



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 9 DE MAYO-VIGENCIA 5 AÑOS





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
Luis.castellanosg@usantoto.edu.co
Karen.cuervo@usantoto.edu.co

Email:
Luis.castellanosg@usantoto.edu.co
Karen.cuervo@usantoto.edu.co

Phone:
3214582098
3105856930

Topics

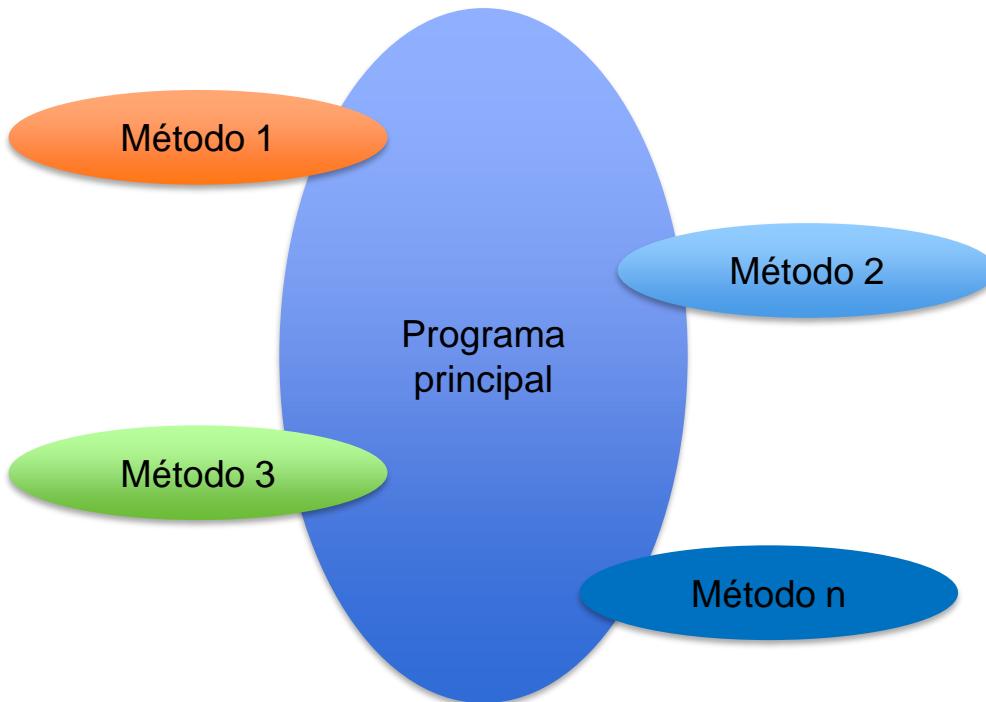
Metodos – funciones / void

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

¡Siempre
hacia lo alto!



Methods / functions / void



¡Siempre
hacia lo alto!



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 específica cada vez que se la llama por su identificador.

Formato

- Modificadores Tipo_de_dato_devuelto Nombre_método(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

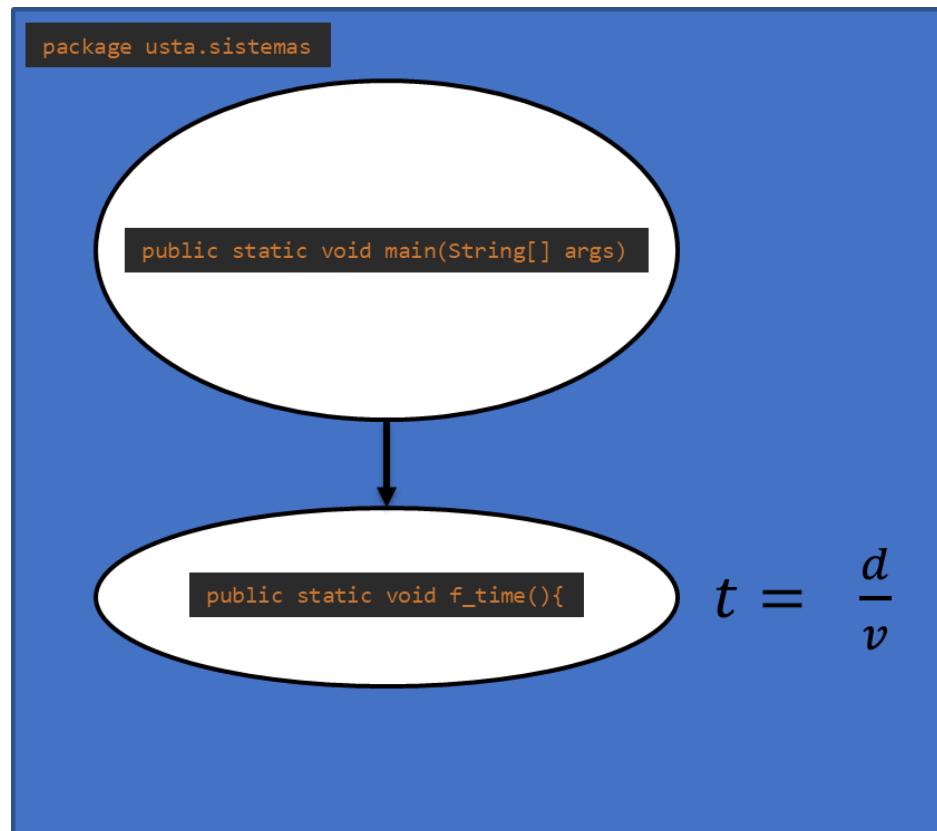
It is required to determine the time it takes a person to get from one city to another by bicycle, considering that it has a constant speed and distance (which is received as a parameter in a **function**).

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.

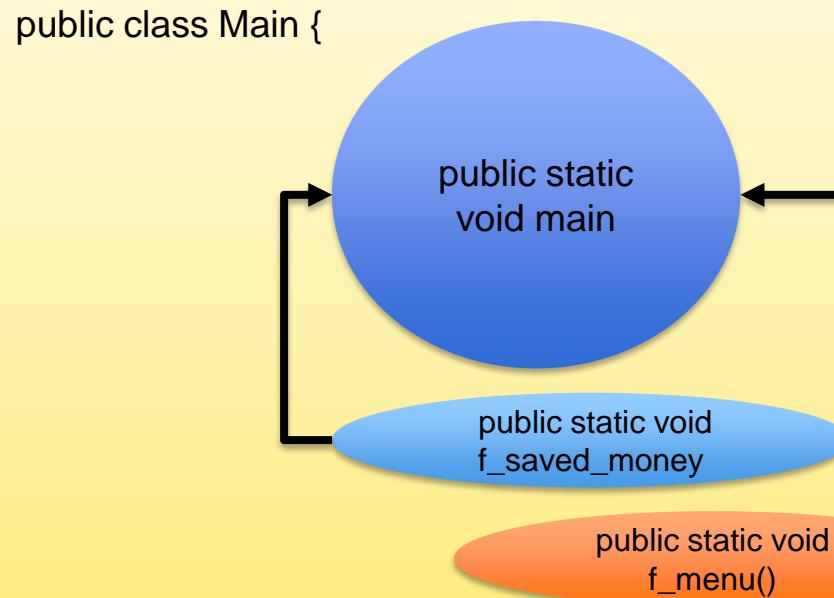
Realice un programa que determine cuánto dinero ahorra una persona en un año si considera que cada semana ahorra 15% de su sueldo (considere cuatro semanas por mes y que no cambia el sueldo). Construya una función para tal fin.



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 $^{\circ}\text{C}$ to $^{\circ}\text{F}$ use the formula: $^{\circ}\text{F} = ^{\circ}\text{C} \times 1.8 + 32$.
- o Convert from $^{\circ}\text{F}$ to $^{\circ}\text{C}$ use the formula: $^{\circ}\text{C} = (^{\circ}\text{F}-32) \div 1.8$.
- o Convert from K to $^{\circ}\text{C}$ use the formula: $^{\circ}\text{C} = \text{K} - 273.15$.
- o Convert from $^{\circ}\text{C}$ to K use the formula: $\text{K} = ^{\circ}\text{C} + 273.15$.

Construir un programa que utilice métodos diferentes para:

- o Convertir de $^{\circ}\text{C}$ a $^{\circ}\text{F}$ use la fórmula: $^{\circ}\text{F} = ^{\circ}\text{C} \times 1.8 + 32$.
- o Convertir de $^{\circ}\text{F}$ a $^{\circ}\text{C}$ use la fórmula: $^{\circ}\text{C} = (^{\circ}\text{F}-32) \div 1.8$.
- o Convertir de K a $^{\circ}\text{C}$ use la fórmula: $^{\circ}\text{C} = \text{K} - 273.15$.
- o Convertir de $^{\circ}\text{C}$ a K use la fórmula: $\text{K} = ^{\circ}\text{C} + 273.15$.



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

LET'S
GO!



¡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.

¡Siempre
hacia lo alto!



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."

¡Siempre
hacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 1 / Nivel 2 – Lección 1

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





Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 1 / Nivel 2 – Lección 1

★ Excercise: N2_L1_E1

Implement the print method

Verify Help Discuss New/Reset Run Code analysis

Conditions Class tree

In the `print` method, display the passed string **4** times. Each time, on a new line.

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 es.codegym.task.task02.task0201;
2
3 /*
4  * Implement the print method
5  */
6
7 public class Solution {
8     public static void main(String[] args) {
9         print("Java is easy to learn!");
10        print("Java opens many opportunities!");
11    }
12
13     public static void print(String s) {
14         //escribe aquí tu código
15     }
16 }
17
```

Siempre
hacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 2 / Nivel 2 – Lección 2



Primitive data types

Sintaxis de Java
Level 2, Lesson 2

AVAILABLE

¡Siempre
hacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 2 / Nivel 2 – Lección 2

Java code	Description
<pre>1 public class Person 2 { 3 String name; 4 int age; 5 }</pre>	A new composite type is declared – <code>Person</code> . Its data is stored in the <code>String</code> (composite type) variable <code>name</code> and (primitive type) <code>int</code> variable <code>age</code>
<pre>1 public class Rectangle 2 { 3 int x, y, width, height; 4 }</pre>	A new composite type is declared – <code>Rectangle</code> . It consists of four <code>int</code> (primitive type) variables.
<pre>1 public class Cat 2 { 3 Person owner; 4 Rectangle territory; 5 int age; 6 String name; 7 }</pre>	A new composite type is declared – <code>Cat</code> . It has the following variables: — <code>owner</code> , composite type <code>Person</code> — <code>territory</code> , composite type <code>Rectangle</code> — <code>age</code> , primitive type <code>int</code> — <code>name</code> , composite type <code>String</code>

empre
ja lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 2 / Nivel 2 – Lección 2

Java code	Description
<pre>1 String s; 2 String s = null;</pre>	Equivalent statements.
<pre>1 Person person; 2 person = new Person(); 3 person = null;</pre>	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.
<pre>1 Cat cat = new Cat(); 2 cat.owner = new Person(); 3 cat.owner.name = "God";</pre>	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

Where does a Person come from?

Verify Help Discuss New/Reset Run Code analysis

Conditions Class tree

object and store a reference to it in the variable **person**.

Hint: Use the following construct to create a **Person** object and assign a reference to that object to the variable **person**:

