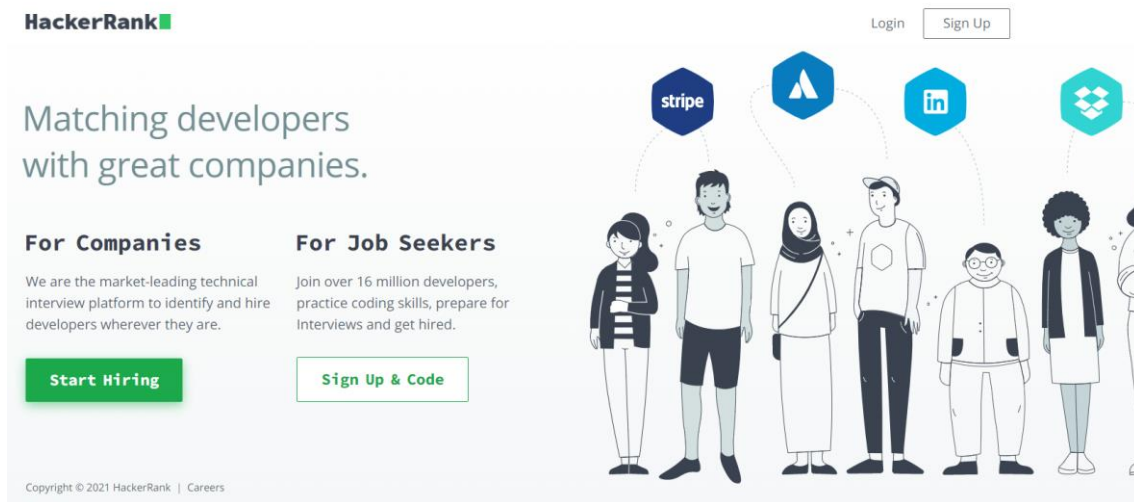


# Descripción de la plataforma

## Introducción

Hackerrank es una plataforma online pensada para realizar competiciones de programación.

## Funcionamiento interno



- la primera vez que entremos a HackerRank, debemos de crear una cuenta con solo ingresar 3 datos (*nombre completo, correo electrónico, contraseña*) y dar clic en **Sign up**.
- Una vez registrado, nos mostrará un formulario (tal como se ve en la imagen debajo) que deberemos completar y marcar ambos checks (requisito de HackerRank) para ingresar al test.

ivan@laboratoria.la ▼

Complete this form to start the challenge.

Nombre

Apellidos

Codigo Laboratoria

Squad

Sede

☐ Share my [HackerRank profile](#) with Laboratoria <sup>?</sup>

☐ I will not consult/copy code from any source including a website, book, or friend/colleague to complete these tests, though may reference language documentation or use an IDE that has code completion features.

Start Challenge

HackerRank

PREPARE

NEW

CERTIFY

COMPETE

Prepare

Prepare

Y dentro de las certificaciones buscamos la correspondiente, por ejemplo Java Basic:

## Java (Basic) ⓘ

Get Certified



Y procedemos a cumplimentar nuestros datos:

## Verify your Java Skills. Accelerate your Job Search.

Take the HackerRank Skills Certification Test and showcase your knowledge as a HackerRank verified developer.

**Take the HackerRank Skills Test**

Expected year of Graduation \*  
 2022

Expected month of Graduation \*  
 Month

Degree  
 ex: B.E

Program  
 ex: Computer Science and Engineering

LinkedIn / Resume  
 Add your LinkedIn url

or [Upload File](#)  
 (Accepted file formats: PDF, DOCX, TXT)

WORK AUTHORIZATION

Are you legally authorized to work in the United States? \*

☐ Yes ☐ No

Please fill all the required fields

**Proceed**

- Una vez dentro del test, veremos el tiempo restante para culminarlo así como las preguntas del test:

PA001 - Variables, Tipos de Datos y Operadores				
		03 : 59 to test end	0/10 Attempted	ivan@laboratoria.la
	QUESTION NAME	TYPE	STATUS	
☆	Q1 Suma de dos números	Coding	not answered	Solve Question
☆	Q2 Promedio de Notas	Coding	not answered	Solve Question
☆	Q3 Mayor número de N cifras	Coding	not answered	Solve Question
☆	Q4 Es pulpina?	Coding	not answered	Solve Question
☆	Q5 Suma de Números Consecutivos	Coding	not answered	Solve Question
☆	Q6 Área de un Círculo	Coding	not answered	Solve Question
☆	Q7 Conversor de monedas	Coding	not answered	Solve Question
☆	Q8 Repartición de Caramelos	Coding	not answered	Solve Question
☆	Q9 Will you?	Coding	not answered	Solve Question
☆	Q10 Ángulo Horario	Coding	not answered	Solve Question

