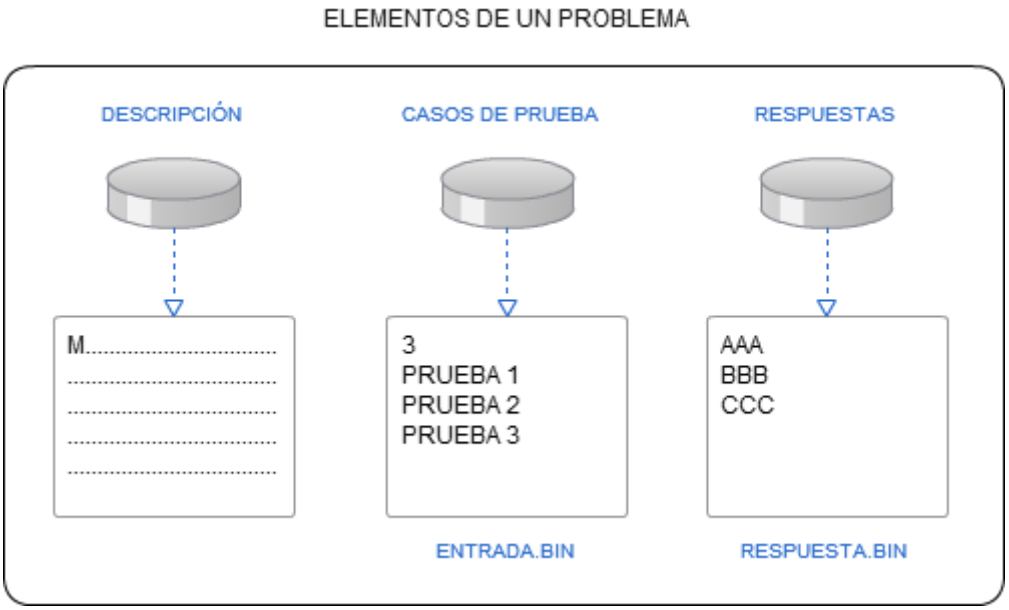
Contenido

Sobre el problema

¿Qué es un problema?

Un **PROBLEMA** busca cuestionar la capacidad de un competidor de encontrar soluciones correctas y optimas, el reto consiste en desafiar el **ANÁLISIS** y la **IMPLEMENTACIÓN** mediante un **LENGUAJE DE PROGRAMACIÓN** de una serie de pasos que den respuesta a una prueba específica, en otras palabras el competidor debe hacer uso de capacidad de **RAZONAMIENTO**, su **LÓGICA**, y sus conocimientos básicos sobre un **LENGUAJE DE PROGRAMACIÓN** cualquiera para satisfacer la necesidad de solución de un problema. Los elementos de un problema se describen en el siguiente gráfico.



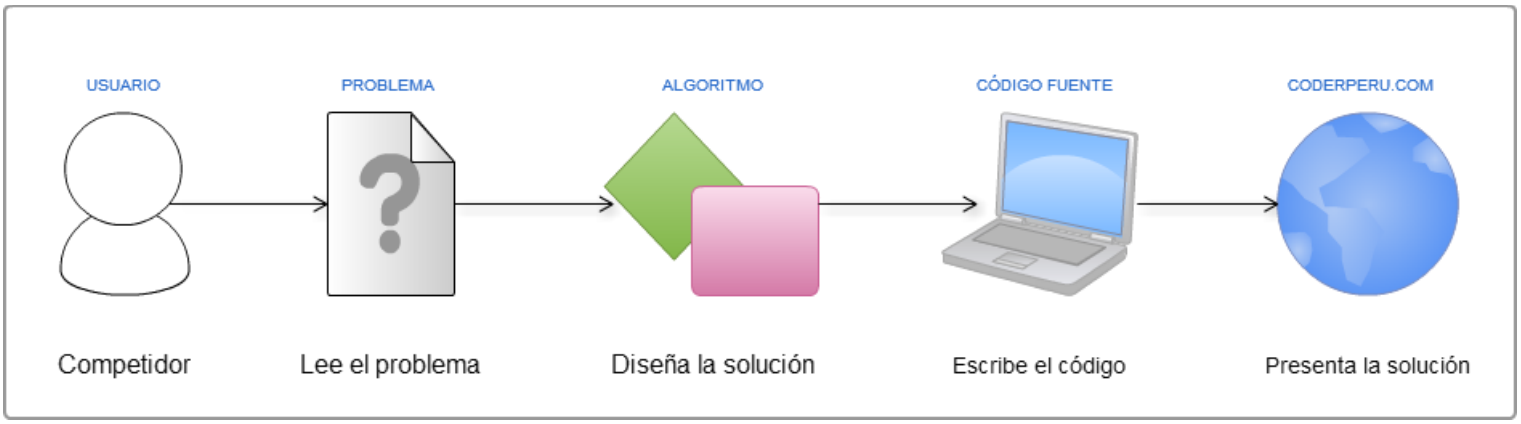
La **DESCRIPCIÓN** propiamente dicha es lo que se le pide al competidor que resuelva. Mediante el uso de los **CASOS DE PRUEBA** el **JUEZ** evaluará la efectividad de la solución propuesta. Cabe mencionar que la **SOLUCIÓN** es **CÓDIGO FUENTE** escrito en algún **LENGUAJE DE PROGRAMACIÓN**. El último elemento de un problema son las **RESPUESTAS**, este elemento contiene las respuestas correctas a todos los **CASOS DE PRUEBA**, servirá para compararlas con las respuesta que dé la **SOLUCIÓN** del competidor. Para definir, un ejemplo:

Considere el problema como una prueba de **SALTO ALTO**:

PRUEBA *→ el atleta debe dar un salto sobre una barra de metal colocada a 70 cm desde el nivel del suelo.*
SALTO *→ después de dar el primer salto, si el atleta supera los 70 cm entonces subiremos la altura a 80 cm.*
SALTOS *→ siguiendo el mismo ritmo, mientras el atleta supere la altura dada, aumentaremos esta en 10 cm cada vez hasta la altura que consideremos sea suficiente para aprobar la habilidad del atleta.*

De esta misma manera funcionan las cosas aquí, la **SOLUCIÓN** de un competidor se probará tantas veces sea necesaria hasta considerar que sea **CORRECTA**, estas pruebas son llamados **CASOS DE PRUEBA**, cada respuesta que dé la **SOLUCIÓN** será como un salto del atleta, si es superado se seguirá probando, si no es superado la **SOLUCIÓN** será rechazada.