```
VariableType variableName = new TypeOfObjectBeingCreated();
```

Requirements:

Solution.java

```
1 package es.codegym.task.task02.task0202;
2
3 /*
4  Where does a Person come from?
5  */
6 public class Solution {
7     public static void main(String[] args) {
8         //escribe aquí tu código
9     }
10
11
12     public static class Person {
13         //escribe aquí tu código
14     }
15 }
```

Siempre
hacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 2 / Nivel 2 – Lección 2

★ Excercise: N2_L2_E2

Our first converter!

Verify Help Discuss New/Reset Run Code analysis

Conditions Class tree Solution.java

The `convertCelsiusToFahrenheit(int celsius)` method accepts a value in degrees Celsius. The method should convert the temperature and return the value in degrees Fahrenheit.

The Celsius temperature (TC) and the Fahrenheit temperature (TF) are related by the following relationship:

$$TF = (9 / 5) * TC + 32.$$

Consider this example:
A value of 41 is passed to the `convertCelsiusToFahrenheit` method.

Example output:

105.8

```
1 package es.codegym.task.task01.task0130;
2
3 /*
4 Our first converter!
5 */
6
7 public class Solution {
8     public static void main(String[] args) {
9         System.out.println(convertCelsiusToFahrenheit(41));
10    }
11
12    public static double convertCelsiusToFahrenheit(int celsius) {
13        //escribe aquí tu código
14
15        return 0;
16    }
17
18 }
```

¡Siempre
hacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 2 / Nivel 2 – Lección 2

★ Excercise: N2_L2_E3

Family relations

Solution.java

In the `main` method, create a `Man` object and save a reference to it in the variable `man`.
Also, create a `Woman` object and save a reference to it in the variable `woman`.

Hint: Use the following construct to create a `Woman` object and assign a reference to that object to the variable `woman`:

```
VariableType variableName = new  
TypeOfObjectBeingCreated();  
Save a reference to the previously  
created Woman object in man.wife.  
Save a reference to the previously  
created Man object in woman.husband  
(Hint: woman.husband = man).
```

```
1 package es.codegym.task.task02.task0204;  
2  
3 /*  
4 Family relations  
5 */  
6  
7 public class Solution {  
8     public static void main(String[] args) {  
9         //escribe aquí tu código  
10    }  
11  
12    public static class Man {  
13        public int age;  
14        public int height;  
15        public Woman wife;  
16    }  
17  
18    public static class Woman {  
19        public int age;  
20        public int height;  
21        public Man husband;  
22    }  
23  
24 }
```

¡Siempre
hacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 2 / Nivel 2 – Lección 2

★ Excercise: N2_L2_E4

Pay raise

Verify Help Discuss New/Reset Run Code analysis

Conditions Class tree

In the `public static void hackSalary(int a)` method, increase the salary by **1000** and display the following: "Your salary is: <a+1000> dollars per month." where <a+1000> is the salary increased by **1000**.

Example output for `a = 8000`:

Your salary is: 9000 dollars per month.

Solution.java

```
1 package es.codegym.task.task02.task0205;
2 
3 /*
4 Pay raise
5 */
6 public class Solution {
7     public static void main(String[] args) {
8         hackSalary(7000);
9     }
10 
11     public static void hackSalary(int a) {
12         //escribe aquí tu código
13     }
14 }
15 
```

empre
nacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 2 / Nivel 2 – Lección 2

★ Excercise: N2_L2_E5

Calculate the circumference of a circle

Verify Help Discuss New/Reset Run Code analysis

Conditions Class tree

Display the circumference of a circle, which is calculated using the formula: $C = 2 * \pi * radius$.
The result is a **fractional number** (**double**).
Use **3.14** as the value of **pi**.

Requirements:

- In the `printCircleCircumference` method, you need to display the circle circumference, which is

Solution.java

```
1 package es.codegym.task.task01.task0129;  
2  
3 /*  
4 Calculate the circumference of a circle  
5 */  
6  
7 public class Solution {  
8     public static void main(String[] args) {  
9         printCircleCircumference(5);  
10    }  
11  
12    public static void printCircleCircumference(int radius) {  
13        //escribe aquí tu código  
14    }  
15}  
16}
```

empre
nacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 2 / Nivel 2 – Lección 2

★ Excercise: N2_L2_E6

Part of a calculator

Verify Help Discuss New/Reset Run Code analysis

Conditions Class tree

In the `main` method, declare two `int` variables `a` and `b`. Assign them any values. Display their `sum` and `product` on the screen. Display each value on a new line.

Example output for `a = 5, b = 7:`

12
35

Solution.java

```
1 package es.codegym.task.task02.task0207;
2
3 /*
4 Part of a calculator
5
6 */
7 public class Solution {
8     public static void main(String[] args) {
9         //escribe aquí tu código
10    }
11 }
12 }
```

siempre
hacia lo alto!



To practice!

LET'S
GO!



¡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: $(\text{radius} ^ 2) * \text{PI}$
- Triangle: $(\text{base} * \text{height}) / 2$
- Square: $\text{side} * \text{side}$

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: $(\text{radio}^2)*\text{PI}$

Triangulo: $(\text{base} * \text{altura}) / 2$

Cuadrado: $\text{lado} * \text{lado}$



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

LET'S
GO!



¡Siempre
hacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

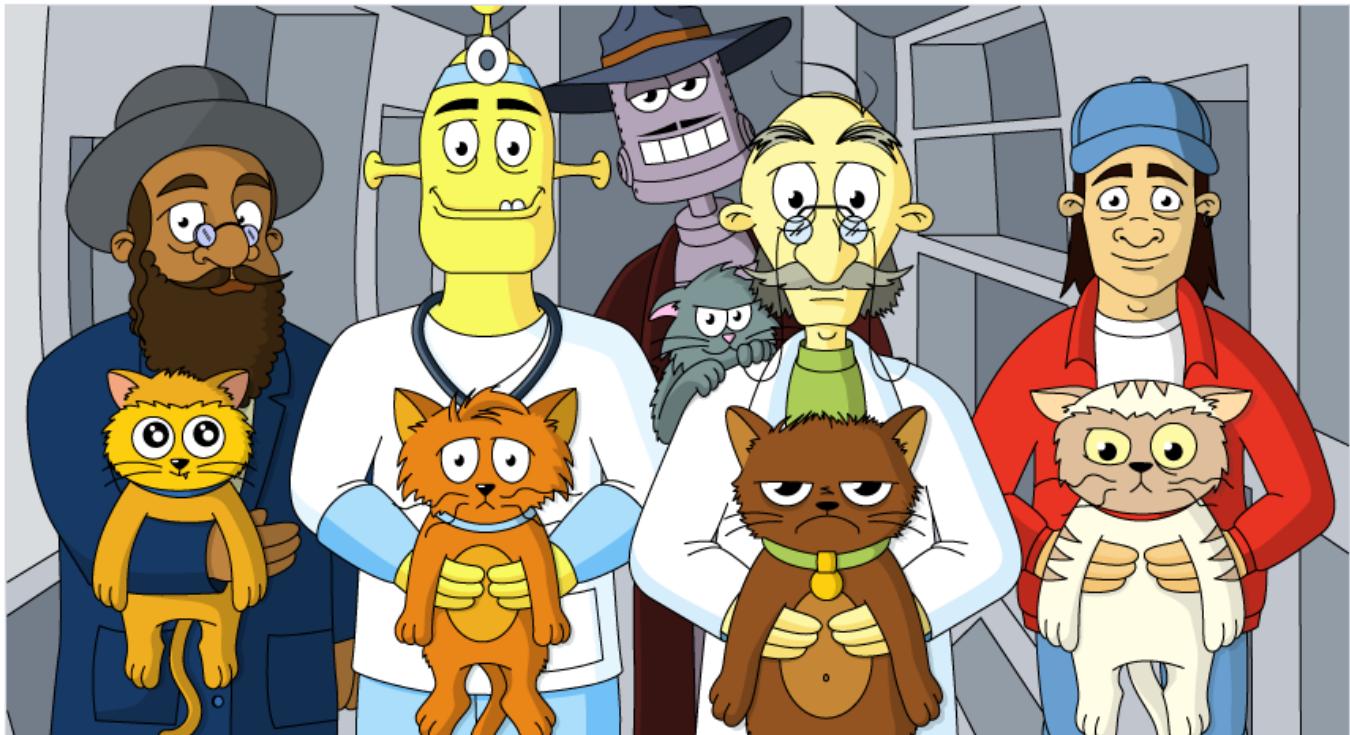
Level 2 - Lesson 3 / Nivel 2 – Lección 3



Creating objects

Sintaxis de Java
Level 2, Lesson 3

AVAILABLE



¡Siempre
hacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 3 / Nivel 2 – Lección 3

Code	Description
1 <code>Cat cat;</code>	Declares a Cat reference variable named <code>cat</code> . The variable <code>cat</code> 's value is null.
1 <code>new Cat();</code>	Creates a Cat object.
1 <code>Cat cat = new Cat();</code>	Creates a Cat reference variable named <code>cat</code> . Creates a new Cat object. Assigns a reference to the newly created object to the variable <code>cat</code> .
1 <code>Cat kitty = new Cat();</code> 2 <code>Cat smokey = new Cat();</code>	Two objects are created. References to them are assigned to two different variables.
1 <code>Cat kitty = new Cat();</code> 2 <code>Cat smokey = new Cat();</code> 3 4 <code>smokey = kitty;</code>	Two objects are created. References to them are assigned to two different variables. Then we set the variable <code>smokey</code> equal to a reference to the object referenced by the variable <code>kitty</code> . Both variables now refer to the first created objects. (Because the second object is no longer referenced anywhere, it is now considered garbage)
1 <code>Cat kitty = new Cat();</code> 2 <code>Cat smokey = null;</code>	One Cat object is created, and a reference to it is assigned to the first variable (<code>kitty</code>). The second variable (<code>smokey</code>) stores an empty (null) reference.

Siempre
... hacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 3 / Nivel 2 – Lección 3

★ Excercise: N2_L3_E1

One cat isn't enough

Verify Help Discuss New/Reset Run Code analysis

Conditions Class tree

Create a **Cat** object **twice**.
Store each **instance in its own variable**.
The variable names must be **different**.

Requirements:

- The program should not display text on the screen.
- The main method should have only two Cat variables.

Solution.java

```
1 package es.codegym.task.task02.task0208;
2
3 /*
4 One cat isn't enough
5
6 */
7 public class Solution {
8     public static void main(String[] args) {
9         //escribe aquí tu código
10    }
11
12     public static class Cat {
13
14    }
15
16 }
```

siempre
hacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 3 / Nivel 2 – Lección 3

★ Excercise: N2_L3_E2

Max, Bella, and Jack

Verify Help Discuss New/Reset Run Code analysis

Conditions Class tree

Create 3 Dog objects. Store each instance in a separate variable. Give them the names "Max", "Bella", and "Jack".

Requirements:

- The program should not display text on the screen.
- The main method should have only three Dog variables.

Solution.java

