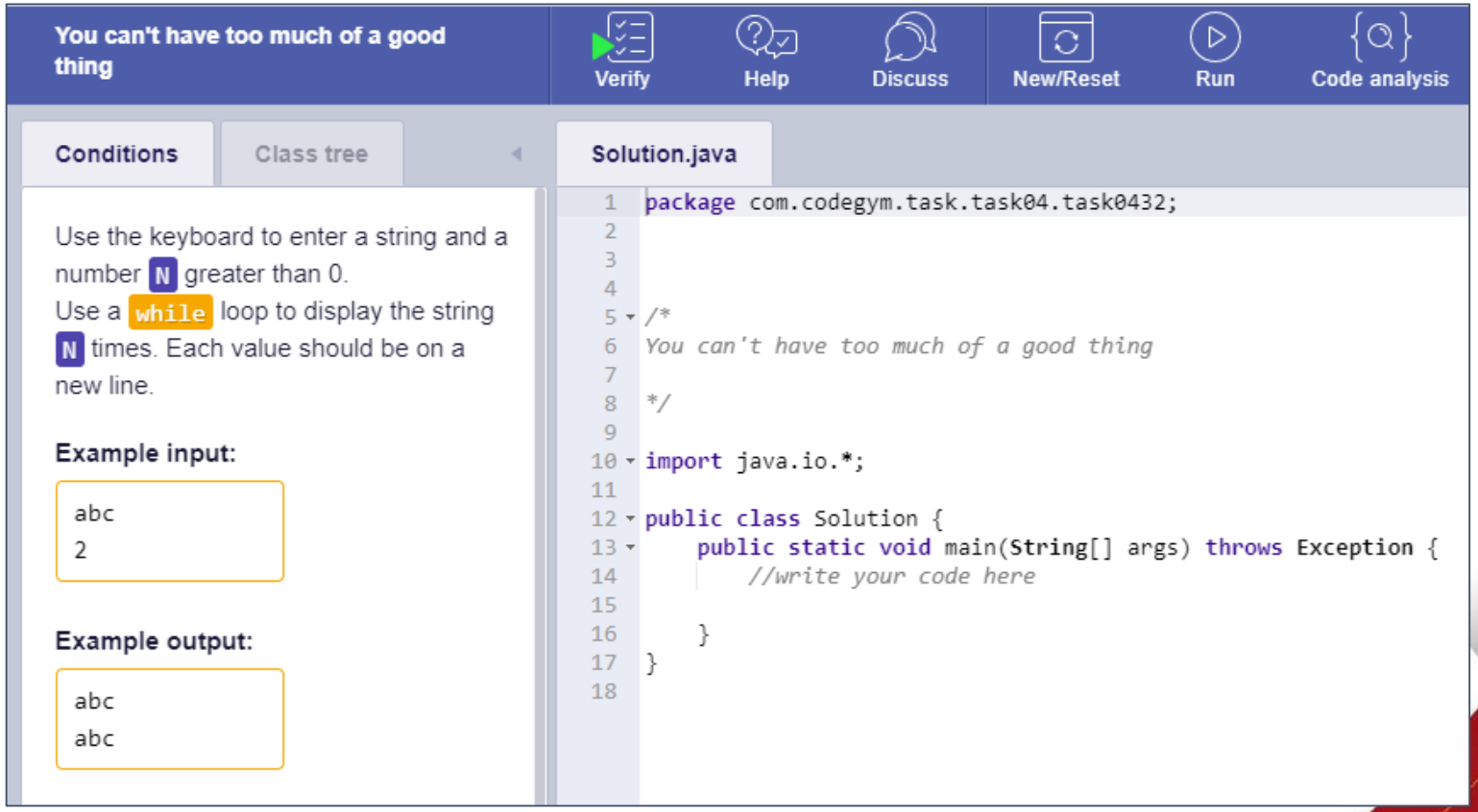


¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 10 / Nivel 4 – Lección 10

Excercise: **N4_L10_E3**

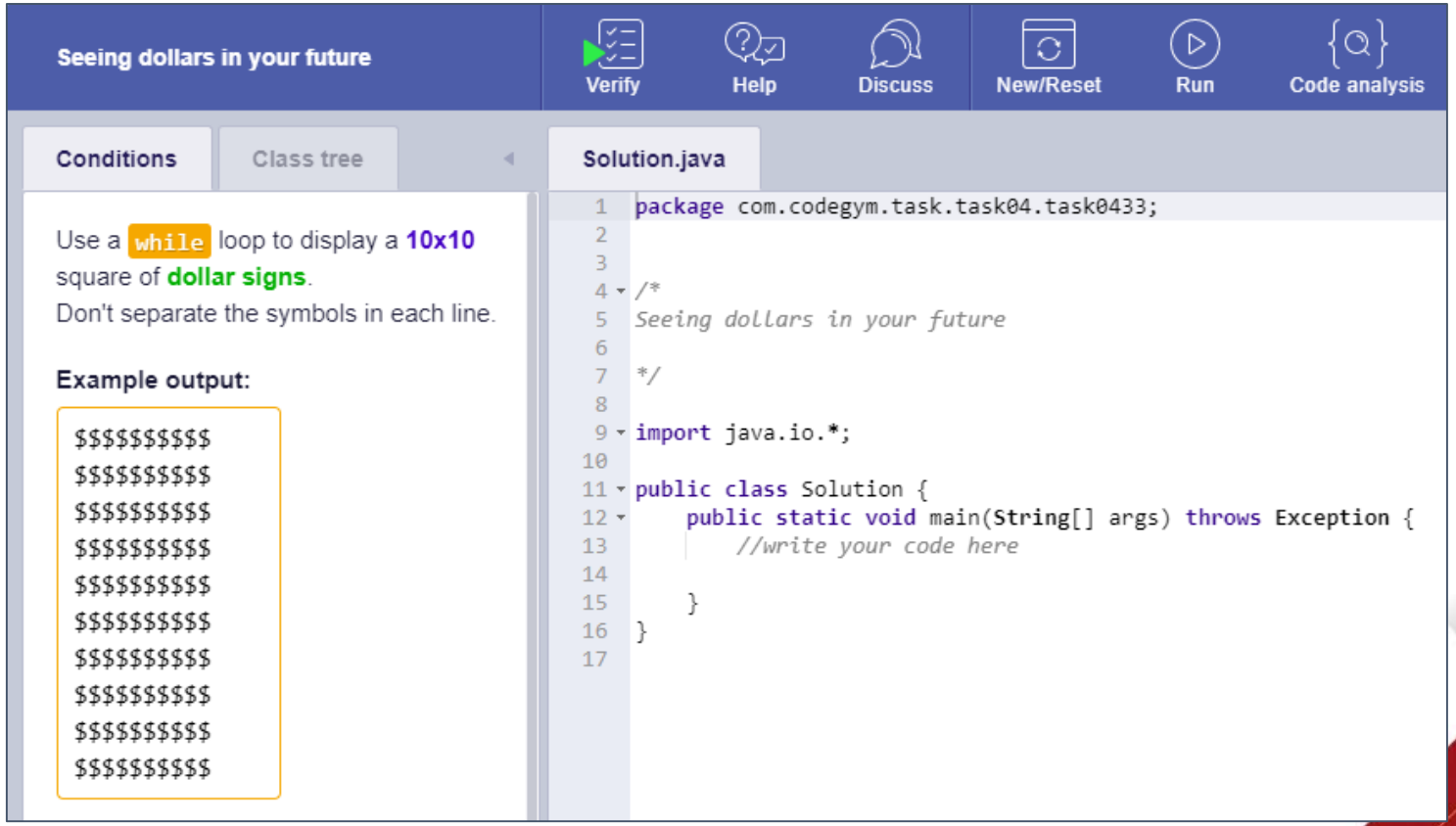


¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 10 / Nivel 4 – Lección 10

Excercise: **N4_L10_E4**

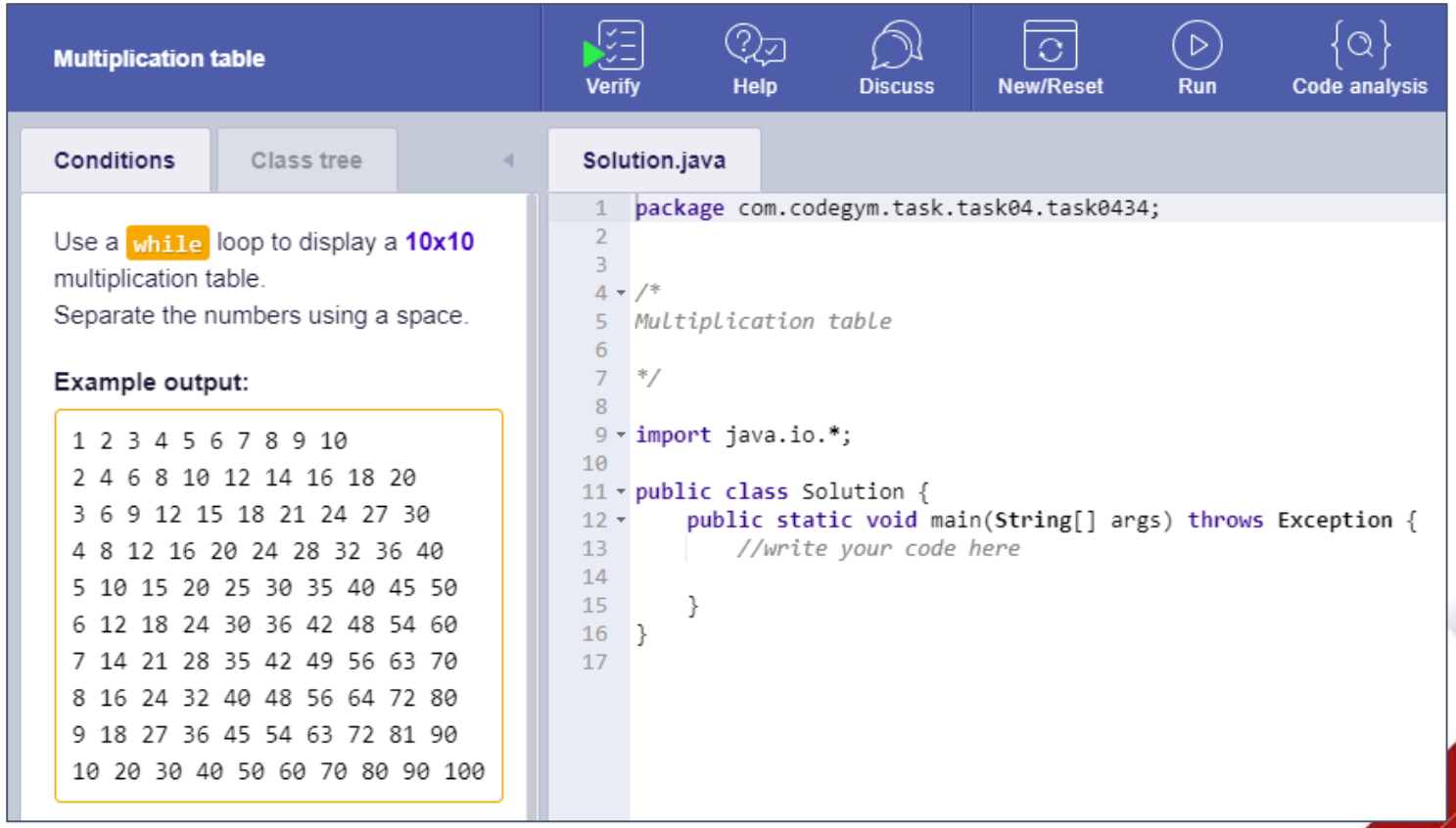


¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 10 / Nivel 4 – Lección 10

Excercise: **N4_L10_E5**



¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 11 / Nivel 4 – Lección 11

For loop

Java Syntax
Level 4, Lesson 11

AVAILABLE

"I want to tell you about one more loop. The for loop. It's just another way to express a while loop, just more compact and convenient (for programmers). Here are some examples:"

while

```
1  int i = 3;
2  while (i >= 0)
3  {
4      System.out.println(i);
5      i--;
6  }
```

for

```
1
2  for (int i = 3; i >= 0; i--)
3  {
4      System.out.println(i);
   }
```

Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 12 / Nivel 4 – Lección 12

For loop in Pascal

Java Syntax
Level 4, Lesson 12

AVAILABLE

"Amigo, you won't believe it, but there is a **For** loop in the Pascal language too. This construct exists in almost every programming language. But in Pascal it's much clearer. Look:"

Pascal

```
1  For i := 1 to 10 do
2  Begin
3      Writeln(i);
4  End;
```

Java

```
1  for (int i = 1; i <= 10; i++)
2  {
3      System.out.println(i);
4  }
```

Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 13 / Nivel 4 – Lección 13

For loop in practice

Java Syntax
Level 4, Lesson 13

AVAILABLE

"Today is my lucky day. I came up with five new tasks for you. I'm brimming with creativity. Good luck, my friend. You will need it..."