¡ENTONCES! ¿CÓMO SE RESUELVE UN PROBLEMA? A continuación detallaremos los pasos que usted debe seguir para hacerlo y las herramientas que puede utilizar para competir. El flujo de actividades del competidor de un concurso se representa en el siguiente gráfico.



Leer el problema

Es muy importante saber **LEER** el problema, si un problema no es bien leído o mal comprendido, la solución será incorrecta y el tiempo invertido en este estará **PERDIDO**. Un problema es una descripción de un evento o suceso que supone una dificultad, esta dificultad es gradualmente aumentada en cada **PROBLEMA**. Esto quiere decir que si un concurso propone cinco problemas, el problema numero cinco será de mayor dificultad que el número cuatro, el número cuatro de mayor dificultad que el número tres, y así sucesivamente hacia atrás.

Es importante señalar que dentro de un problema se encuentran las **RESTRICCIONES**, estos son los pequeños detalles que moldearán nuestro código de solución, las **RESTRICCIONES** son los **LÍMITES** para los datos de los **CASOS DE PRUEBA** del problema. Por ejemplo:

Considere una variable de nombre **PESO** con la siguiente restricción:

1 <= PESO <= 100 - En este caso estamos diciendo que la variable **PESO** puede adquirir un valor aleatorio entre **1** y **100**, ósea en algún momento puede valor **10**, **50**, **1** o simplemente **100**.

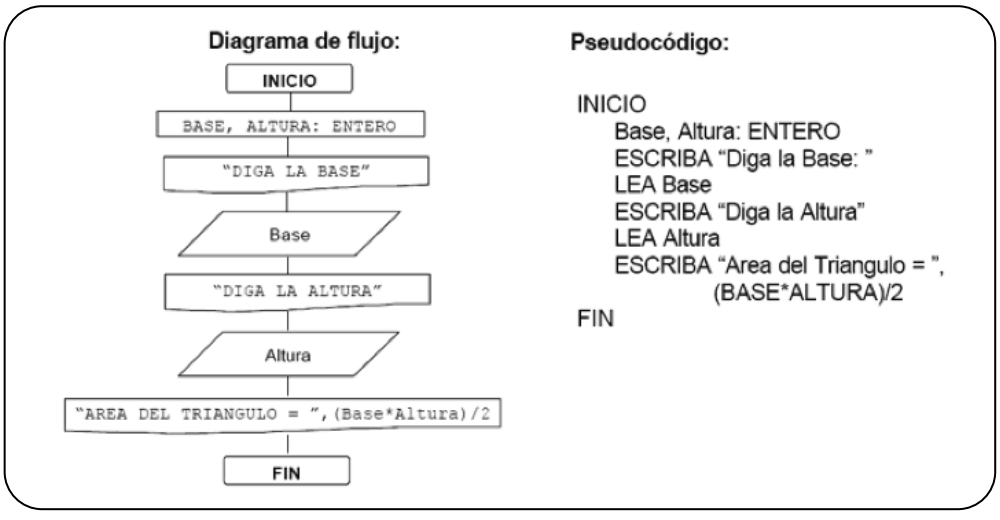
Estas restricciones son útiles para calcular hasta que valores pueden llegar algunas operaciones. Por ejemplo:

Considere que se nos pide imprimir el resultado de **PESO** x **PESO**. Entonces podemos deducir que la respuesta a esta operación puede ser a lo mucho **10000**, puesto que si en algún momento esta variable tomara el valor de **100**.

¡IMPORTANTE! Todos los problemas indican una **RESTRICCIÓN** para los **CASOS DE PRUEBA.**

Diseñar la solución

Tras la comprensión del problema y las restricciones de este, procedemos a elaborar nuestro **ALGORITMO DE SOLUCIÓN**, es conocida por muchos las técnicas que ayudan a este tipo de actividades, el competidor puede hacer uso de **DIAGRAMAS DE FLUJO**, **PSEUDOCÓDIGO** o simplemente **GRÁFICAS** y **EXPRESIONES**, esta última informal herramienta consiste en escribir símbolos, letras y números que solo podrá entender el autor. Es una buena costumbre siempre hacer uso de de lápiz y papel. Todas estas técnicas solo cumplen con un fin específico… **¡ENCONTRAR LA SOLUCIÓN CORRECTA!**



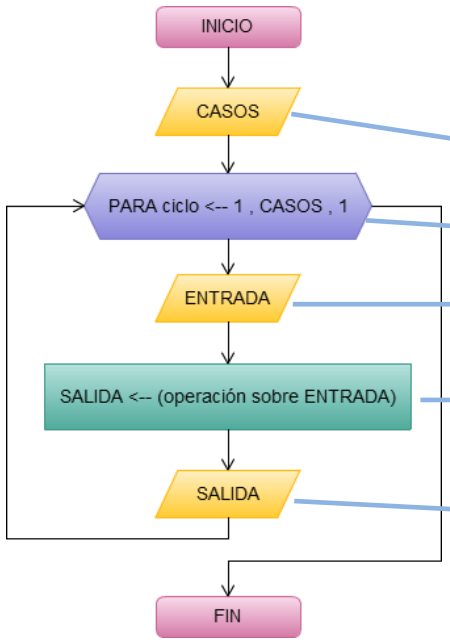
Escribir el código

En esta parte de la actividad, el competidor escribe el **CÓDIGO FUENTE** que finalmente presentará al **JUEZ** para ser evaluado, haciendo uso del algún **LENGUAJE DE PROGRAMACIÓN** y un **EDITOR DE TEXTO,** el competidor en base a sus conocimientos previos escribirá cada línea de código requerida por su programa. Es importante no olvidar las **RESTRICCIONES** del problema en la construcción del **CÓDIGO**, puesto que en problemas de mayor dificultad esto es **VITAL**.