```
1 package es.codegym.task.task02.task0209;
2
3 /*
4 Max, Bella, and Jack
5 */
6
7 public class Solution {
8     public static void main(String[] args) {
9         Dog dog1 = new Dog();
10        dog1.name = "Max";
11
12        //escribe aquí tu código
13    }
14
15     public static class Dog {
16         public String name;
17     }
18 }
```

siempre
cambia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 3 / Nivel 2 – Lección 3

★ Excercise: N2_L3_E3

Subjective reality

Verify Help Discuss New/Reset Run Code analysis

Conditions Class tree

Solution.java

```
1 package es.codegym.task.task02.task0210;
2
3 /*
4 Subjective reality
5 */
6 public class Solution {
7     public static void main(String[] args) {
8         //escribe aquí tu código
9     }
10 }
```

Write a program that displays: "*If you don't code well, Java will come and eat your memory*".

Requirements:

- The program should display text on the screen.

¡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

```
1  r
2  |public class Variables
3  | |
4  | |
5  | |    private static String TEXT = "The end.";
6  | |
7  | |    public static void main (String[] args)
8  | |
9  | |    {
10 | |        System.out.println("Hi");
11 | |        String s = "Hi!";
12 | |        System.out.println(s);
13 | |        if (args != NULL)
14 | |        {
15 | |            String s2 = s;
16 | |            System.out.println(s2);
17 | |        }
18 | |
19 | |
20 | |
21 | |
22 | |    Variables variables = new Variables();
23 | |    System.out.println(variables.instanceVariable);
24 | |    System.out.println(TEXT);
25 | |
26 | |
27 | |
28 | |    public String instanceVariable;
29 | |
30 | |    public Variables()
31 | |
32 | |        instanceVariable = "Instance variable test.";
33 | |
34 | |
35 | |
36 | |
```

¡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

¡Siempre
hacia lo alto!



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:"

¡Siempre
hacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 5 / Nivel 2 – Lección 5

Example	Explanation
<pre>1 String s1 = new String(); 2 String s2 = "";</pre>	Create two identical empty strings.
<pre>1 int a;</pre>	Create an <code>int</code> variable;
<pre>1 int a = 5;</pre>	Create an <code>int</code> , variable named <code>a</code> and set its value equal to <code>5</code>
<pre>1 int a = 5, b = 6;</pre>	Create an <code>int</code> , variable named <code>a</code> and set its value equal to <code>5</code> Create an <code>int</code> , variable named <code>b</code> and set its value equal to <code>6</code>
<pre>1 int a = 5, b = a + 1;</pre>	Create an <code>int</code> variable named <code>a</code> and set its value equal to <code>5</code> Create an <code>int</code> variable named <code>b</code> and set its value equal to <code>6</code>
<pre>1 Date date = new Date();</pre>	Create a Date object. It is initialized to the current date and time.
<pre>1 boolean isTrue = true;</pre>	Initialize a <code>boolean</code> variable to <code>true</code>
<pre>1 boolean isLess = (5 > 6);</pre>	Assign <code>false</code> to the <code>isLess</code> variable. <code>Boolean</code> variables only accept the values true and false.



Siempre
hacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 5 / Nivel 2 – Lección 5

★ Excercise: N2_L5_E1

The required number

Verify Help Discuss New/Reset Run Code analysis

Conditions Class tree

Solution.java

Comment out as many lines as possible to display the number **19**

Requirements:

- The program should display the number 19.
- Don't change the lines that declare variables.
- Don't change the line responsible for screen output.
- You need to comment out some lines and leave the remaining lines unchanged.

```
1 package es.codegym.task.task02.task0211;
2
3 /*
4 * The required number
5 */
6
7 public class Solution {
8     public static void main(String[] args) {
9         int x = 1;
10        int y = 0;
11
12        y = y + 3 * x;
13        x = x * 2;
14        x = x * 16;
15        y = y + 2 * x;
16        y = y + x;
17
18        System.out.println(y);
19    }
20}
21
```

¡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

```
1 package es.codegym.task.task02.task0212;
2
3 /*
4  * Crazy eights
5  *
6  */
7 public class Solution {
8     public static void main(String[] args) {
9         //escribe aquí tu código
10        Cat cat8 = new Cat();
11        Cat cat9;
12         //escribe aquí tu código
13    }
14
15    public static class Cat {
16    }
17
18 }
19
```

¡Siempre
hacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 5 / Nivel 2 – Lección 5

★ Excercise: N2_L5_E3

Pets need people

Verify Help Discuss New/Reset Run Code analysis

Conditions Class tree

Create **Cat**, **Dog**, **Fish**, and **Woman** objects.
Assign an owner to each animal.

Requirements:

- The program should not display text on the screen.
- In the main method, create Cat, Dog, Fish, and Woman objects, and store references to them in variables.
- In the main method, set the Woman object as the owner of each animal.
- The Cat, Dog, and Fish classes must have only one Woman variable named owner.
- The Woman class must not have variables.

Solution.java

```
1 package es.codegym.task.task02.task0213;
2
3 /*
4 Pets need people
5 */
6
7 public class Solution {
8     public static void main(String[] args) {
9         //escribe aquí tu código
10    }
11
12     public static class Cat {
13         public Woman owner;
14     }
15
16     public static class Dog {
17         public Woman owner;
18     }
19
20     public static class Fish {
21         public Woman owner;
22     }
23
24     public static class Woman {
25     }
26
27 }
```

¡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?"

hacia lo alto!



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

LET'S
GO!



¡Siempre
hacia lo alto!



Survey



¡Siempre
hacia lo alto!



Clarification of doubts



¡Siempre
hacia lo alto!



5° Session CodeGym

LET'S
GO!



¡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."

¡Siempre
hacia lo alto!



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 }
```



Siempre
alcancia lo alto!



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
acia 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."

¡Siempre
hacia lo alto!



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

Even to the moon!

Conditions Class tree

Verify Help Discuss New/Reset Run Code analysis

"Amigo, did you know that **lunar gravity** is about **17%** of **gravity on Earth**?"
"Nope."
"Neither did I. Now this information will be used a lot. To avoid having to manually calculate it each time, implement a **getWeight(int)** method that takes a person's body weight on Earth (*in newtons*), and returns the weight of that person on the moon (*in newtons*).
The method should return a **double**."

Consider this example:
The **getWeight** method is called with the argument **888**.

Example output:
150.96

Solution.java

```
1 package es.codegym.task.task01.task0136;
2
3 /*
4 Even to the moon!
5
6 */
7
8 public class Solution {
9     public static void main(String[] args) {
10         System.out.println(getWeight(888));
11     }
12
13     public static double getWeight(int earthWeight) {
14         // escribe aquí tu código
15     }
16 }
```

¡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

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

```
1 package es.codegym.task.task02.task0216;
2
3 /*
4 Minimum of three numbers
5
6 */
7 public class Solution {
8     public static int min(int a, int b, int c) {
9         //escribe aquí tu código
10    }
11
12    public static void main(String[] args) throws Exception {
13        System.out.println(min(1, 2, 3));
14        System.out.println(min(-1, -2, -3));
15        System.out.println(min(3, 5, 3));
16        System.out.println(min(5, 5, 10));
17    }
18
19 }
```

¡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 Solution.java

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)`.

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

¡Siempre
hacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 8 / Nivel 2 – Lección 8

★ Excercise: N2_L8_E5

Repetition is the mother of all learning

Verify Help Discuss New/Reset Run Code analysis

Conditions Class tree

Implement the `print3` method. The method should display the passed string **3 times**. Each time, on a new line.

Requirements:

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

Solution.java

```
1 package es.codegym.task.task02.task0218;
2
3 /*
4 Repetition is the mother of all learning
5 */
6 public class Solution {
7     public static void print3(String s) {
8         //escribe aquí tu código
9     }
10
11 }
12
13 public static void main(String[] args) {
14     print3("I love you!");
15 }
16 }
```

¡Siempre
hacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 8 / Nivel 2 – Lección 8

★ Excercise: N2_L8_E6

Print three times

Verify Help Discuss New/Reset Run Code analysis

Conditions Class tree

Implement the `print3` method. The method should **display the passed string (word) three times**, but on the same line.
Words must be separated by a space and should not merge into one.

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 es.codegym.task.task02.task0219;
2
3 /*
4 Print three times
5 */
6 public class Solution {
7     public static void print3(String s) {
8         //escribe aquí tu código
9     }
10
11
12
13     public static void main(String[] args) {
14         print3("window");
15         print3("file");
16     }
17 }
```

¡Siempre hacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 9 / Nivel 2 – Lección 9

 **Full class name**
Sintaxis de Java
Level 2, Lesson 9
AVAILABLE



¡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
1 String	java.lang	java.lang.String
1 FileInputStream	java.io	java.io.FileInputStream
1 ArrayList	java.util	java.util.ArrayList
1 IOException	java.io	java.io.IOException;

¡Siempre
hacia lo alto!



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 }
```

empre
ia lo alto!