×2

Task Java Syntax, level 4, lesson 13

EASY

AVAILABLE

Even numbers

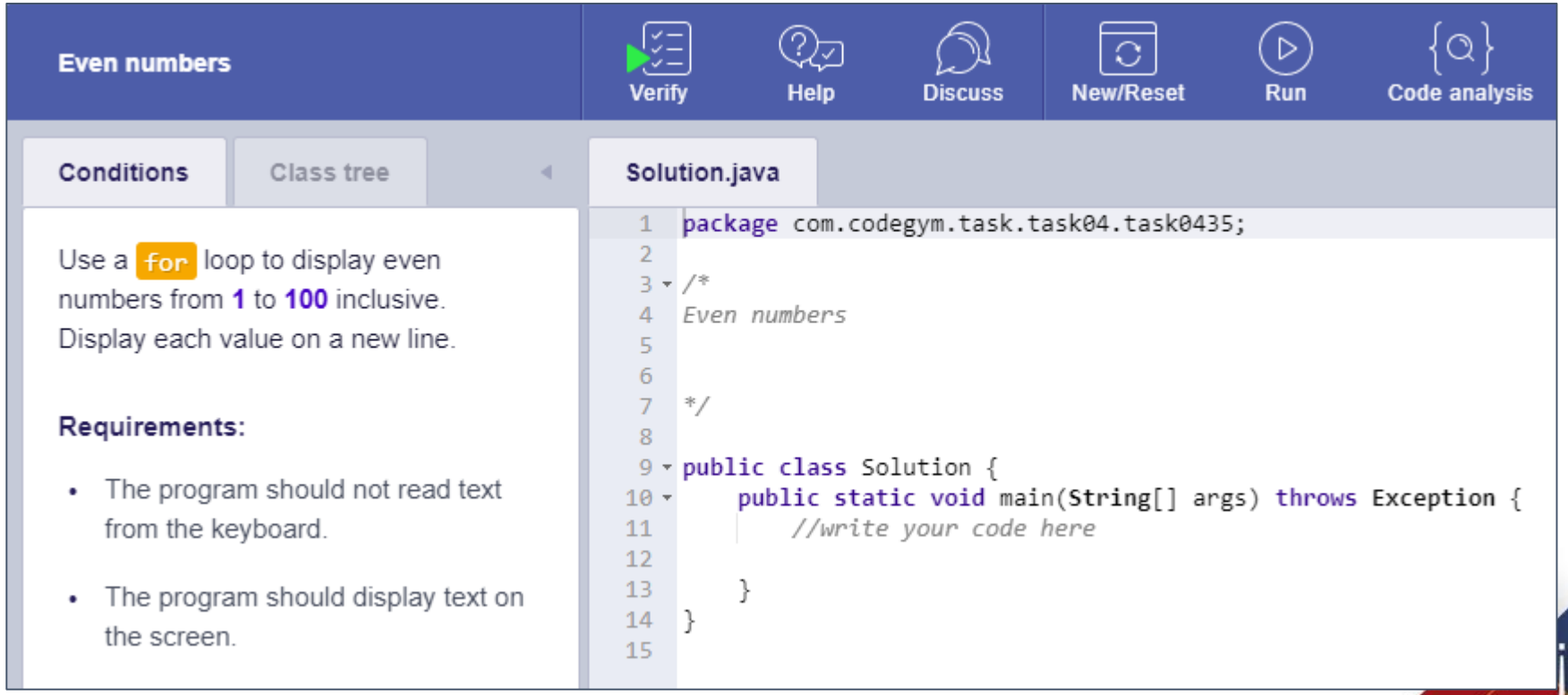
A for loop simplifies the counting, simplifies displaying text, and simplifies... the life of a programmer! And it saves a lot of time: just a couple of simple lines of code and you can display every even number from 1 to 100 inclusive. Let's try it, but be sure to put each value on a separate line.