## Test de HackerRank

- Para resolver un test, solo es necesario dar clic al nombre del ejercicio y nos llevará a una pantalla con el enunciado de la pregunta y un editor de código embebido:

<>

PA001 - Variables, Tipos de Datos y Operadores

03:57  
to test end

0/10 Attempted

ivan@laboratoria.la

☰

☆ Suma de dos números

1

2

3

4

5

6

7

8

9

10

Escriba un algoritmo que calcule la suma de dos números (num1 y num2).

Ejemplo:

**Input**  
num1 = 10  
num2 = 20

**Output**  
sumaNumeros(num1, num2) = 30

**YOUR ANSWER**

We recommend you take a quick tour of our editor before you proceed. The timer will pause up to 90 seconds for the tour. [Start tour](#)

Draft saved 04:33 pm

Original code

JavaScript

⌵ ⚙

```

1 // No tocar esta sección
2 process.stdin.resume();
3 process.stdin.setEncoding('utf-8');
4
5 var __input_stdin = "";
6 var __input_stdin_array = "";
7 var __input_currentline = 0;

```

## Ejercicio de un test de HackerRank

- Dentro del código que viene por defecto en el editor, se encuentra una función, esta es la que se ejecutará al momento de enviar tu solución para comprobar que los parámetros (**entrada**) mediante la función (**algoritmo**) obtengan el resultado correcto (**salida**).

```

13 /*
14  * Implementar el algoritmo dentro de la función.
15  */
16 function sumaNumeros(num1, num2) {
17     var suma = num1 + num2;
18     return suma;
19 }

```

num1 y num2 son las entradas, en este caso, suma es la salida.

**Nota:** Debe de escribirse la palabra clave **return** para que la función retorne el resultado que se espera.

- Luego de implementar la solución pensada, aparecen 2 botones **Run Code** y **Submit code & Continue**. El primer botón, permite ejecutar la función reemplazando los parámetros por casos de prueba y comparar el resultado esperado con el resultado obtenido. El segundo botón, hace lo mismo que *Run Code* pero envía tu solución y cierra el ejercicio entendiéndose de que ya no volverás a editar tu solución.

☐ Test against custom input

Run Code

Submit code & Continue

(You can submit any number of times)

[Download sample test cases](#) The input/output files have Unix line endings. Do not use Notepad to edit them on windows.

Status: **No test cases passed.**

Testcase 1: **Wrong Answer**

Input [Download](#)

```
10
20
```

Your Output

```
1020
```

Expected Output [Download](#)

```
30
```

Run Code.

- Si te sale el mensaje en rojo **Wrong Answer**, quiere decir que la solución no cumplió los casos de prueba. La sección debajo de **Input** son los valores que reemplazarán a los parámetros de la función. **Output** es el resultado que devuelve la solución (lo que pusiste luego de return en el código). **Expected Output** es el resultado esperado. Para que tu solución esté correcta, el **Output** y **Expected Output** deben ser exactamente lo mismo.

Compiled successfully. All available test cases passed!

Test Case #1: ✓  
Test Case #2: ✓

Test Case #3: ✓  
Test Case #4: ✓

Test Case #5: ✓

Testcase 1: **Success**

Input [Download](#)

```
10
20
```

Your Output

```
30
```

Expected Output [Download](#)

```
30
```

Casos de prueba correctos

- Quando todos los casos de prueba pasan correctamente, debes enviar tu solución dando clic en **Submit code & Continue**. HackerRank te redireccionará al siguiente ejercicio automáticamente.
- Quando quieras terminar el test debes volver a la pantalla principal del listado de ejercicios y darle clic a **I am done with the test**.

✓ I am done with the test

Cada problema tiene asociada una puntuacion maxima que sera conseguida cuando el codigo que suba el usuario pase todos los casos de entrada a los que se le somete.