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 }
```



¡Siempre
hacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 10 / Nivel 2 – Lección 10



Useful links from the Professor – 2

Sintaxis de Java
Level 2, Lesson 10

AVAILABLE



¡Siempre
hacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 2 - Lesson 11 / Nivel 2 – Lección 11

 Game of Codes: a song of bytes and wire

Sintaxis de Java
Level 2, Lesson 11

AVAILABLE

"Hi, Amigo! I hope you're ready for a hard and exhausting break."


JavaZone 2014: Game of Codes

Ver más tarde Compartir

GAME OF CODES
A SONG OF BYTES AND WIRE

¡Siempre
hacia lo alto!

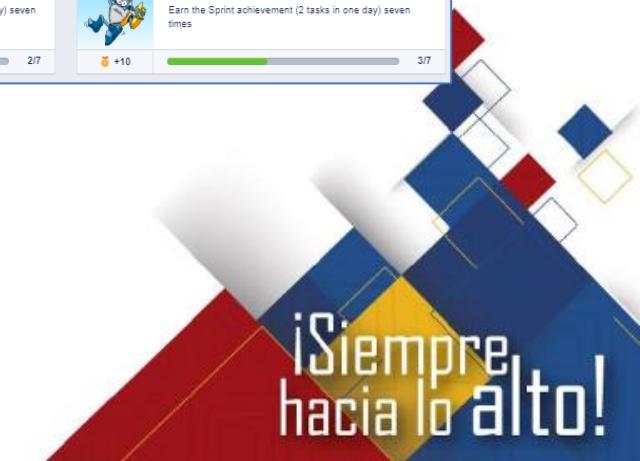


Working document with full level 2! / Documento de trabajo con nivel 2 completo!



Upload it to your drive folder: /
Súbelo a tu carpeta en drive.:

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



¡Siempre
hacia lo alto!



Feedback

LET'S
GO!



¡Siempre
hacia lo alto!



Survey





Clarification of doubts



¡Siempre
hacia lo alto!



To practice!

LET'S
GO!



¡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 matrícula. 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 cuento tiempo se va a demorar procesando el producto el total de los vueltos.



6° Session CodeGym

LET'S
GO!



¡Siempre
hacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

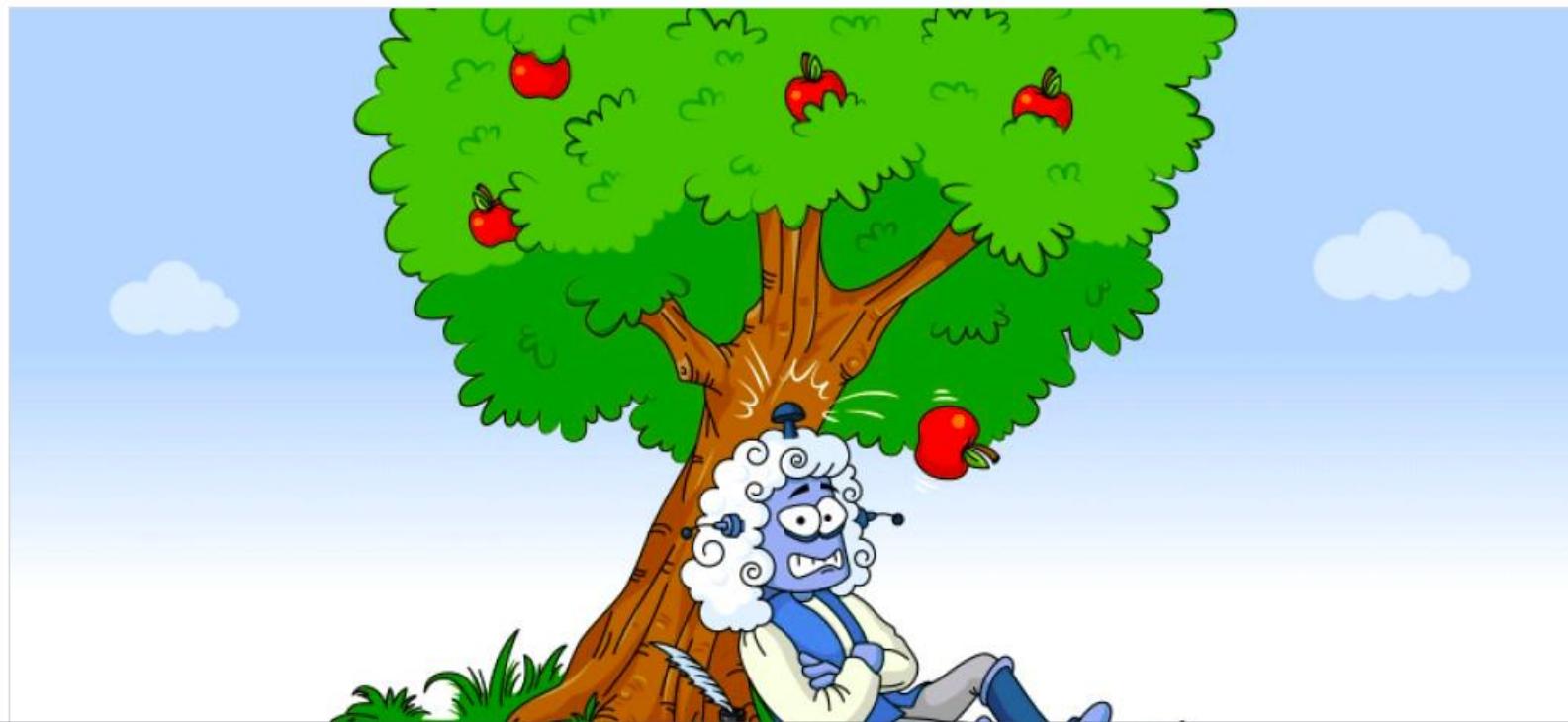
Level 3 - Lesson 0 / Nivel 3 – Lección 0



You've leveled up!

Java Syntax
Level 3, Lesson 0

AVAILABLE



¡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

Dividing is good

Verify Help Discuss New/Reset Run Code analysis

Conditions Class tree

Implement `public static void div(int a, int b)`.
The method should divide the first number by the second, and display the result of dividing `a` by `b`.
The displayed result should be an integer.

Requirements:

- The div method must divide a by b.
- The div method must display the result of the division.
- The main method should call the div method 3 times.

Solution.java

```
1 package com.codegym.task.task03.task0301;
2
3 /*
4 Dividing is good
5 */
6
7 public class Solution {
8     public static void main(String[] args) {
9         div(6, 3);
10        div(10, 6);
11        div(2, 4);
12    }
13
14    public static void div(int a, int b) {
15        //write your code here
16    }
17
18 }
19
```

Siempre
hacia lo alto!



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

Display right away

Verify Help Discuss New/Reset Run Code analysis

Conditions Class tree

Add the `public static void printString(String s)` method, and write code that will make it display the passed string.

Requirements:

- Add a `printString` method that takes a `String` argument.
- The `printString` method must be `void`.
- The `printString` method must be `static`.

Solution.java

```
1 package com.codegym.task.task03.task0302;
2
3 /*
4 * Display right away
5 */
6
7 public class Solution {
8     // write your code here
9
10    public static void main(String[] args) {
11        printString("Hello, Amigo!");
12    }
13
14 }
```

iSiempre
hacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 3 - Lesson 3 / Nivel 3 – Lección 3

★ Excercise: N3_L3_E2

Currency exchange

Verify Help Discuss New/Reset Run Code analysis

Conditions Class tree

Solution.java

```
1 package com.codegym.task.task03.task0303;
2 /*
3 * Currency exchange
4 */
5
6 public class Solution {
7     public static void main(String[] args) {
8         //write your code here
9     }
10
11     public static double convertEurToUsd(int eur, double exchangeRate) {
12         //write your code here
13     }
14 }
15
16
17 
```

Write code for the **convertEurToUsd** method, which converts **euros** to **dollars** at a given exchange rate. Use a **return** statement to return the result from the **convertEurToUsd** method. **Example:** `return 123*435;` Call the **convertEurToUsd** method twice in the **main** method with **any** arguments. Display the results, each time on a new line.

Hint:
The result is calculated using the following formula: **US dollars = euros * exchange rate**

Requirements:

- The **convertEurToUsd** method must multiply the euro amount by the exchange rate and return the result.

iSiempre
hacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 3 - Lesson 3 / Nivel 3 – Lección 3

★ Excercise: N3_L3_E3

Task with percentages

Verify Help Discuss New/Reset Run Code analysis

Conditions Class tree

Solution.java

```
1 package com.codegym.task.task03.task0304;
2
3 /*
4 Task with percentages
5
6 */
7
8 public class Solution {
9     public static double addTenPercent(int i) {
10         //write your code here
11     }
12
13     public static void main(String[] args) {
14         System.out.println(addTenPercent(9));
15     }
16
17 }
```

Write the code for the **addTenPercent** method, which increases the passed integer by **10%**.
Use the **return** statement to return the result from the **addTenPercent** method.

Consider this example:

```
return 123 * 435;
```

Requirements:

- The **addTenPercent** method should increase the passed number by 10% percent.

¡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:"



Task Java Syntax, level 3, lesson 4

AVAILABLE

EASY



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

Red scare

Verify Help Discuss New/Reset Run Code analysis

Conditions Class tree

Display your birth date in the form:
MAY 1 2012

Requirements:

- The displayed text should contain the month, day and year, separated by spaces.
- The name of the month should be printed in English in uppercase letters.

Solution.java

```
1 package com.codegym.task.task03.task0305;
2
3 /*
4 Red scare
5
6 */
7
8 public class Solution {
9     public static void main(String[] args) {
10         //write your code here
11     }
12 }
13
```

¡Siempre hacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 3 - Lesson 4 / Nivel 3 – Lección 4

★ Excercise: N3_L4_E2

Parenthetical brainteaser

Verify Help Discuss New/Reset Run Code analysis

Conditions Class tree

Arrange the brackets correctly (differently) so that the number 382 is displayed on the screen.

Requirements:

- The program must not read data from the keyboard.
- The main method must contain only one call to System.out.println.
- The sequence of digits and arithmetic operations can not be changed.

Solution.java

```
1 package com.codegym.task.task03.task0306;
2
3 /*
4 Parenthetical brainteaser
5
6 */
7
8 public class Solution {
9     public static void main(String[] args) {
10         System.out.println((2 * 3) + 4 * 5 + (6 * 7));
11     }
12 }
```

¡Siempre
hacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 3 - Lesson 4 / Nivel 3 – Lección 4

★ Excercise: N3_L4_E3

Hello, StarCraft!

Verify Help Discuss New/Reset Run Code analysis

Conditions Class tree

Create **5 Zerg** units, **3 Protoss** units and **4 Terran** units. Give them all **unique names**.

Requirements:

- Don't change the Zerg, Protoss, and Terran classes.
- Create 5 Zerg objects and name each of them.
- Create 3 Protoss objects and name each of them.
- Create 4 Terran objects and name each of them.

Solution.java

```
1 package com.codegym.task.task03.task0307;
2
3 /*
4 Hello, StarCraft!
5 */
6
7 public class Solution {
8     public static void main(String[] args) {
9         //write your code here
10    }
11
12    public static class Zerg {
13        public String name;
14    }
15
16    public static class Protoss {
17        public String name;
18    }
19
20    public static class Terran {
21        public String name;
22    }
23
24 }
25 }
```

¡Siempre
hacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 3 - Lesson 4 / Nivel 3 – Lección 4

★ Excercise: N3_L4_E4

Product of 10 numbers

Verify Help Discuss New/Reset Run Code analysis

Conditions Class tree

Solution.java

```
1 package com.codegym.task.task03.task0308;
2
3 /*
4 Product of 10 numbers
5
6 */
7
8 public class Solution {
9     public static void main(String[] args) {
10         //write your code here
11     }
12 }
13 }
```

Display the product of the 10 numbers from 1 to 10.
The result is a single number.

Hint:
it will be three million and change.

Requirements:

- The program must display an integer.
- The main method should call the System.out.println method.

Siempre
nacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 3 - Lesson 4 / Nivel 3 – Lección 4

★ Excercise: N3_L4_E5

Sum of 5 numbers

Verify Help Discuss New/Reset Run Code analysis

Conditions Class tree

Solution.java

```
1 package com.codegym.task.task03.task0309;
2
3 /*
4  Sum of 5 numbers
5
6 */
7
8 public class Solution {
9     public static void main(String[] args) {
10         //write your code here
11     }
12 }
13
```

Display the sum of numbers from 1 to 5, line by line (*there should be 5 lines*):

1
1+2=3
1+2+3=6
...

Example output:

1
3
6
...

¡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:"

iSiempre
hacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 3 - Lesson 5 / Nivel 3 – Lección 5

★ Excercise: N3_L5_E1

Fill a pool with water

Verify Help Discuss New/Reset Run Code analysis

Conditions Class tree Solution.java

Amigo, today our task is to fill the ship's pool. Calculate how many liters of water are needed to fill the pool all the way up. The pool is known to have linear dimensions **a** x **b** x **c**, given in meters. These dimensions are passed to the **getVolume** method. The method should return the number of liters of water needed to fill the pool.

Consider this example:
The **getVolume** method is called with the arguments **25**, **5**, and **2**.

Example output:

```
250000
```

1 package com.codegym.task.task01.task0134;
2
3 /*
4 Fill a pool with water
5 */
6
7 public class Solution {
8 public static void main(String[] args) {
9 System.out.println(getVolume(25, 5, 2));
10 }
11
12 public static long getVolume(int a, int b, int c) {
13 //write your code here
14 }
15
16 }

¡Siempre
hacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 3 - Lesson 5 / Nivel 3 – Lección 5

★ Excercise: N3_L5_E2

Printing strings

Verify Help Discuss New/Reset Run Code analysis

Conditions Class tree Solution.java

Implement the `public static void writeToConsole(String s)` method, which adds "printing:" to the beginning of the string and displays the altered string.

Example output for "Hello, World!":

printing: Hello, World!

Requirements:

- Add a `writeToConsole` method that takes a String argument.