Open

Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 13 / Nivel 4 – Lección 13

Excercise: **N4_L13_E1**

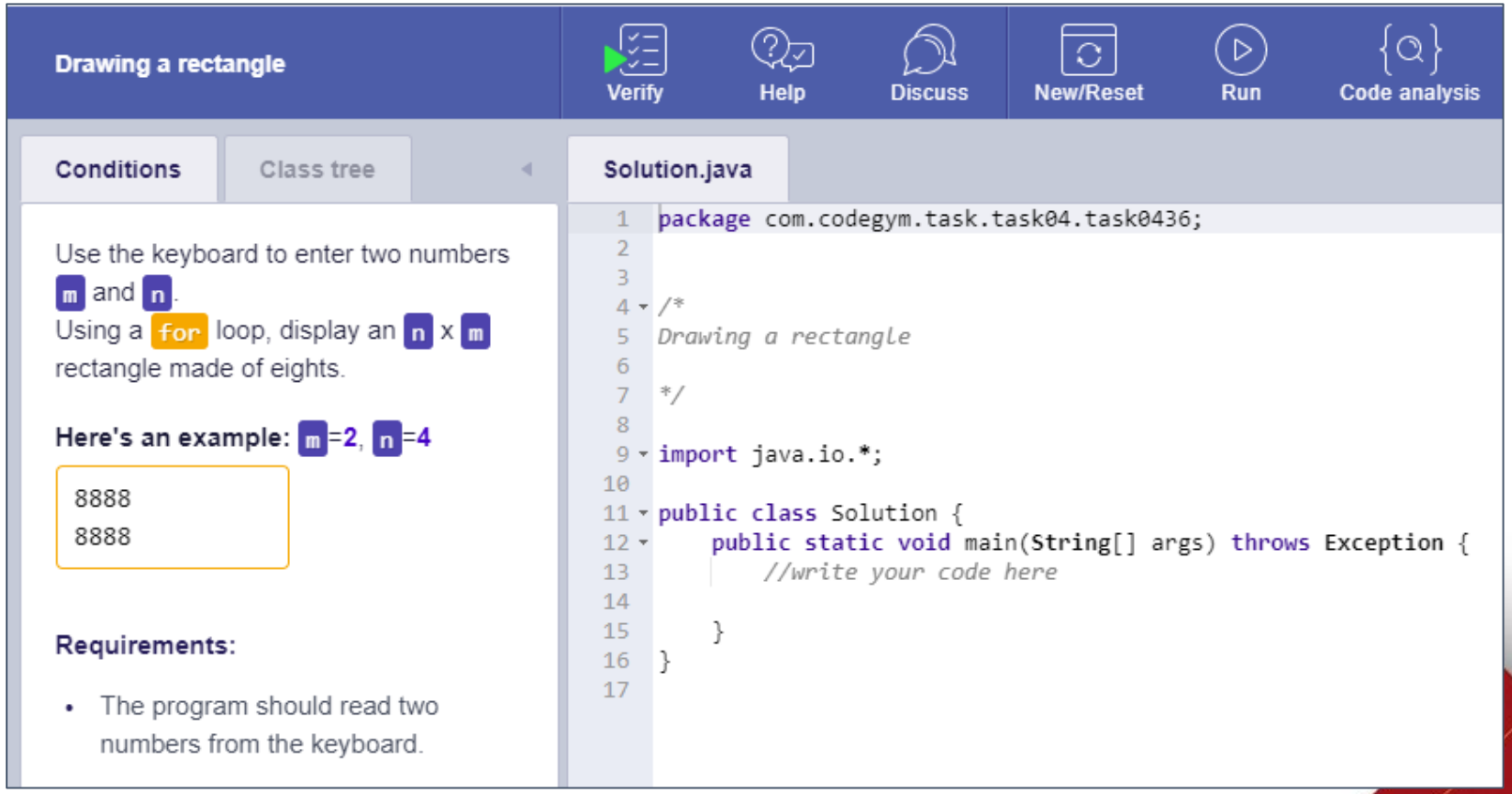


¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 13 / Nivel 4 – Lección 13

Excercise: **N4_L13_E2**

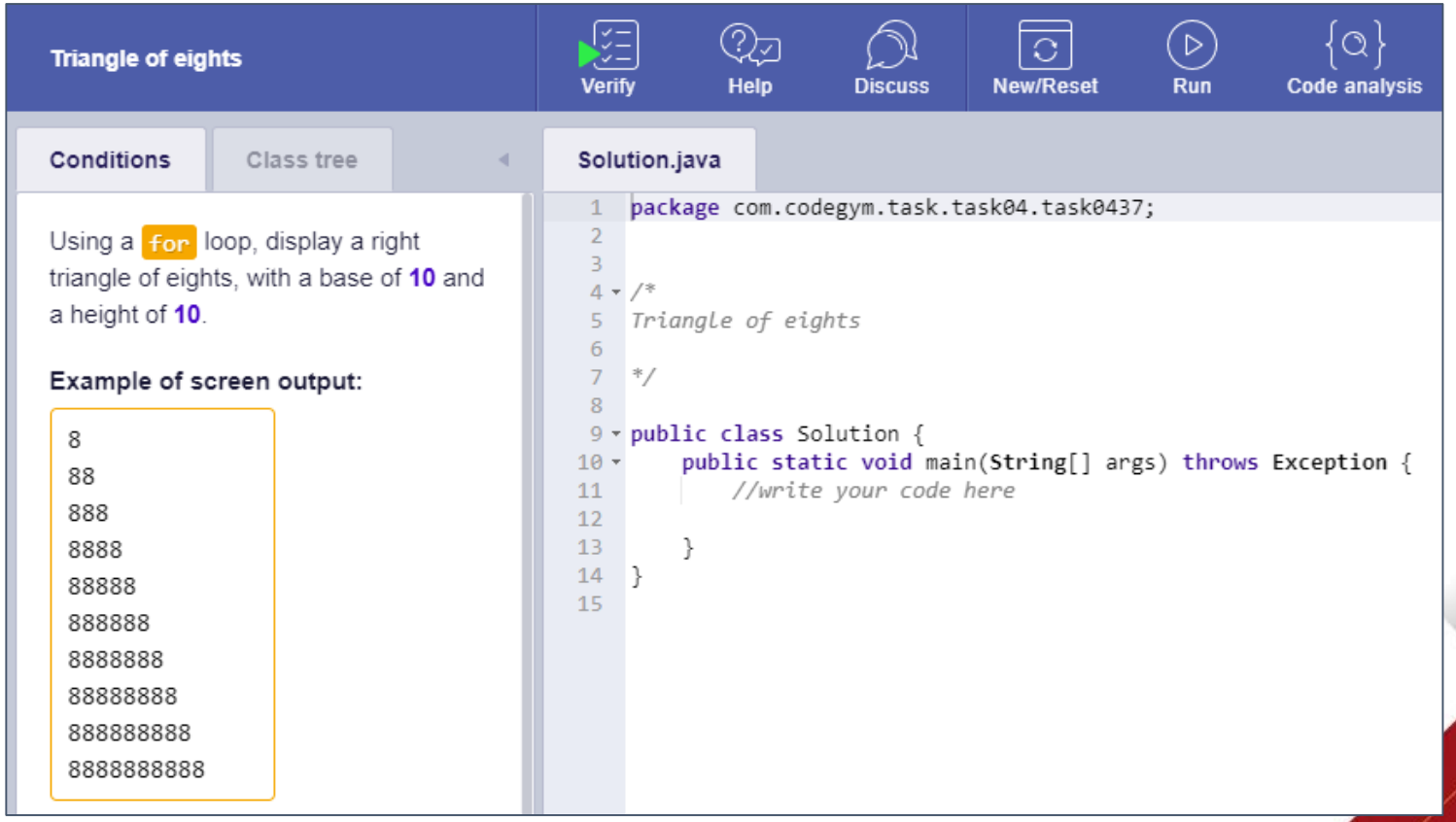


¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 13 / Nivel 4 – Lección 13

Excercise: **N4_L13_E3**

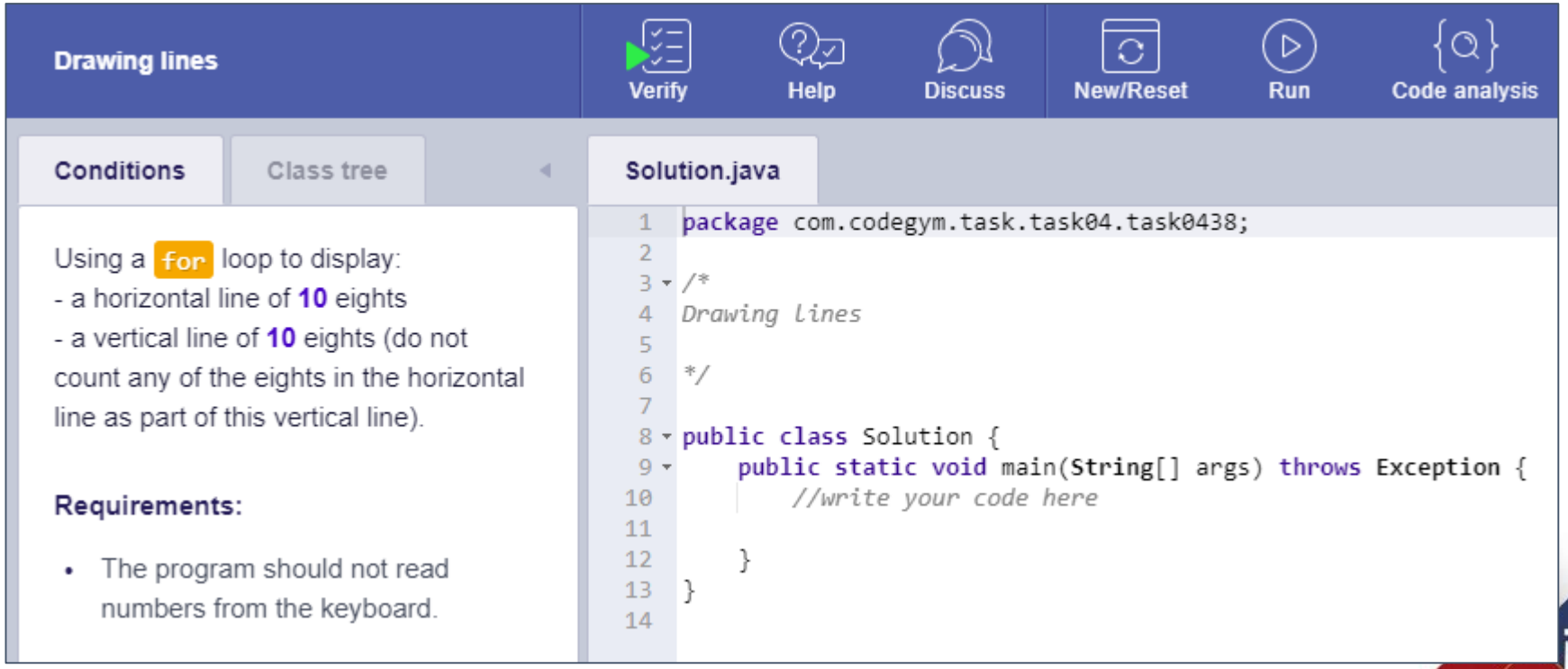


¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 13 / Nivel 4 – Lección 13

Excercise: **N4_L13_E4**

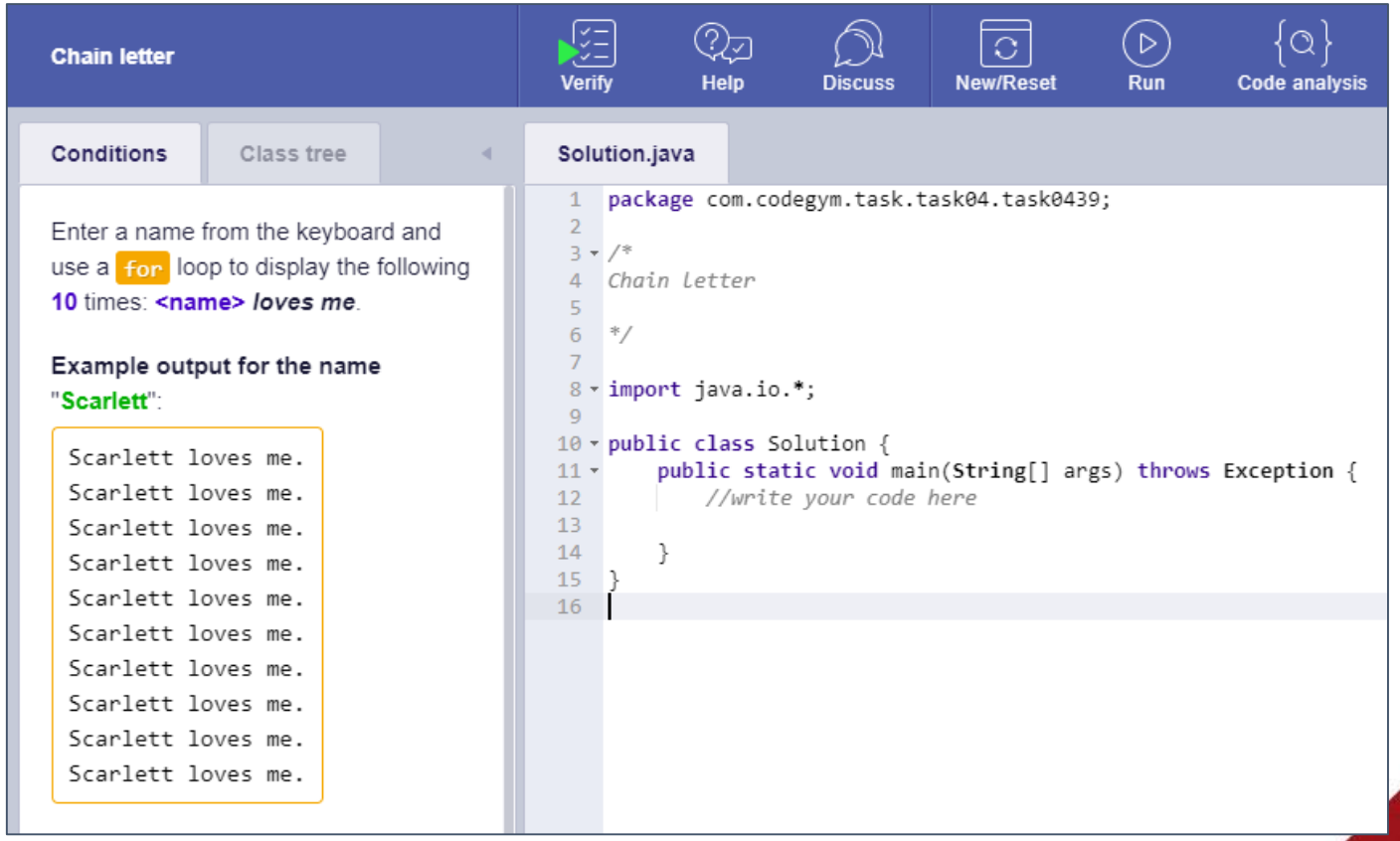


¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 13 / Nivel 4 – Lección 13

Excercise: **N4_L13_E5**



¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 14 / Nivel 4 – Lección 14

Useful links from the Professor – 4

Java Syntax
Level 4, Lesson 14

AVAILABLE

"Hi, Amigo. How's it going?"

"Very good, Professor Noodles. I've already learned about the for and while loops. Now I can go wild without repeating myself."

"That's great. I knew I was the best teacher in the world!"

"You shouldn't listen to all these people who say that only practice matters! Theory underpins everything! What's that you're mumbling? That tasks are more important? Whatever, doesn't matter. I offer you three wonderful articles to help you digest the material even better."

Equals and string comparisons

"Comparing objects is different from comparing primitive data types. You've probably already guessed why. With objects, we're passing a reference. But with primitives, we're passing the value. You'll learn the rest from this fascinating article: "Comparing objects". It also has good examples."

Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 15 / Nivel 4 – Lección 15

Nerd break

Java Syntax
Level 4, Lesson 15

AVAILABLE

"Hi, Amigo. I'm supposed to give you a lesson today, but let's watch some TV before we start, okay?"