JAVA, **C++**, **RUBY**, **PHYTON**, entre otros, son los **LENGUAJES DE PROGRAMACIÓN** más utilizados en estas actividades, en nuestro caso y para fines prácticos explicaremos nuestros ejemplos haciendo uso del lenguaje **JAVA**.

En la construcción del **CÓDIGO FUENTE** es de suma importancia considerar una **ÚNICA ESTRUCTURA**, la misma que servirá para la correcta ejecución de este por parte del **COMPILADOR**. A continuación el punto más importante de resolver un problema… **¡ESCRIBIR EL CÓDIGO CORRECTO!** Vea detenidamente la siguiente imagen.

DIAGRAMA DE FLUJO	CÓDIGO FUENTE (ESCRITO EN JAVA)




```
2
3   import java.util.Scanner;
4
5   public class Main {
6
7       public static void main(String[] args) {
8           Scanner entrada = new Scanner(System.in);
9           int casos = Integer.parseInt(entrada.nextLine());
10
11          for (int i = 0; i < casos; i++) {
12
13              /* En esta sección el competidor
14                 obtiene un caso de prueba a la vez */
15
16              /* Se generará la solución del problema */
17
18              /* Mediante el uso de consola
19                 se imprime la respuesta */
20
21              System.out.println("");
22          }
23      }
24
25  }
```

Okey! ¿Estás listo? Empecemos por describir el **DIAGRAMA DE FLUJO**.

Dado el inicio del flujo podemos ver que este solicita un valor para la variable **CASOS**, este dato representa el **NÚMERO DE CASOS DE PRUEBA** con los que se evaluará la solución, para la descripción del diagrama asumiremos que su valor es "**3**". Así mismo podemos incluir en nuestra definición las siguientes equivalencias:

Variable **"CASOS" = CASOS DE PRUEBA = NÚMERO DE REPETICIONES = LÍMITE DEL BUCLE = 3**

A continuación el flujo entra en un bucle "**PARA**" o **CICLO REPETITIVO**, siguiendo los siguientes parámetros. Podemos observar el primer carácter dado es el número "**1**", luego tras una coma el nombre de la variable "**CASOS**" y por último tras un segunda coma el valor "**1**" nuevamente. Entonces mucho cuidado al leer, que esta representación significa lo siguiente:

El primer "1" → *indica que el ciclo de repeticiones comenzará desde uno.*
La variable "CASOS" → *indica que el bucle se repetirá hasta alcanzar el valor de **CASOS**.*
El segundo "1" → *indica que tras cada repetición el bucle aumentará su valor en uno.*

Entonces en la práctica decimos. Como el valor de la variable **CASOS** es "**3**", nuestro bucle se repetirá "**3**" veces:

Primera vuelta → *el bucle inicia en "1" y al terminar la primera vuelta aumentará su valor en uno, ósea valdrá "2".*
Segunda vuelta → *el bucle que ahora está en "2" al terminar la segunda vuelta aumentará su valor en uno, ósea valdrá "3".*
Tercera vuelta → *el bucle que ahora está en "3" al terminar la tercera vuelta aumentará su valor en uno, ósea valdrá "4".*
Cuarta vuelta → *el bucle no dará una cuarta vuelta, puesto que su valor actual es "4" y ha superado el valor de la variable "**CASOS**" que es "3".*

Entonces habiendo comprendido la **LÓGICA** de este bucle seguimos con la explicación del diagrama. Vemos que del objeto "**PARA**" se desprende una **LÍNEA** de secuencia hacia abajo, llega al objeto otra **LÍNEA** de secuencia por la parte izquierda, y por último se desprende del objeto una **LÍNEA** por la parte derecha.

La LÍNEA hacia abajo *→ indica que durante una vuelta, el bucle ejecutará todas las sentencias relacionadas con esta línea hasta encontrar la línea de retorno.*
La LÍNEA de la izquierda *→ esta es la línea de retorna, indica que mientras el bucle no haya superado el* ***LIMITE*** *volverá a ejecutar las sentencias que estén dentro, como se explica previamente.*
La LÍNEA de la derecha *→ esta línea es claramente la línea que seguirá el flujo del diagrama cuando hayan terminado todas las vueltas del bucle "****PARA****", en otras palabras, terminado el bucle, continua su camino.*

Por último explicare el contenido del bucle "**PARA**". Podemos ver que el código vuelve a solicitar un **DATO**, esta vez se trata de una "**ENTRADA**" o "**CASO DE PRUEBA**" propiamente dicho. A continuación el código trabajará con el **DATO** obtenido para generar una respuesta, a esto llamamos "**SALIDA**". Como vemos cada vez que el código genere una "**SALIDA**" este se ha de **IMPRIMIR**.

Finalmente y terminado el bucle "**PARA**" nuestro flujo llega a su fin. De esta manera se representa la **ESTRUCTURA BÁSICA** que se pide que el competidor implemente en cada una de sus **SOLUCIONES** a presentar.