```
1 package com.codegym.task.task03.task0311;
2
3 /*
4 Printing strings
5
6 */
7
8 public class Solution {
9     public static void main(String[] args) {
10         writeToConsole("Hello, World!");
11     }
12
13     public static void writeToConsole(String s) {
14         //write your code here
15     }
16
17 }
```

¡Siempre
hacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 3 - Lesson 5 / Nivel 3 – Lección 5

★ Excercise: N3_L5_E3

Time conversion

Verify Help Discuss New/Reset Run Code analysis

Conditions Class tree

Add a `public static int convertToSeconds(int hour)` method that converts hours to seconds. Call it twice in the `main` method with any arguments. Display the results, each time on a new line.

Requirements:

- Add the `convertToSeconds` method. It must be public static, and take and return ints.

Solution.java

```
1 package com.codegym.task.task03.task0312;
2
3 /*
4 Time conversion
5 */
6
7 public class Solution {
8     //write your code here
9
10    public static void main(String[] args) {
11        //write your code here
12    }
13
14 }
15
```

Siempre
hacia lo alto!



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."



empre
nacía lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 3 - Lesson 6 / Nivel 3 – Lección 6

★ Excercise: N3_L6_E1

Sam I Am

Verify Help Discuss New/Reset Run Code analysis

Conditions Class tree

Display all possible combinations of the words "Sam", "I", "Am".

Hint: there are 6 combinations.
Display each combination on a new line.
Do not separate the words.

Consider this example:

IAmSam
AmSamI
...

Solution.java

```
1 package com.codegym.task.task03.task0313;
2
3 /*
4 Sam I Am
5 */
6 public class Solution {
7     public static void main(String[] args) {
8         //write your code here
9     }
10 }
11
12 }
```

empre
nacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 3 - Lesson 6 / Nivel 3 – Lección 6

★ Excercise: N3_L6_E2

Multiplication table

Verify Help Discuss New/Reset Run Code analysis

Conditions Class tree

Display a **10 x 10** multiplication table in the following form:

1	2	3	4	...
2	4	6	8	...
3	6	9	12	...
4	8	12	16	...
...				

Solution.java

```
1 package com.codegym.task.task03.task0314;
2
3 /*
4 Multiplication table
5 */
6
7
8 public class Solution {
9     public static void main(String[] args) {
10         //write your code here
11     }
12 }
13 }
```

¡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

```
1 package com.codigym.task.task03.task0315;
2
3 //*
4
5 public class Solution {
6     public static void main(String[] args) {
7         //write your code here
8     }
9
10    public static class Red {
11        public Red() {
12            System.out.println("Red");
13        }
14    }
15
16    public static class Orange {
17        public Orange() {
18            System.out.println("Orange");
19        }
20    }
21
22    public static class Yellow {
23        public Yellow() {
24            System.out.println("Yellow");
25        }
26    }
27
28    public static class Green {
29        public Green() {
30            System.out.println("Green");
31        }
32    }
33
34    public static class Blue {
35        public Blue() {
36            System.out.println("Blue");
37        }
38    }
39
40    public static class Indigo {
41        public Indigo() {
42            System.out.println("Indigo");
43        }
44    }
45
46    public static class Violet {
47        public Violet() {
48            System.out.println("Violet");
49        }
50    }
51
52
53
54
55 }
```

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.

```
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55 }

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");
    }
}
```

iSiempre
hacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 3 - Lesson 6 / Nivel 3 – Lección 6

★ Excercise: N3_L6_E4

Escaping characters

Verify Help Discuss New/Reset Run Code analysis

Conditions Class tree

Solution.java

```
1 package com.codegym.task.task03.task0316;
2
3 /*
4 Escaping characters
5
6 */
7
8 public class Solution {
9     public static void main(String[] args) {
10         //write your code here
11     }
12 }
13
```

Display the following text (two lines):

This is a Windows path:
"C:\\Program
Files\\Java\\jdk1.8.0_172\\bin"
This is a Java string:
\\\"C:\\\\Program
Files\\\\Java\\\\jdk1.8.0_172\\\\bin\\\\"

Hint:

\" – Insert a double quote character in the text at this point.
\\\\ – Insert a backslash character in the text at this point.

¡Siempre
hacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 3 - Lesson 6 / Nivel 3 – Lección 6

★ Excercise: N3_L6_E5

The way of the Samurai

Verify Help Discuss New/Reset Run Code analysis

Conditions Class tree Solution.java

Display the following: 日本語

Requirements:

- The program should output text.
- The text should begin with "日".
- The text should end with "語".
- The text should consist of 3 symbols.
- The displayed text must match the task conditions.

```
1 package com.codegym.task.task03.task0317;
2
3 /*
4 * The way of the Samurai
5 */
6
7
8 public class Solution {
9     public static void main(String[] args) {
10         //write your code here
11     }
12 }
13
```

iSiempre
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

```
1 InputStream inputStream = System.in;
2 Reader inputStreamReader = new InputStreamReader(inputStream);
3 BufferedReader bufferedReader = new BufferedReader(inputStreamReader);
4
5 String name = bufferedReader.readLine(); //Read a string from the keyboard
6 String sAge = bufferedReader.readLine(); //Read a string from the keyboard
7 int nAge = Integer.parseInt(sAge); //Convert the string to a number.
```

Example 2

A more compact version of the previous example:

```
1 BufferedReader reader = new BufferedReader(new InputStreamReader(System.in));
2
3 String name = reader.readLine();
4 String sAge = reader.readLine();
5 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();
```



¡Siempre
hacia lo alto!



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



mpre
nacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 3 - Lesson 8 / Nivel 3 – Lección 8

★ Excercise: N3_L8_E1

Plan to conquer the world

Verify Help Discuss New/Reset Run Code analysis

Conditions Class tree

Enter the **number** and **name** from the keyboard. Display the following string:
<name> will take over the world in
<number> years. Mwa-ha-ha!

Here's an example:

Kevin will take over the world
in 8 years. Mwa-ha-ha!

The order in which the data is input matters a lot.

Solution.java

```
1 package com.codegym.task.task03.task0318;
2
3 /*
4 Plan to conquer the world
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 }
```

empre
hacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 3 - Lesson 8 / Nivel 3 – Lección 8

★ Excercise: N3_L8_E2

Predictions

Verify Help Discuss New/Reset Run Code analysis

Conditions Class tree

Solution.java

```
1 package com.codegym.task.task03.task0319;  
2  
3 /*  
4 Predictions  
5 */  
6  
7 import java.io.*;  
8  
9  
10 public class Solution {  
11     public static void main(String[] args) throws Exception {  
12         //write your code here  
13     }  
14 }  
15
```

Use the keyboard to separately enter the **name**, **number1**, and **number2**. Display the following phrase:
<name> will receive <number1> in <number2> years.

Here's an example:

Nick will receive 10000 in 5 years.

empre
hacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 3 - Lesson 8 / Nivel 3 – Lección 8

★ Excercise: N3_L8_E3

The humble programmer

Verify Help Discuss New/Reset Run Code analysis

Conditions Class tree

Solution.java

```
1 package com.codegym.task.task03.task0320;
2
3
4 /*
5 * The humble programmer
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 }
16
```

Use the keyboard to enter a name and display the following:
<name> makes \$120,000 a year. Ha-ha-ha!

For example:

Sara makes \$120,000 a year.
Ha-ha-ha!

Requirements:

- The program should output text.

¡Siempre
nacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 3 - Lesson 8 / Nivel 3 – Lección 8

★ Excercise: N3_L8_E4

Don't think about seconds...

Verify Help Discuss New/Reset Run Code analysis

Conditions Class tree

Write code that calculates **how many seconds** have passed since 3:00 PM when the clock says it's 3:30 PM. Display the result.

Requirements:

- The program should output text.
- The displayed text should be a positive integer.

Solution.java

```
1 package com.codegym.task.task01.task0133;
2
3 /*
4 Don't think about seconds...
5 */
6
7
8 public class Solution {
9     public static void main(String[] args) {
10         int secondsAfter15 = 0;
11         System.out.println(secondsAfter15);
12     }
13 }
```

siempre
hacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 3 - Lesson 8 / Nivel 3 – Lección 8

★ Excercise: N3_L8_E5

More conversions

Verify Help Discuss New/Reset Run Code analysis

Conditions Class tree

Solution.java

```
1 package com.codegym.task.task01.task0131;
2
3 /*
4 More conversions
5 */
6
7 public class Solution {
8     public static void main(String[] args) {
9         System.out.println(getFeetFromInches(28));
10    }
11
12
13     public static int getFeetFromInches(int inches) {
14         //write your code here
15     }
16 }
```

Implement the `getFeetFromInches(int inches)` method. The method takes the number of inches. Your task is to make the method return the number of full feet represented by the variable inches. (1 foot = 12 in).

Consider this example:
The `getFeetFromInches` method is called with the argument 243.

Example output:

20

¡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.