Supongamos que existen 10 casos de entrada en un determinado problema; si el codigo subido por el usuario supera 6 de los 10 casos de entrada, la puntuacion obtenida en ese problema sera de 60 puntos ( $6/10 * 100$  puntos).

Un problema puede ser intentado cuantas veces se quiera. Para esta competicion no habra penalizacion por intentos fallidos.

Dependiendo del lenguaje empleado existira un tiempo maximo de ejecucion del problema. Algunos casos de entrada se han escogido para forzar este tiempo maximo, por lo tanto tu codigo debe estar optimizado. Mas informacion sobre el tiempo maximo de ejecucion y librerias permitidas en: <https://www.hackerrank.com/environment>

Por favor, si vais a usar Java / C# / Clojure / Scala debeis llamar a la clase subida "Solution" (mas info. en el link de arriba).

Descripción de los problemas

Cada problema tendra una descripcion suficientemente detallada de la tarea a programar por el participante. Junto a la descripcion se proporcionan el formato de los datos de entrada que el programa del participante debe recibir. Los datos **SIEMPRE** seran leidos de la entrada estandar (STDIN) y los resultados deberan ser **SIEMPRE** mostrados por la salida estandar (STDOUT).

La manera en que deben leerse los datos de entrada y mostrarse los datos de salida depende del lenguaje usado, pero aqui hay algunos ejemplos:

C:

IN: scanf() gets() getchar()

OUT: printf() puts() putchar()

*C++:*

IN: cin

OUT: cout

*C#:*

IN: Console.ReadLine()

OUT: Console.WriteLine()

*Python2:*

IN: raw\_input()

OUT: print "..."

*Python3:*

IN: input()

OUT: print()

*Java:*

Ver [http://en.wikipedia.org/wiki/Standard\\_streams#1995:\\_Java](http://en.wikipedia.org/wiki/Standard_streams#1995:_Java)

Entrada

El formato de los datos de entrada sera proporcionado en la seccion *Input Format* (debajo de la descripcion). Le sigue inmediatamente las "Constraints" o restricciones para la entrada. Indican el tamaño maximo de los datos de entrada y el rango para los numeros

Salida

El formato de los datos de salida sera proporcionado en la seccion *Output Format* (debajo de *Input Format*)

Ejemplos de entrada y salida

Junto a los formatos de entrada y salida se proporcionara un ejemplo para que el participante entienda mejor el problema. El ejemplo **SIEMPRE** formara parte de los casos de entrada contra los que se evalua el problema.

## Problema de ejemplo

Para que entendais todos el funcionamiento de la plataforma vamos a suponer un caso sencillo.

El problema contendria el siguiente enunciado:

Se quiere instalar un sencillo Siri en la puerta del edificio de Teleco. Cada vez que una persona entre al edificio esta dira su nombre y el Siri debe darle los buenos dias. Asumimos que existe la tecnologia de Voz a Texto (Speech Recognition), de tal manera que tu programa tan solo recibira el nombre de la persona y debe saludarla correctamente.



(Abajo sigue el formato de entrada y salida)

### **Input Format**

Una sola línea (acabada en salto de línea) con el nombre de la persona a saludar.

Constraints

Longitud del nombre de la persona < 100 caracteres

### **Output Format**

La frase "Hola XXXX" donde XXXX es el nombre de la persona leído de la entrada estándar. Acabará siempre con un salto de línea.

### **Sample Input**

Perico el de los palotes

### **Sample Output**

Hola Perico el de los palotes

### **Explanation**

"Perico el de los palotes" es leído de la entrada estándar (por ejemplo `raw_input()` con python 2) y la frase "Hola Perico el de los palotes" es mostrada (por ejemplo con `print` ).

Ese programa en Python2 se acometería de la siguiente manera:

```
nombre = raw_input()
print "Hola " + nombre
```

Al final, si hemos conseguido superarlo, nos debe aparecer una certificación de este tipo.

## Java (Basic) Certificate

Google Translate



### Share this Certificate

<https://www.hackerrank.com/cert/> [Copy Link](#)

### Java (Basic)

It will cover basic topics in Java language such as classes, data structures, inheritance, exception handling, etc. You are expected to be proficient in either Java 7 or Java 8.

### Standout from the crowd

Take the HackerRank Skills Certification Test and make your profile stand out to employers and peers

[Learn More](#)