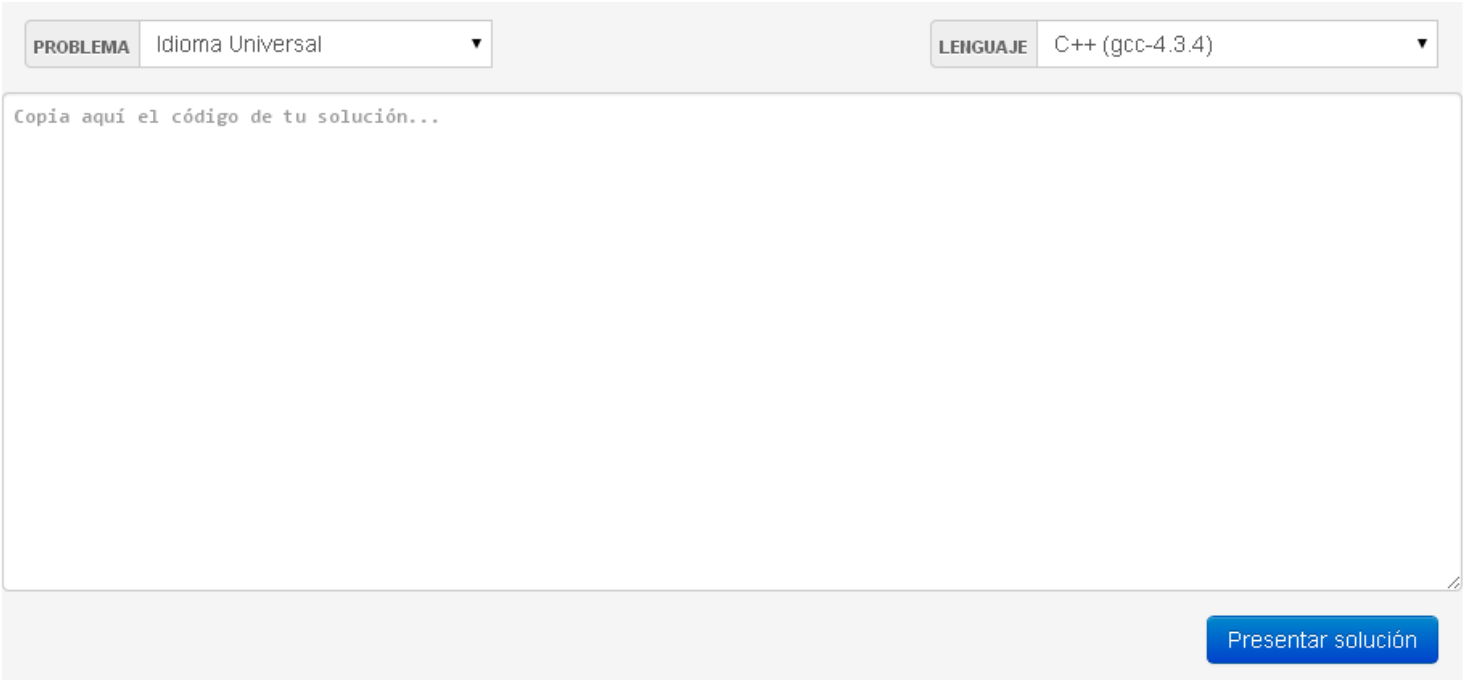
Transcripción a código fuente

Solo basta un poco de conocimiento de algún **LENGUAJE DE PROGRAMACIÓN** para comenzar a escribir **CÓDIGO FUENTE**, si usted conoce **JAVA** podrá entender a la perfección cada uno de los códigos escritos en la imagen anterior. Si usted conoce otro **LENGUAJE DE PROGRAMACIÓN** solo será necesario que escriba su propia estructura siguiendo la **LÓGICA** del flujo explicado anteriormente.

Cabe mencionar que esta no es la única forma de escribir el **CÓDIGO FUENTE** siguiendo la lógica explicada, para algunos programadores serán más cómodos sus propios métodos, lo único imprescindible es que **FUNCIONE** de la misma manera.

NOTA.- Si usted no está familiarizado aun con **JAVA** o algún otro lenguaje de programación, solo le bastará unas pocas lecciones que puede encontrar en artículos, videos, manuales entre otros contenidos de internet para comenzar sus actividades en nuestra web, ¡Ojo! **Usted no necesita ser un experto programador para resolver problemas aquí.**

Presentar la solución



Para presentar una **SOLUCIÓN** solo basta estar registrado en un **CONCURSO** o **ENTRENAMIENTO** y acceder al módulo de presentación de soluciones mediante el botón "**PRESENTAR SOLUCIÓN**", usted puede encontrar estos accesos en la descripción del problema o en lista de problemas del evento.

IMPORTANTE recordar que debe elegir el **LENGUAJE DE PROGRAMACIÓN** correcto en el selector "**LENGUAJE**", de no ser correcta su elección el compilador le dará un veredicto de tipo "**ERROR EN COMPILACIÓN**".