```
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 }
```

Siempre
hacia lo alto!



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:"

¡Siempre
hacia lo alto!



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
nacía 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

```
1 package com.codegym.task.task01.task0132;
2
3 /*
4 Sum of the digits of a three-digit number
5 */
6
7 public class Solution {
8     public static void main(String[] args) {
9         System.out.println(sumDigitsInNumber(546));
10    }
11
12    public static int sumDigitsInNumber(int number) {
13        //write your code here
14    }
15 }
```

¡Siempre
hacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 3 - Lesson 12 / Nivel 3 – Lección 12

★Excercise: N3_L12_E2

Mercantile intentions

Conditions Class tree

Verify Help Discuss New/Reset Run Code analysis

Conditions

Solution.java

```
1 package com.codegym.task.task03.task0324;
2
3 /*
4 Mercantile intentions
5 */
6
7
8 public class Solution {
9     public static void main(String[] args) {
10         //write your code here
11     }
12 }
13
```

Display ten times the phrase "*I want a big salary, and that's why I'm studying Java*"

Requirements:

- The program must not read data from the keyboard.
- The program should output text.

¡Siempre hacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 3 - Lesson 12 / Nivel 3 – Lección 12

★Excercise: N3_L12_E3

Financial expectations

Conditions **Class tree**

Use the keyboard to enter the number **n**.
Display the phrase "**I will earn \$n per hour**" on the screen.

For example:

I will earn \$100 per hour

Requirements:

- The program should output text.

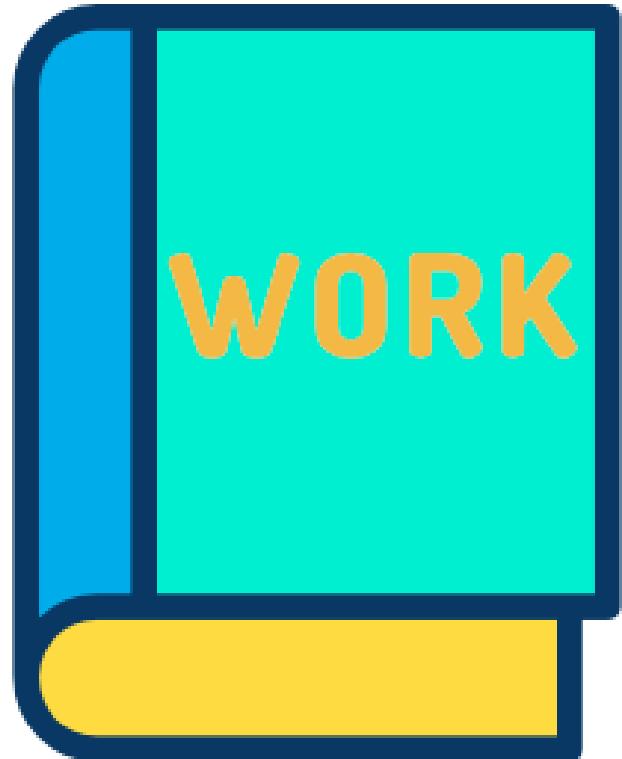
Solution.java

```
1 package com.codegym.task.task03.task0325;
2
3 import java.io.*;
4
5 /*
6 Financial expectations
7 */
8
9
10 public class Solution {
11     public static void main(String[] args) throws Exception {
12         //write your code here
13     }
14 }
```

Siempre
nacía lo alto!



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



Upload it to your drive folder: /
Súbelo a tu carpeta en drive.:

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



iSiempre
hacia lo alto!



Feedback

LET'S
GO!



¡Siempre
hacia lo alto!



Survey



¡Siempre
hacia lo alto!



Clarification of doubts



¡Siempre
hacia lo alto!



To practice!

LET'S
GO!



¡Siempre
hacia lo alto!



P1TX: Exercise

Create ...

Crear...



7° Session CodeGym

LET'S
GO!



¡Siempre
hacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 0 / Nivel 4 – Lección 0



You've leveled up!

Java Syntax
Level 4, Lesson 0

AVAILABLE





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."

iSiempre
hacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 1 / Nivel 4 – Lección 1

★ Excercise: N4_L1_E1

This age doesn't work for me...

Verify Help Discuss New/Reset Run Code analysis

Conditions Class tree

Think about what the program is doing. Fix the programming **error** so that **person.age** changes value.

Hint: carefully review the **adjustAge** method.

Requirements:

- The program should display text on the screen.
- The Person class should have a public int field called age.
- The Person class's adjustAge method should display text on the screen.

Solution.java

```
1 package com.codegym.task.task04.task0401;
2 /*
3 * This age doesn't work for me...
4 */
5
6 public class Solution {
7     public static void main(String[] args) {
8
9         Person person = new Person();
10        System.out.println("Age: " + person.age);
11        person.adjustAge(person.age);
12        System.out.println("Adjusted age: " + person.age);
13    }
14
15    public static class Person {
16        public int age = 20;
17
18        public void adjustAge(int age) {
19            age = age + 20;
20            System.out.println("The age in adjustAge() is " + age);
21        }
22    }
23
24 }
25 }
```

¡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

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

¡Siempre
hacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 2 / Nivel 4 – Lección 2



Practice with variable visibility

Java Syntax
Level 4, Lesson 2

AVAILABLE

"Hi, Amigo."

"Hi, Diego."

"I brought you a few tasks about the visibility of variables."



Task Java Syntax, level 4, lesson 2

AVAILABLE



What's the cat's name?

The first thing a person/robot/variable receives after it is born/spawned/created is a name. Our brains aren't very comfortable with the idea of a world without names and titles. Let's come to their rescue: let's create a special automated naming method that sets the value of the private String variable name, i.e. names the object.

Open

¡Siempre
busca lo alto!



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 * What's the cat's name?
4 */
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

Cat register

Conditions Class tree

Write code in the `addNewCat` method to increase the number of cats by **1** each time it is called.
The variable `catCount` corresponds to the number of cats.

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: `addNewCat` and `main`.
- The Cat class's `addNewCat` method should increase the number of cats by 1.

Cat.java

```
1 package com.codegym.task.task04.task0404;
2
3 /*
4 * Cat register
5 */
6
7 public class Cat {
8     private static int catCount = 0;
9
10    public static void addNewCat() {
11        //write your code here
12    }
13
14    public static void main(String[] args) {
15    }
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_E3

Setting the number of cats

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 public class Cat {
8     private static int catCount = 0;
9
10    public static void setCatCount(int catCount) {
11        //write your code here
12    }
13
14    public static void main(String[] args) {
15    }
16
17 }
18
19 }
```

iSiempre
hacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 2 / Nivel 4 – Lección 2

★ Excercise: N4_L2_E4

Name register

Conditions Class tree

Verify Help Discuss New/Reset Run Code analysis

Finish writing the code of the `setName` method so that it sets the value of `private String fullName` to the value of the local `String` variable `fullName`.

Requirements:

- The Cat class must contain only one `fullName` variable.
- The variable `fullName` must be a `String` and have a `private` access modifier.
- The `Cat` class must have only two methods: `setName` and `main`.

Cat.java

```
1 package com.codegym.task.task04.task0406;
2
3 /*
4  * Name register
5  */
6
7 public class Cat {
8     private String fullName;
9
10    public void setName(String firstName, String lastName) {
11        String fullName = firstName + " " + lastName;
12
13        //write your code here
14    }
15
16    public static void main(String[] args) {
17    }
18
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_E5

Cats in the Universe

Verify Help Discuss New/Reset Run Code analysis

Conditions Class tree

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 }
```

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.

¡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 `;`



empre
nada lo alto!



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;
```



empre
nada lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 4 / Nivel 4 – Lección 4

★ Excercise: N4_L4_E1

Good or bad?

Conditions Class tree

Verify Help Discuss New/Reset Run Code analysis

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 public class Solution {
8     public static void main(String[] args) {
9         compare(3);
10        compare(6);
11        compare(5);
12    }
13
14    public static void compare(int a) {
15        //write your code here
16    }
17
18 }
```

iSiempre
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

Conditions Class tree

Verify Help Discuss New/Reset Run Code analysis

Solution.java

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

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()`.

¡Siempre
hacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 4 / Nivel 4 – Lección 4

★ Excercise: N4_L4_E3

Come on, lucky seven!

Conditions Class tree

Verify Help Discuss New/Reset Run Code analysis

Write the `checkInterval` method. The method should check whether an integer is in the range from **50** to **100** and display the result on the screen as follows: "*The number a is not in the interval.*" or "*The number a is in the interval.*", where **a** is the method argument.

Example for **112**:

The number 112 is not in the interval.

Example for **60**:

The number 60 is in the interval.

Solution.java

```
1 package com.codegym.task.task04.task0410;
2
3 /*
4 Come on, Lucky seven!
5
6 */
7
8 public class Solution {
9     public static void main(String[] args) {
10         checkInterval(60);
11         checkInterval(112);
12         checkInterval(10);
13     }
14
15     public static void checkInterval(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_E4

Seasons on Terra

Verify Help Discuss New/Reset Run Code analysis

Conditions Class tree

Solution.java

Write the `checkSeason` method. Based on a number representing a month, the method should determine and display the season (winter, spring, summer, autumn).

Example for number 2:

winter

Example for number 5:

spring

Hint: the numbers 12, 1, and 2 are winter months; 3, 4, and 5 are spring, etc.

```
1 package com.codegym.task.task04.task0411;
2
3 /*
4 Seasons on Terra
5
6 */
7
8 public class Solution {
9     public static void main(String[] args) {
10         checkSeason(12);
11         checkSeason(4);
12         checkSeason(7);
13         checkSeason(10);
14     }
15
16     public static void checkSeason(int month) {
17         //write your code here
18     }
19 }
```

¡Siempre
hacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 4 / Nivel 4 – Lección 4

★ Excercise: N4_L4_E5

Positive and negative numbers

Conditions Class tree

Use the keyboard **to enter** a number.
If the number is **positive**, **then double it**.
If the number is **negative**, **add one**.
If the entered **number is zero**, display **zero**.
Display the result on the screen.

Requirements:

- The program should read a number from the keyboard.

Solution.java

```
1 package com.codegym.task.task04.task0412;
2
3 /*
4 Positive and negative numbers
5 */
6
7 import java.io.*;
8
9 public class Solution {
10     public static void main(String[] args) throws Exception {
11         //write your code here
12     }
13 }
14
15 }
16 }
```

Verify Help Discuss New/Reset Run Code analysis





Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 4 / Nivel 4 – Lección 4

★ Excercise: N4_L4_E6

Day of the week

Conditions Class tree

Verify Help Discuss New/Reset Run Code analysis

Use the keyboard to enter a **number representing a day of the week**. Then, depending on the entered number, display the name of the day of the week: **"Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday", "Sunday"**, if you enter a number **greater than 7 or less than 1**, display "**No such day of the week**".

Example for number 5:

Friday

Example for number 10:

No such day of the week

Solution.java

```
1 package com.codegym.task.task04.task0413;
2
3 /*
4 Day of the week
5 */
6
7 import java.io.*;
8
9
10 public class Solution {
11     public static void main(String[] args) throws Exception {
12         //write your code here
13     }
14 }
```

iSiempre
hacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 4 / Nivel 4 – Lección 4

★ Excercise: N4_L4_E7

Number of days in the year

Conditions Class tree

Verify Help Discuss New/Reset Run Code analysis

Use the keyboard to enter a year, and then determine the number of days in the year. Display the result as follows:

Number of days in the year: x
, where
x is 366 for a leap year, and
x is 365 for an ordinary year.

Hint:
A leap year is 366 days, but an ordinary year is 365 days.
1) if the year is evenly divisible by 400, then it is a leap year;
2) for other years, if the year is evenly divisible by 100, then it is a regular year;
3) for other years, if the year is evenly divisible by 4, then it is a leap year;
4) all remaining years are not leap years.
Thus, the years 1700, 1800, and 1900 are not leap years, since they are multiples of 100 but not 400.
The years 1600 and 2000 are leap years, since they are multiples of 100 and multiples of 400.
The years 2100, 2200 and 2300 are not leap years.