×2

Task Java Syntax, level 4, lesson 15

EASY

AVAILABLE

★★★★★

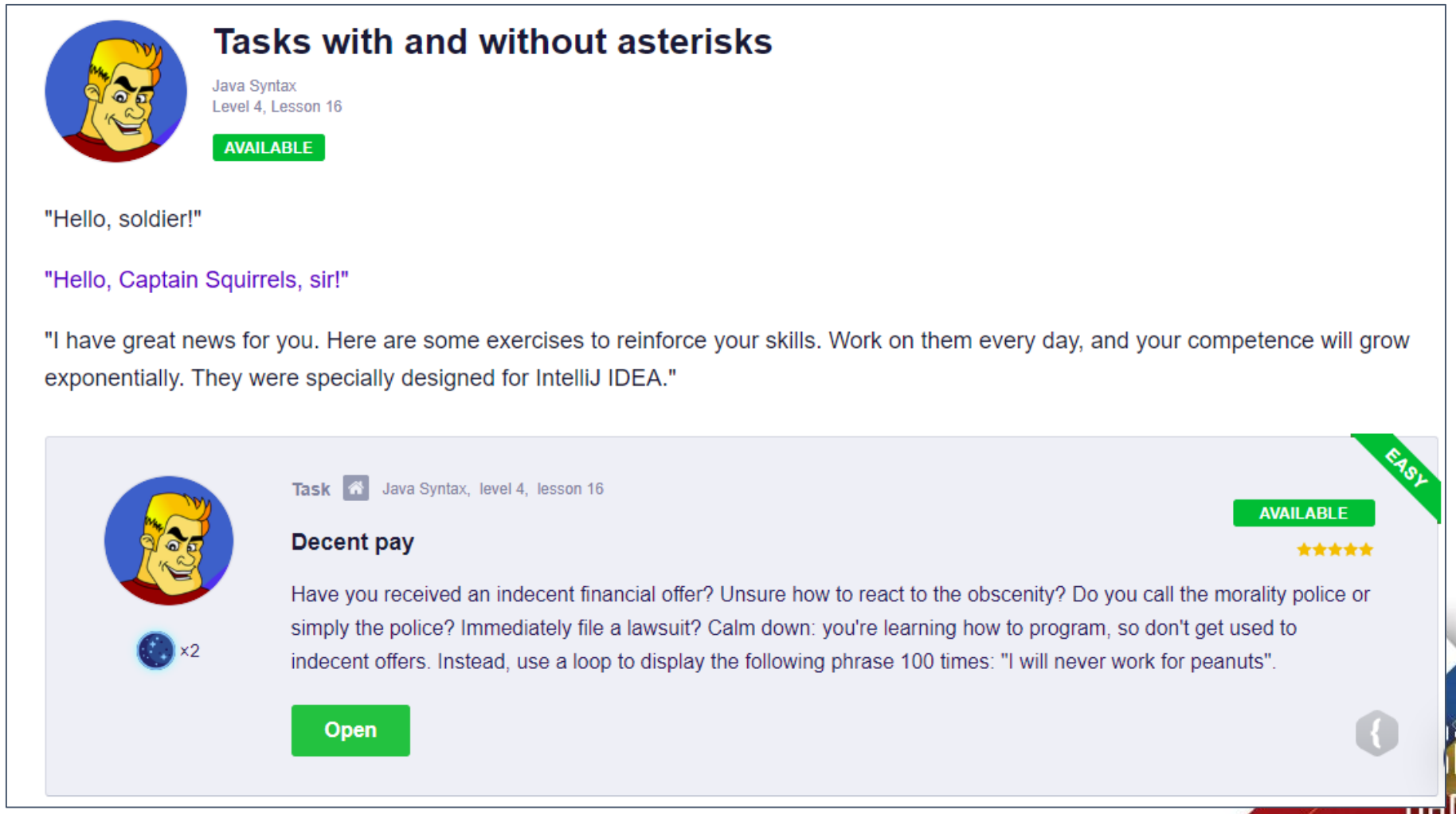
Nerd break

Did you know that before the final power lift and transition to Level 5, you need to relax a bit and watch a video from the "Did You Know?" series. This isn't just entertainment. It's part of the comprehensive curriculum developed by the pedagogical board at the secret CodeGym center. The video is dedicated to the technologies and progress achieved by mankind.

Open

Gamified platform work / Trabajo en plataforma gamificada.

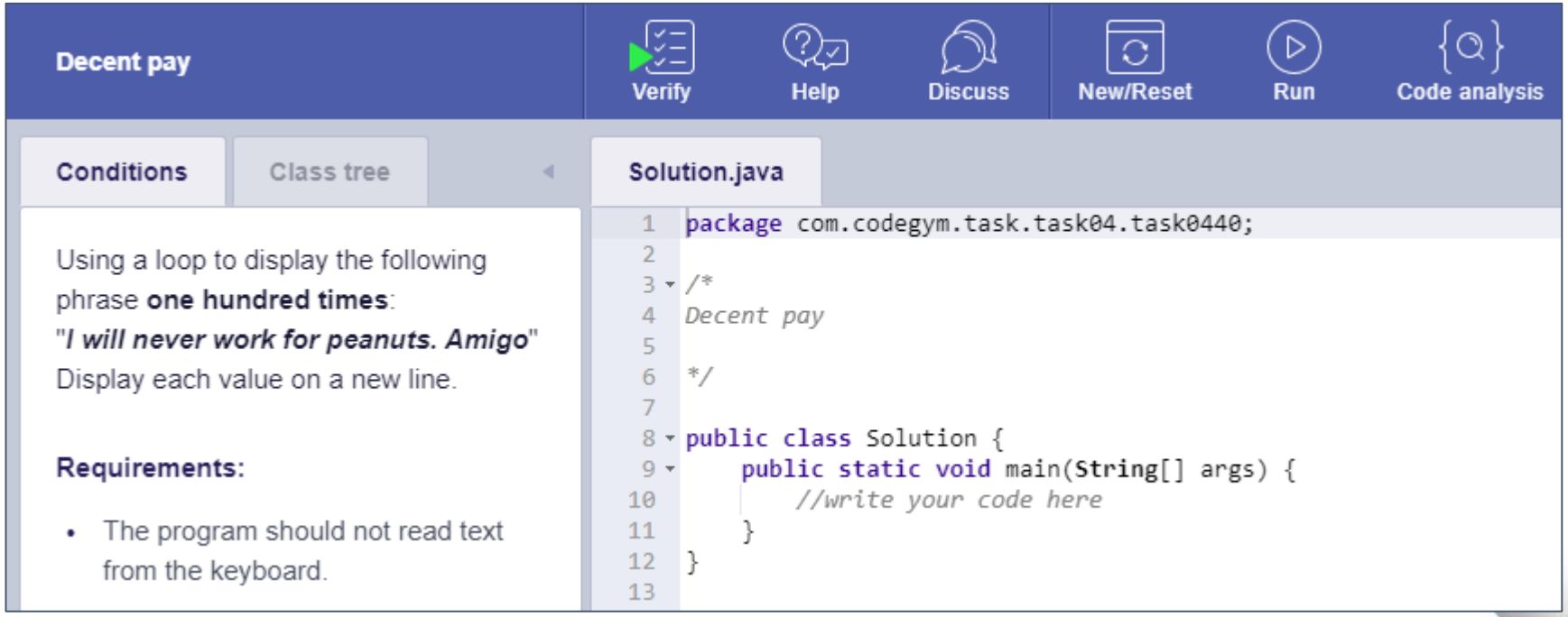
Level 4 - Lesson 16 / Nivel 4 – Lección 16



Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 16 / Nivel 4 – Lección 16

Excercise: **N4_L16_E1**

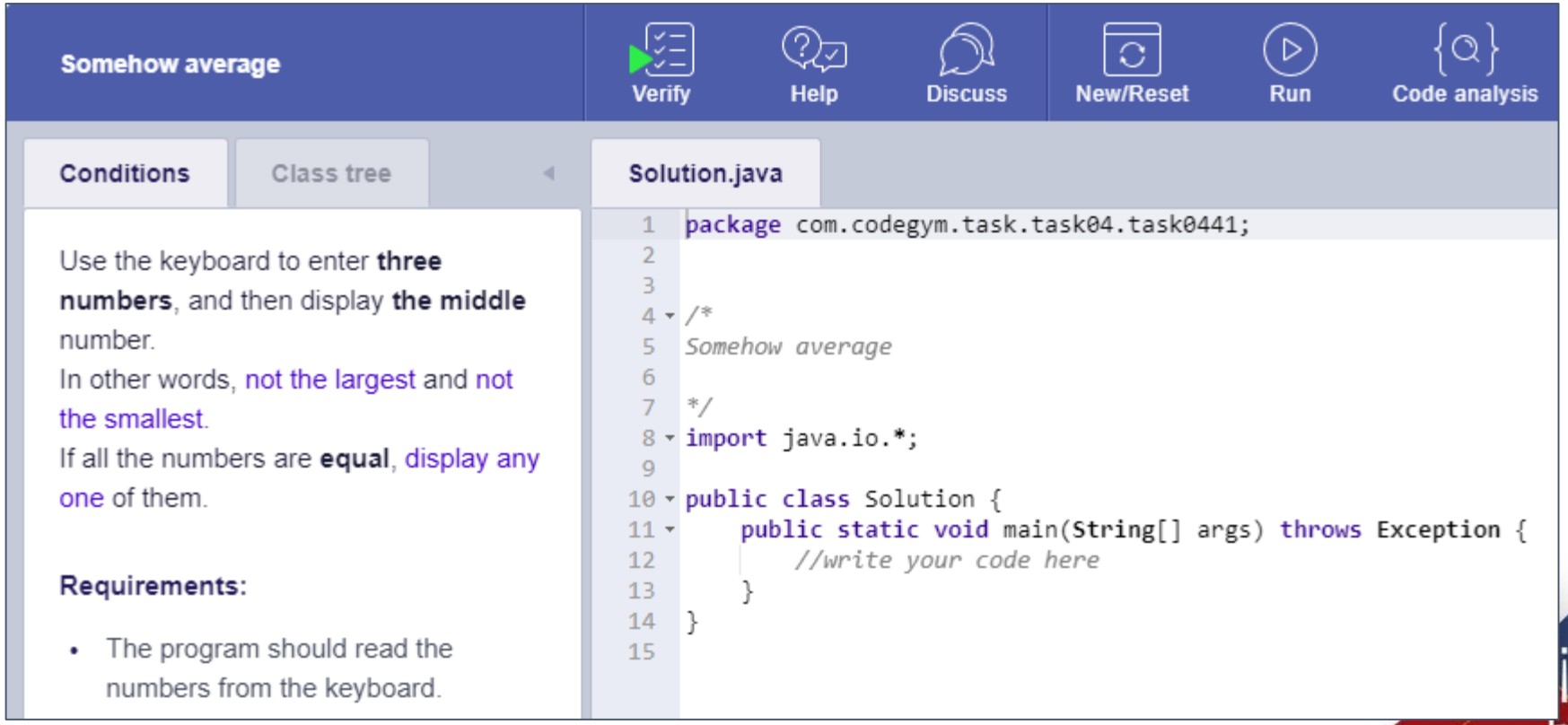


¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 16 / Nivel 4 – Lección 16

Excercise: **N4_L16_E2**

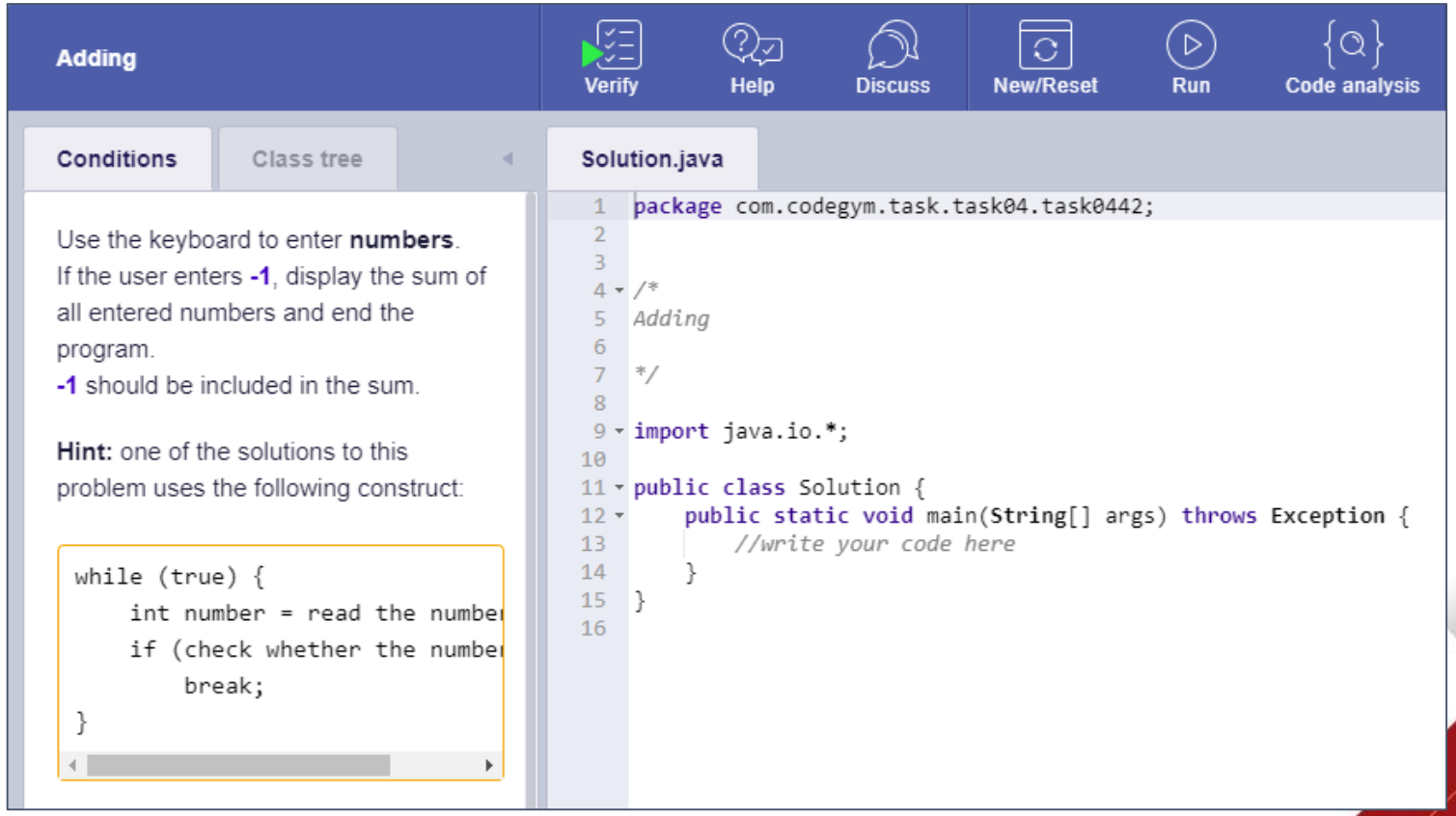


¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 16 / Nivel 4 – Lección 16

Excercise: **N4_L16_E3**

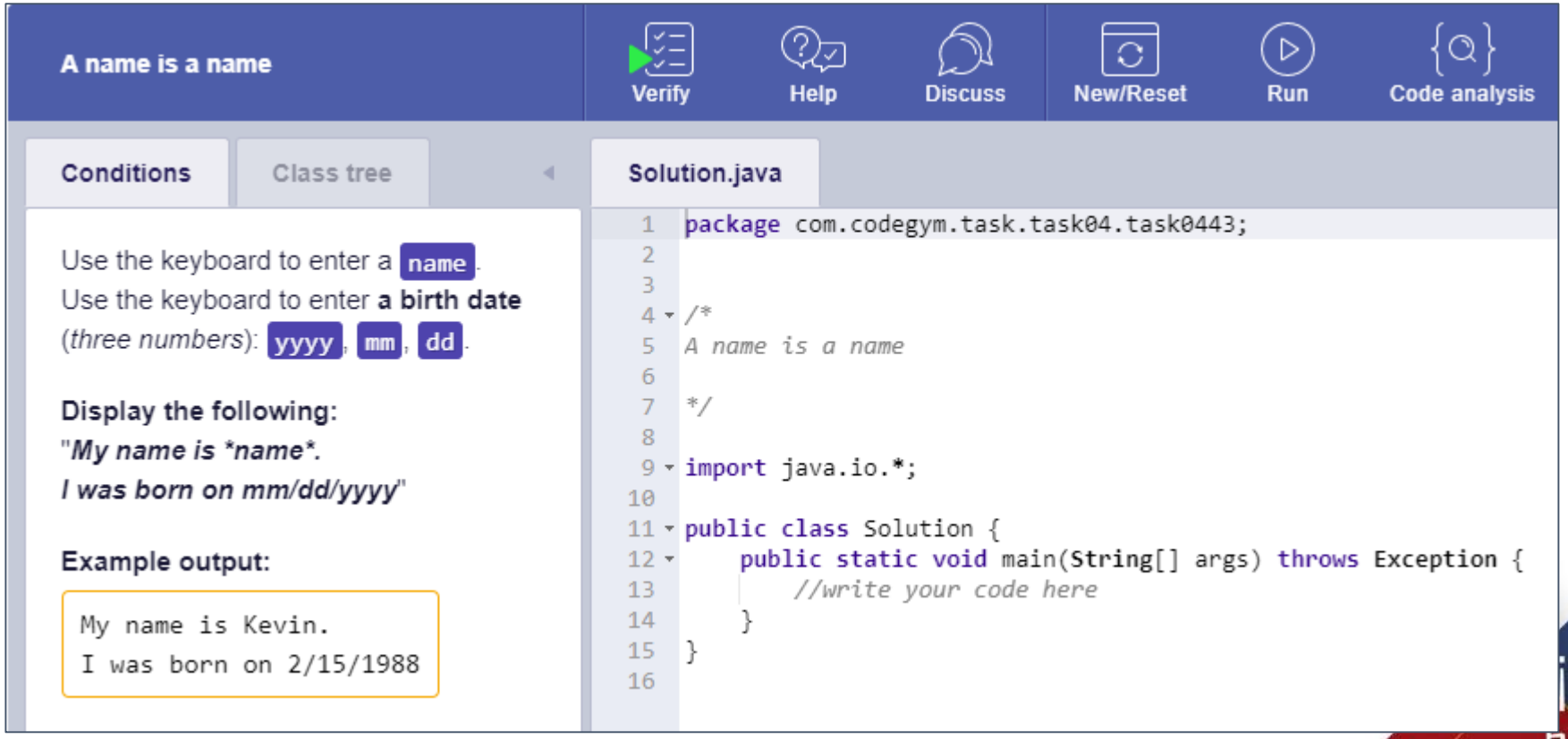


¡Siempre hacia lo alto!

Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 16 / Nivel 4 – Lección 16

Excercise: **N4_L16_E4**



¡Siempre hacia lo alto!

Feedback

¡Siempre hacia lo alto!

Survey

¡Siempre hacia lo alto!

Clarification of doubts

¡Siempre hacia lo alto!

To practice!

¡Siempre hacia lo alto!


PROYECTO
CRUD
PERSISTENCIA
MATRICES
VECTORES
ARREGLOS
CICLOS
MÉTODOS
CONCEPTOS BÁSICOS


GOOD JOB

UNIVERSIDAD SANTO TOMÁS

PRIMER CLAUSTRO UNIVERSITARIO DE COLOMBIA

SECCIONAL TUNJA

VIGILADA MINEDUCACIÓN - SNIES 1732

¡Siempre hacia lo alto!

USTATUNJA.EDU.CO

@santotomastunja