Sobre la solución

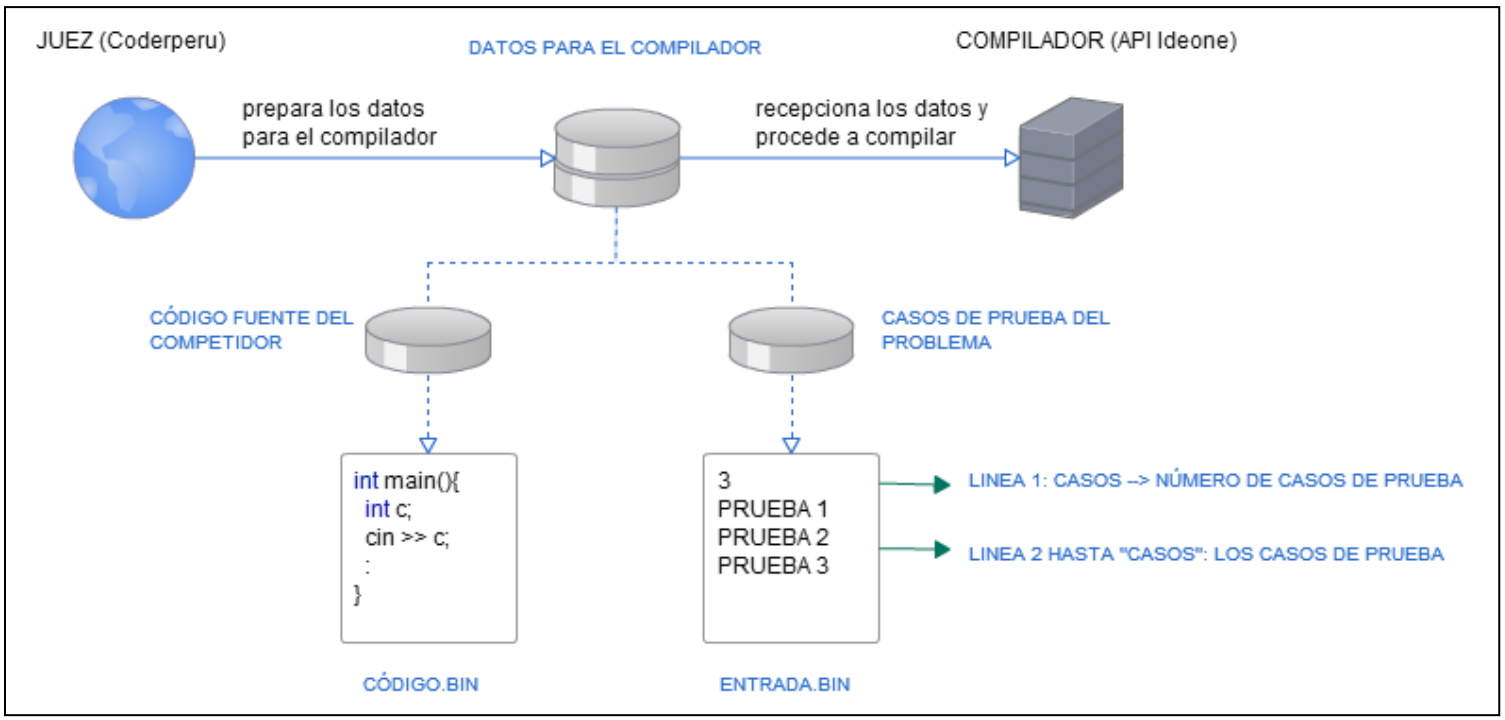
¿Cómo evalúa el juez el código de solución?

Cada vez que un competidor presenta una solución al juez del concurso, se desencadena una serie de actividades que termina por mostrarle al competidor el veredicto de su intento de resolver el problema, los tipos de veredicto son "**ACEPTADO**", "**ERROR EN SALIDA**", "**ERROR EN COMPILACIÓN**" y "**TIEMPO LIMITE EXCEDIDO**" según responda el compilador al juez tras validar y compilar el código fuente de dicha solución.

IDEONE.- Es importante señalar que todo código fuente presentado al juez de Coderperu es validado y compilado por ideone.com, esta herramienta nos permite tener respuestas a tiempo real en cuando a compilación de código fuente se refiere, lo que sirve de manera eficaz a los trabajos de coderperu.com

Siguiendo con la solución de problemas veremos entonces como es que funciona este sistema al que llamamos "**JUEZ**" y como evalúa la solución de un competidor. Como está claro que nuestro **JUEZ** trabaja de la mano con el compilador de **IDEONE** estas son las actividades que debemos mencionar.

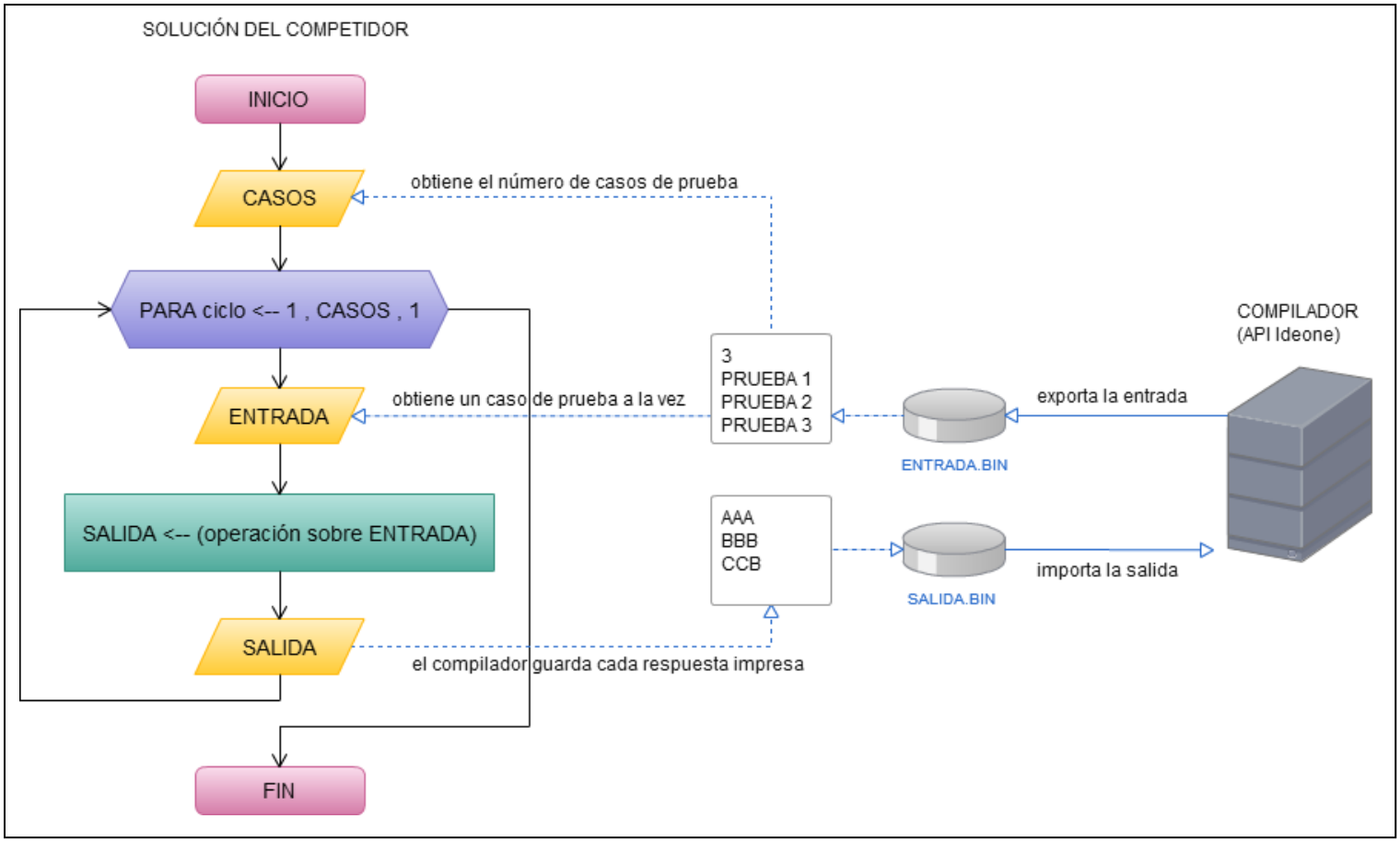
Comenzamos. El **JUEZ** prepara los datos necesarios para el trabajo del compilador, esta fase del proceso se representa en el siguiente gráfico.