Solution.java

```
1 package com.codegym.task.task04.task0414;
2
3 /*
4 Number of days in the year
5 */
6
7 import java.io.*;
8
9 public class Solution {
10     public static void main(String[] args) throws Exception {
11         //write your code here
12     }
13 }
```

Input data

2002

¡Siempre
hacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 4 / Nivel 4 – Lección 4

★ Excercise: N4_L4_E8

Rule of the triangle

Verify Help Discuss New/Reset Run Code analysis

Conditions Class tree

Use the keyboard to enter three numbers **a**, **b**, and **c** (the lengths of the sides of the proposed triangle). Determine whether a triangle with these sides can exist. Display the result as follows:
"The triangle is possible." - if a triangle with these sides **could exist**.
"The triangle is not possible." - if a triangle with these sides **cannot exist**.

Hint:
A triangle can exist only if the sum of two of its sides is greater than the third side.
You need to compare each side with the sum of the other two.
If even one side is larger or equal to the sum of the other two sides, then no such triangle exists.

Solution.java

```
1 package com.codegym.task.task04.task0415;
2
3 /*
4 Rule of the triangle
5 */
6
7 import java.io.*;
8
9 public class Solution {
10     public static void main(String[] args) throws Exception {
11         //write your code here
12     }
13 }
```

¡Siempre
hacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 4 / Nivel 4 – Lección 4



Excercise: N4_L4_E9

Crossing the road blindly

Conditions **Class tree**

The pedestrian traffic light is programmed as follows:
at the beginning of **each hour**, the **green** signal is on **for three minutes**, then the signal is **yellow** for one minute, and then it is **red** for one minute. Then the light is **green** again for three minutes, etc.
Use the keyboard to enter a real number **t** that represents the number of minutes that have elapsed since the beginning of the hour. Determine what color the traffic light is at the specified time.
Display the result as follows:
"green" if the light is green,
"yellow" if the light is yellow, and
"red" if the light is red.

Example for 2.5:
green

Example for 3:
yellow

Solution.java

```
1 package com.codegym.task.task04.task0416;
2
3 /*
4 Crossing the road blindly
5 */
6
7 import java.io.*;
8
9 public class Solution {
10     public static void main(String[] args) throws Exception {
11         //write your code here
12     }
13 }
```

Input data

2.5



Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 4 / Nivel 4 – Lección 4

★Excercise: N4_L4_E10

Do we have a pair?

Conditions Class tree

Verify Help Discuss New/Reset Run Code analysis

Solution.java

```
1 package com.codegym.task.task04.task0417;
2
3 /*
4  Do we have a pair?
5  */
6
7 import java.io.*;
8
9 public class Solution {
10    public static void main(String[] args) throws Exception {
11        //write your code here
12    }
13 }
```

Use the keyboard to enter **three integers**. Determine whether there is at least **one pair of equal numbers among them**. If such a pair exists, display **the numbers separated by a space**. If all three numbers are equal, then display **all three**.

Here are some examples:

a) if you enter the numbers
1
2
2
then we display
2 2

b) if you enter the numbers
2
2
2
then we display
2 2 2

Input data

4
6
6



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");
```

¡Siempre
hacia lo alto!



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."

Siempre
busca lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 6 / Nivel 4 – Lección 6

★ Excercise: N4_L6_E1

Minimum of two numbers

Verify Help Discuss New/Reset Run Code analysis

Conditions Class tree

Solution.java

```
1 package com.codegym.task.task04.task0418;
2
3 /*
4 Minimum of two numbers
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 }
```

Use the keyboard to enter **two integers**, and display **the minimum**. If the two numbers are **equal**, display **either** of them.

Requirements:

- The program should read the numbers from the keyboard.
- The program must display a number on the screen.

iSiempre
hacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 6 / Nivel 4 – Lección 6

★ Excercise: N4_L6_E2

Maximum of four numbers

Verify Help Discuss New/Reset Run Code analysis

Conditions Class tree

Solution.java

```
1 package com.codegym.task.task04.task0419;
2
3 /*
4 Maximum of four numbers
5 */
6
7 import java.io.*;
8
9
10 public class Solution {
11     public static void main(String[] args) throws Exception {
12         //write your code here
13     }
14 }
15
```

Use the keyboard to enter **four numbers**, and display **the maximum** of them. If the maximum occurs **more than once**, just display it **once**.

Requirements:

- The program should read the numbers from the keyboard.
- The program must display a number on the screen.

Siempre hacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 6 / Nivel 4 – Lección 6

★ Excercise: N4_L6_E3

Sorting three numbers

Verify Help Discuss New/Reset Run Code analysis

Conditions Class tree

Use the keyboard to enter **three numbers**, and display them in **descending order**.
The displayed numbers must be **separated by spaces**.

Requirements:

- The program should read the numbers from the keyboard.
- The program should display numbers on the screen.

Solution.java

```
1 package com.codegym.task.task04.task0420;
2
3 /*
4 Sorting three numbers
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
```

Siempre
hacia lo alto!



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

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
```

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.

Siempre
... hacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 6 / Nivel 4 – Lección 6

★ Excercise: N4_L6_E5

18+

Verify Help Discuss New/Reset Run Code analysis

Conditions Class tree Solution.java

Use the keyboard to enter **the name** and **age**. If the age is less than **18**, display "**Grow up a little more**".

Requirements:

- The program should read data from the keyboard twice.
- The program should use System.out.println() or System.out.print().

```
1 package com.codegym.task.task04.task0422;
2
3 /*
4 18+
5 */
6
7 import java.io.*;
8
9
10 public class Solution {
11     public static void main(String[] args) throws Exception {
12         //write your code here
13     }
14 }
15
```

Siempre
hacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 6 / Nivel 4 – Lección 6

★ Excercise: N4_L6_E6

Bouncer policy

Conditions Class tree

Use the keyboard to enter **the name** and **age**. If the age is **more than 20**, display "**18 is old enough**".

Requirements:

- The program should read lines from the keyboard.
- The program should use System.out.println() or System.out.print().

Verify Help Discuss New/Reset Run Code analysis

Solution.java

```
1 package com.codegym.task.task04.task0423;
2
3 /*
4  * Bouncer policy
5  */
6
7 import java.io.*;
8
9
10 public class Solution {
11     public static void main(String[] args) throws Exception {
12         //write your code here
13     }
14 }
15
```

siempre
hacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 6 / Nivel 4 – Lección 6

★ Excercise: N4_L6_E7

Three numbers

Conditions Class tree

Verify Help Discuss New/Reset Run Code analysis

Use the keyboard to enter **three integers**. One of the numbers is **unique**. The other two are **identical**. Display the **ordinal number** of the number that is **different** from the others.

Example for 4 6 6:

1

Example for 6 6 3:

3

Solution.java

```
1 package com.codegym.task.task04.task0424;
2
3 /*
4 Three numbers
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
```

¡Siempre hacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 6 / Nivel 4 – Lección 6



Excercise: N4_L6_E8

Target locked!

Conditions Class tree

Verify Help Discuss New/Reset Run Code analysis

Use the keyboard to enter **two integers** representing the coordinates of a point not on the coordinate axes. Display **the number of the quadrant** that contains the given point.

Hint:
Point (a, b) belongs to a quadrant if the following conditions are true:
for the first quadrant: $a > 0$ and $b > 0$;
for the second quadrant: $a < 0$ and $b > 0$;
for the third quadrant: $a < 0$ and $b < 0$;
for the fourth quadrant: $a > 0$ and $b < 0$.

Example for 4 6:

1

Example for -6 -6:

3

Solution.java

```
1 package com.codegym.task.task04.task0425;
2
3 /*
4 Target Locked!
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 }
```

Input data

3
-4



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	<pre>1 boolean m;</pre>	These two expressions are equivalent. The default value of a boolean variable is false .
2	<pre>1 boolean m = false;</pre>	
3	<pre>1 if (a > b) 2 System.out.println(a);</pre>	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 .
	<pre>1 boolean m = (a > b);</pre>	

¡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

Solution.java

```
1 package com.codegym.task.task04.task0426;
2 /*
3 * Labels and numbers
4 */
5
6 import java.io.*;
7
8 public class Solution {
9     public static void main(String[] args) throws Exception {
10        //write your code here
11    }
12 }
13
14
15
```

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

¡Siempre
hacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 7 / Nivel 4 – Lección 7



Excercise: N4_L7_E2

Describing numbers

Conditions Class tree

Solution.java

Enter an integer from the keyboard in the range **1 - 999**. Display a string description as follows:

"**even single-digit number**" - if the number is **even** and has **one digit**,
"**odd single-digit number**" - if the number is **odd** and has **one digit**,
"**even two-digit number**" - if the number is **even** and has **two digits**,
"**odd two-digit number**" - if the number is **odd** and has **two digits**,
"**even three-digit number**" - if the number is **even** and has **three digits**,
"**odd three-digit number**" - if the number is **odd** and has **three digits**.
If the entered number does not fall in the range **1 - 999**, don't display anything.

Example for **100**:

even three-digit number

Example for **51**:

odd two-digit number

Verify Help Discuss New/Reset Run Code analysis

```
1 package com.codegym.task.task04.task0427;
2
3 /*
4 Describing numbers
5 */
6
7 import java.io.*;
8
9
10 public class Solution {
11     public static void main(String[] args) throws Exception {
12         //write your code here
13     }
14 }
```

Input data

42



Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 7 / Nivel 4 – Lección 7



Excercise: N4_L7_E3

Positive number

Conditions Class tree

Verify Help Discuss New/Reset Run Code analysis

Use the keyboard to enter **three integers**. Display **the number of positive numbers** in the original set.

Here are some examples:

a) if you enter the numbers
-4
6
5
then we display
2

b) if you enter the numbers
-6
-6
-3
then we display
0

c) if you enter the numbers
0
1
2
then we display
2

Solution.java

```
1 package com.codegym.task.task04.task0428;
2
3 /*
4  * Positive number
5  */
6
7 import java.io.*;
8
9 public class Solution {
10     public static void main(String[] args) throws Exception {
11         //write your code here
12     }
13 }
14
15
16
```

Input data

```
-5
-6
6
```



Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 7 / Nivel 4 – Lección 7



Excercise: N4_L7_E4

Positive and negative numbers

Verify Help Discuss New/Reset Run Code analysis

Solution.java

```
1 package com.codegym.task.task04.task0429;
2
3 /*
4  * Positive and negative numbers
5  */
6
7 import java.io.*;
8
9 public class Solution {
10    public static void main(String[] args) throws Exception {
11        //write your code here
12    }
13
14 }
15
16
```

Use the keyboard to enter **three integers**. Display the number of positive numbers and **the number of negative numbers** in the original set, in the following form:
"Number of negative numbers: a" and
"Number of positive numbers: b", where **a** and **b** are the unknowns.