Como podemos identificar los datos enviados al compilador contienen el **CÓDIGO FUENTE** que el competidor presentó y los **CASOS DE PRUEBA** con los que dicho código será ejecutado.

Tanto el **CÓDIGO FUENTE** como los **CASOS DE PRUEBA** son dos archivos de texto planos que contienen diferentes líneas con diferentes caracteres, estos archivos son transportados de forma binaria.

Es importante recordar que el **CÓDIGO FUENTE** del competidor debe contener un **BUCLE DE REPETICIONES** con un límite desconocido que servirá para evaluar los casos de prueba uno a la vez mediante cada repetición. Del mismo modo usted debe conocer que el archivo de **CASOS DE PRUEBA** contiene en la primera línea un número que representa el "**NÚMERO DE CASOS DE PRUEBA**" que servirá como el **LÍMITE** para el **BUCLE DE REPETICIONES** incrustado en el código fuente del competidor. ¡**OJO**! Que esto no significa que el código fuente se ejecutará tantas veces como el valor de "**CASOS DE PRUEBA**" sino que solo se ejecutará una solo vez y dentro de la ejecución habrá un bucle que se repetirá tantas veces como el valor de "**CASOS DE PRUEBA**". En la siguiente gráfica podemos aclarar estas definiciones.



Para comenzar a describir este gráfico primero ubiquemos los paquetes **ENTRADA.BIN** y **SALIDA.BIN**, estos paquetes son archivos de texto planos. El archivo **ENTRADA.BIN** contiene líneas de caracteres en diferentes cantidades y el archivo **SALIDA.BIN** se encuentra inicialmente vacío. La interacción entre el código fuente y estos archivos será de **LECTURA** y **ESCRITURA**. Entonces queda claro que al decir "**LEER**" y "**ESCRIBIR**" significa que estamos leyendo una determinada línea del archivo **ENTRADA** y escribiendo una determinada línea en el archivo **SALIDA** respectivamente.

A continuación representamos el **CÓDIGO FUENTE** que ha presentado el competidor como el diagrama de flujo que se ha explicado antes para su mayor compresión.

Como vemos al inicio de la compilación el código solicita un **DATO**, el mismo que se encuentra en la primera línea del archivo **ENTRADA** y cuyo valor es "**3**". Entonces se **LEE** la primera línea del archivo y se almacena lo leído en la variable **CASOS**.

Luego el código ejecuta el bucle "**PARA**" o **CICLO REPETITIVO**, siguiendo los parámetros "**1**", "**CASOS**", "**1**". Ya conocemos como funciona este bucle, lo que explicare a continuación será como el código fuente obtiene una **ENTRADA** del archivo **ENTRADA.BIN**, **PROCESA** el dato obtenido y genera una **SALIDA** que guardará en el archivo **SALIDA.BIN**.

ENTRADA

Siguiendo con la lectura del archivo **ENTRADA.BIN**, ya hemos obtenido el contenido de la **PRIMERA LÍNEA**, el cual ha sido el valor para la variable **CASOS**. Cuando el código solicite una nueva **LECTURA** al archivo **ENTRADA**, este responderá con el contenido de la **SEGUNDA LÍNEA.** ¿Y esto por qué? Bien, tras la primera lectura el **PUNTERO** del archivo quedo en la posición "**1**", al solicitar otra lectura, el puntero devolverá el contenido de la línea "**2**" y el puntero quedará en la posición "**2**".

De esta manera, cada lectura sobre el archivo **ENTRADA** ira recorriendo todos las líneas q este contenga hasta llegar a la ultima. Es importante mencionar que la lectura nativa del archivo **ENTRADA** es siempre hacia adelante y nunca hacia atrás, ósea el **PUNTERO AUMENTARA** siempre su valor y **NO DISMINUIRÁ**.

Entonces llevado esto a la práctica podemos decir que la variable entrada tomará los siguientes valores:

ENTRADA = "PRUEBA 1" → *en la primera vuelta del bucle.*
ENTRADA = "PRUEBA 2" → *en la segunda vuelta del bucle.*
ENTRADA = "PRUEBA 3" → *en la tercera y última vuelta del bucle.*

PROCESO

El proceso de los datos es básicamente el **CÓDIGO** creado por el competidor para dar solución al problema. Una vez obtenido el **DATO DE ENTRADA DE TURNO**, el código puede hacer uso de otras **VARIABLES**, **ARREGLOS**, **LISTAS DE DATOS** y **FUNCIONES DECLARADAS** para generar la respuesta que se le pide en la descripción del problema. Todo lo que ocurre en esta etapa es responsabilidad del **COMPETIDOR**.

SALIDA

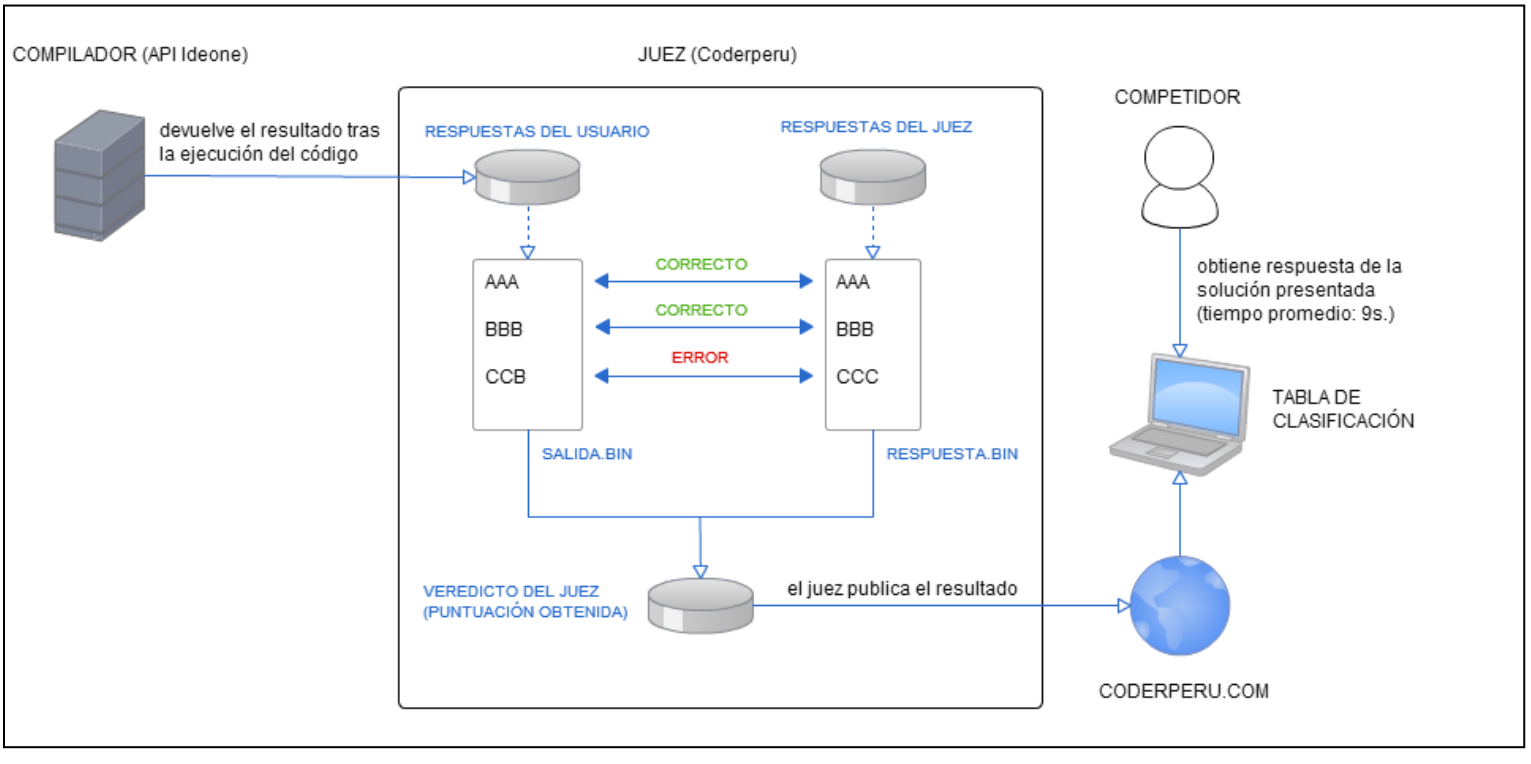
En esta última etapa de un ciclo del bucle el código hará la **IMPRESIÓN** mediante **CONSOLA** de una **RESPUESTA**, la misma que será **GUARDADA** en el archivo **SALIDA.BIN**. Es importante aclarar que cada **RESPUESTA** escrita en el archivo **SALIDA.BIN** debe ocupar **UNA LÍNEA ESPECIFICA**, en esta cuestión el código debe imprimir la **SALIDA + UN SALTO DE LÍNEA**, de esta manera la próxima escritura se dará en la **SIGUIENTE LÍNEA** y así sucesivamente.

¿Y EN LA PRÁCTICA? Bien pues la impresión de respuestas del código en este diagrama específico sería así:

VUELTA DEL BUCLE	ENTRADA	PROCESO	SALIDA
Vuelta # 1	**"PRUEBA 1"**	**<Procesa la entrada 1…>**	**Imprime "AAA"**
Vuelta # 2	**"PRUEBA 2"**	**<Procesa la entrada 2…>**	**Imprime "BBB"**
Vuelta # 3	**"PRUEBA 3"**	**<Procesa la entrada 3…>**	**Imprime "CCB"**

En síntesis podemos decir que en el archivo **SALIDA.BIN** se escribirán tantas respuestas como el valor de la variable **CASOS**, ósea que para cada "**CASO DE PRUEBA**" del archivo **ENTRADA.BIN** habrá una respuesta en el archivo **SALIDA.BIN**.