Here are some examples:

a) if you enter the numbers
2
5
6
then we display

Number of negative numbers: 0
Number of positive numbers: 3

b) if you enter the numbers
-2
-5
6
then we display

Number of negative numbers: 2
Number of positive numbers: 1

Input data

```
-5
5
4
```



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++;
6 }
```

¡Siempre
hacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 10 / Nivel 4 – Lección 10



Tasks about loops

Java Syntax
Level 4, Lesson 10

AVAILABLE

"Hi, Amigo!"

"I heard that you've learned about loops. I'll give you a couple of tasks."



Task Java Syntax, level 4, lesson 10

1 to 10

EASY

AVAILABLE



Human kids quickly learn to count to ten with their fingers. The inhabitants of Planet Skrewy with their 2,048 tentacles are less lucky. Let's count up to ten using Java and a while loop. We won't just count. We'll also display our count, and each value must be displayed on a new line.

Open



alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 10 / Nivel 4 – Lección 10

★ Excercise: N4_L10_E1

1 to 10

Verify Help Discuss New/Reset Run Code analysis

Conditions Class tree

Display the numbers from **1** to **10** using a **while** loop. Each value should be on a new line.

Requirements:

- The program should not read the numbers from the keyboard.
- The program should display numbers on the screen.

Solution.java

```
1 package com.codegym.task.task04.task0430;
2
3 /*
4 1 to 10
5 */
6
7 import java.io.*;
8
9
10 public class Solution {
11     public static void main(String[] args) throws Exception {
12         //write your code here
13     }
14 }
15 }
```

¡Siempre
hacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 10 / Nivel 4 – Lección 10



Excercise: N4_L10_E2

From 10 to 1

Verify Help Discuss New/Reset Run Code analysis

Conditions Class tree

Solution.java

```
1 package com.codegym.task.task04.task0431;
2
3 /*
4 From 10 to 1
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
16 }
```

Display the numbers from **10** to **1** using a **while** loop. Each value should be on a new line.

Requirements:

- The program should not read the numbers from the keyboard.
- The program should display numbers on the screen.
- Each value should be displayed on a new line.

iSiempre
hacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 10 / Nivel 4 – Lección 10



Excercise: N4_L10_E3

You can't have too much of a good thing

Verify Help Discuss New/Reset Run Code analysis

Conditions Class tree Solution.java

Use the keyboard to enter a string and a number **N** greater than 0. Use a **while** loop to display the string **N** times. Each value should be on a new line.

Example input:

```
abc  
2
```

Example output:

```
abc  
abc
```

```
1 package com.codegym.task.task04.task0432;  
2  
3  
4  
5 /* You can't have too much of a good thing */  
6  
7  
8  
9  
10 import java.io.*;  
11  
12 public class Solution {  
13     public static void main(String[] args) throws Exception {  
14         //write your code here  
15     }  
16 }  
17  
18 }
```

¡Siempre
hacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 10 / Nivel 4 – Lección 10

★ Excercise: N4_L10_E4

Seeing dollars in your future

Verify Help Discuss New/Reset Run Code analysis

Conditions Class tree

Solution.java

Use a `while` loop to display a **10x10** square of **dollar signs**.
Don't separate the symbols in each line.

Example output:

```
$$$$$$$$$$  
$$$$$$$$$$  
$$$$$$$$$$  
$$$$$$$$$$  
$$$$$$$$$$  
$$$$$$$$$$  
$$$$$$$$$$  
$$$$$$$$$$  
$$$$$$$$$$  
$$$$$$$$$$
```

```
1 package com.codegym.task.task04.task0433;  
2  
3  
4 *  
5 Seeing dollars in your future  
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 }  
16  
17 }
```

¡Siempre
hacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 10 / Nivel 4 – Lección 10

★ Excercise: N4_L10_E5

Multiplication table

Conditions Class tree

Verify Help Discuss New/Reset Run Code analysis

Use a `while` loop to display a **10x10** multiplication table.
Separate the numbers using a space.

Example output:

```
1 2 3 4 5 6 7 8 9 10
2 4 6 8 10 12 14 16 18 20
3 6 9 12 15 18 21 24 27 30
4 8 12 16 20 24 28 32 36 40
5 10 15 20 25 30 35 40 45 50
6 12 18 24 30 36 42 48 54 60
7 14 21 28 35 42 49 56 63 70
8 16 24 32 40 48 56 64 72 80
9 18 27 36 45 54 63 72 81 90
10 20 30 40 50 60 70 80 90 100
```

Solution.java

```
1 package com.codegym.task.task04.task0434;
2
3
4 /*
5 * Multiplication table
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 }
16
17
```

¡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 }
```

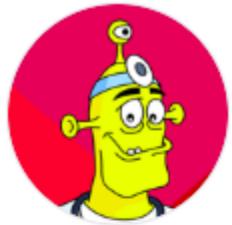
for

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



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..."



Task Java Syntax, level 4, lesson 13

AVAILABLE

EASY



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.

x2

Open





Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 13 / Nivel 4 – Lección 13

★ Excercise: N4_L13_E1

Even numbers

Verify Help Discuss New/Reset Run Code analysis

Conditions Class tree

Use a **for** loop to display even numbers from **1** to **100** inclusive. Display each value on a new line.

Requirements:

- The program should not read text from the keyboard.
- The program should display text on the screen.

Solution.java

```
1 package com.codegym.task.task04.task0435;
2
3 /*
4 Even numbers
5
6 */
7
8
9 public class Solution {
10     public static void main(String[] args) throws Exception {
11         //write your code here
12     }
13 }
14
15 }
```

¡Siempre
hacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 13 / Nivel 4 – Lección 13



Excercise: N4_L13_E2

Drawing a rectangle

Conditions Class tree

Verify Help Discuss New/Reset Run Code analysis

Use the keyboard to enter two numbers **m** and **n**. Using a **for** loop, display an **n** x **m** rectangle made of eights.

Here's an example: **m=2, n=4**

```
8888  
8888
```

Requirements:

- The program should read two numbers from the keyboard.

Solution.java

```
1 package com.codegym.task.task04.task0436;  
2  
3  
4 /*  
5 Drawing a rectangle  
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 }  
16 }  
17 }
```

¡Siempre
hacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 13 / Nivel 4 – Lección 13

★ Excercise: N4_L13_E3

Triangle of eights

Conditions Class tree

Using a **for** loop, display a right triangle of eights, with a base of **10** and a height of **10**.

Example of screen output:

```
8
88
888
8888
88888
888888
8888888
88888888
888888888
```

Solution.java

```
1 package com.codegym.task.task04.task0437;
2
3
4 /*
5 Triangle of eights
6
7 */
8
9 public class Solution {
10     public static void main(String[] args) throws Exception {
11         //write your code here
12     }
13 }
```

¡Siempre
hacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 13 / Nivel 4 – Lección 13

★ Excercise: N4_L13_E4

Drawing lines

Conditions **Class tree**

Using a **for** loop to display:
- a horizontal line of **10** eights
- a vertical line of **10** eights (do not count any of the eights in the horizontal line as part of this vertical line).

Requirements:

- The program should not read numbers from the keyboard.

Solution.java

```
1 package com.codegym.task.task04.task0438;
2
3 /*
4 Drawing Lines
5
6 */
7
8 public class Solution {
9     public static void main(String[] args) throws Exception {
10         //write your code here
11     }
12 }
13
14 }
```

¡Siempre
hacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 13 / Nivel 4 – Lección 13

★ Excercise: N4_L13_E5

Chain letter

Verify Help Discuss New/Reset Run Code analysis

Conditions Class tree

Solution.java

Enter a name from the keyboard and use a **for** loop to display the following 10 times: <name> loves me.

Example output for the name "Scarlett":

```
Scarlett loves me.  
Scarlett loves me.
```

```
1 package com.codegym.task.task04.task0439;  
2  
3 /*  
4 Chain Letter  
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  
16 }
```

¡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?"



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.

x2

Open



re
alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 16 / Nivel 4 – Lección 16



Tasks with and without asterisks

Java Syntax
Level 4, Lesson 16

AVAILABLE

"Hello, soldier!"

"Hello, Captain Squirrels, sir!"

"I have great news for you. Here are some exercises to reinforce your skills. Work on them every day, and your competence will grow exponentially. They were specially designed for IntelliJ IDEA."



Task Java Syntax, level 4, lesson 16

EASY

AVAILABLE



Decent pay

Have you received an indecent financial offer? Unsure how to react to the obscenity? Do you call the morality police or simply the police? Immediately file a lawsuit? Calm down: you're learning how to program, so don't get used to indecent offers. Instead, use a loop to display the following phrase 100 times: "I will never work for peanuts".



Open



Siempre
hacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 16 / Nivel 4 – Lección 16

★ Excercise: N4_L16_E1

Decent pay

Verify Help Discuss New/Reset Run Code analysis

Conditions Class tree

Using a loop to display the following phrase **one hundred times**:
"I will never work for peanuts. Amigo"
Display each value on a new line.

Requirements:

- The program should not read text from the keyboard.

Solution.java

```
1 package com.codegym.task.task04.task0440;
2
3 /*
4 Decent pay
5
6 */
7
8 public class Solution {
9     public static void main(String[] args) {
10         //write your code here
11     }
12 }
```

¡Siempre
hacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 16 / Nivel 4 – Lección 16

★ Excercise: N4_L16_E2

Somehow average

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

Conditions **Class tree**

Use the keyboard to enter **three numbers**, and then display **the middle number**.
In other words, **not the largest** and **not the smallest**.
If all the numbers are **equal**, display **any one** of them.

Requirements:

- The program should read the numbers from the keyboard.

Solution.java

```
1 package com.codegym.task.task04.task0441;
2
3
4 /*
5 * Somehow average
6 */
7 import java.io.*;
8
9
10 public class Solution {
11     public static void main(String[] args) throws Exception {
12         //write your code here
13     }
14 }
15
```

¡Siempre hacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 16 / Nivel 4 – Lección 16

★ Excercise: N4_L16_E3

Adding

Verify Help Discuss New/Reset Run Code analysis

Conditions Class tree

Solution.java

Use the keyboard to enter **numbers**.
If the user enters **-1**, display the sum of all entered numbers and end the program.
-1 should be included in the sum.

Hint: one of the solutions to this problem uses the following construct:

```
while (true) {
    int number = read the number
    if (check whether the number
        break;
}
```

```
1 package com.codegym.task.task04.task0442;
2
3
4 /*
5 Adding
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 }
16
```

¡Siempre
hacia lo alto!



Gamified platform work / Trabajo en plataforma gamificada.

Level 4 - Lesson 16 / Nivel 4 – Lección 16

★ Excercise: N4_L16_E4

A name is a name

Verify Help Discuss New/Reset Run Code analysis

Conditions Class tree

Use the keyboard to enter a **name**.
Use the keyboard to enter a **birth date** (three numbers): **yyyy**, **mm**, **dd**.

Display the following:
"My name is *name*.
I was born on mm/dd/yyyy"

Example output:

My name is Kevin.
I was born on 2/15/1988

Solution.java

```
1 package com.codegym.task.task04.task0443;  
2  
3  
4 /*  
5 A name is a name  
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 }  
16
```

¡Siempre
nacia lo alto!



Feedback

LET'S
GO!



¡Siempre
hacia lo alto!



Survey



¡Siempre
hacia lo alto!



Clarification of doubts



¡Siempre
hacia lo alto!



To practice!

LET'S
GO!



¡Siempre
hacia lo alto!





UNIVERSIDAD SANTO TOMÁS
PRIMER CLAUSTRO UNIVERSITARIO DE COLOMBIA
SECCIONAL TUNJA

VIGILADA MINEDUCACIÓN - SNIES 1732

¡Siempre
hacia lo alto!

USTATUNJA.EDU.CO