Nuestro **JUEZ** está pronto a terminar su trabajo, en esta última fase del proceso, el **JUEZ** validará el contenido del archivo **SALIDA.BIN**.



Como puede observar el **JUEZ** posee un archivo llamado **RESPUESTA.BIN**, este archivo contiene las respuestas correctas para cada "**CASO DE PRUEBA**" que contenía el archivo **ENTRADA.BIN** y que fue utilizado para generar las respuestas del archivo **SALIDA.BIN**. El proceso a continuación esta tan sencillo como una validación, nuestro **JUEZ** compara cada respuesta del archivo **SALIDA** con la del archivo **RESPUESTA**. Cabe mencionar que cada respuesta es una línea en ambos archivos de texto planos.

El trabajo del señor **JUEZ** termina justo aquí, con el **VEREDICTO** de la **PRESENTACIÓN**. Si durante la validación el **JUEZ** encontró una respuesta incorrecta como se ve en el gráfico entonces su veredicto será "**ERROR EN SALIDA**" de lo contrario si todas las respuestas del archivo **SALIDA** fueron correctas su veredicto será "**ACEPTADO**". De la grafica a la práctica, esta solución presentada tendría un error en la respuesta tres que es "**CCB**" y debería ser "**CCC**", veredicto entonces será de "**ERROR EN SALIDA**" y se vería así.

Soluciones presentadas

#	FECHA	CODER	PROBLEMA	<>	JUEZ
460	2013-12-25 18:56:02	Yuichi	Un peculiar Dominó	Java	Error en salida (3)

Ejemplo de una solución

Problema

Para entender mejor como analizar y resolver un problema, vamos a ver el siguiente caso. Comencemos por leer detenidamente el siguiente problema.

PROBLEMA:

Se necesita saber cuánto tiempo en segundos falta para completar un minuto, si se tiene una cantidad de segundos acumulados.

ENTRADA (INPUT):

La primera línea de la ENTRADA comienza con un número **M** que indica el número de casos del problema propuesto.

Cada caso está compuesto de la(s) siguiente(s) linea(s):

S: Será el tiempo en segundos que usted tiene acumulado.

RESTRICCIONES:

1 <= **M** <= 3 (el valor para los casos de prueba estará entre 1 y 3 ambos inclusive)
1 <= **S** <= 60 (el valor del dato **S** estará entre 1 y 60 ambos inclusive)

SALIDA (OUTPUT):

Imprimir la cantidad de segundos que le faltan para completar un minuto.

ENTRADA DE EJEMPLO:

ENTRADA (INPUT)	SALIDA (OUTPUT)
3 45 13 60	15 47 0

Solución

PROBLEMA → Esta claro lo que el problema nos pide que resolvamos, según una rápida comprensión del problema podemos decir que lo que debemos responder será la diferencia entre **60** y el número de segundos q nos dan por cada caso de prueba, ¿Cómo? Yo lo representaría así:

ENTRADA	PROCESO	SALIDA	VARIABLES
segundos = 45	**respuesta = 60 – segundos** **respuesta = 60 – 45**	**Salida = 15**	**segundos** **respuesta**

ENTRADA → La ENTRADA, que son **CASOS DE PRUEBA** del problema. Al momento de escribir el código de solución de este problema, la primera línea de la entrada que vamos a leer será la del número de casos de prueba, en este ejemplo es "3", y debajo de este, tres números diferentes que son los casos de prueba correspondientes. (***Vea la entrada de ejemplo***)

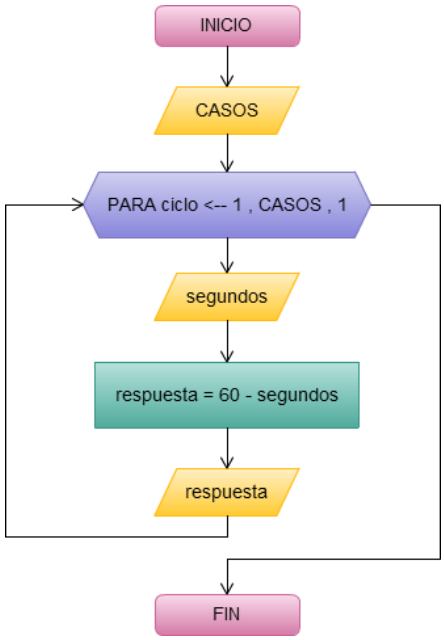
RESTRICCIONES → Como se ha mencionado antes las restricciones son los límites del problema y nos ayudan a calcular o deducir que valores máximos y mínimos pueden tomar los datos con los que trabajamos. En este problema tenemos dos restricciones muy claras y se interpretan así:

Para "**M**" que es el número de casos. Su valor mínimo será "**1**"y su valor máximo será "**3**", ósea que a lo mucho los casos de prueba pueden ser "3".

Para "**S**" su valor máximo puede llegar a ser "**60**" y no más, ni menos que "**1**", puesto que no correspondería un caso de prueba válido para este problema.

SALIDA → Tras nuestra deducción de solución del este problema, solo se pide que nuestro código de cómo respuesta el número de segundos que faltan para completar un minuto, nuestro diseño y código de solución quedarían así respectivamente:

DIAGRAMA DE FLUJO



CÓDIGO FUENTE (ESCRITO EN JAVA)

```
2
3   import java.util.Scanner;
4
5   public class Main {
6
7       public static void main(String[] args) {
8           Scanner entrada = new Scanner(System.in);
9           int casos = Integer.parseInt(entrada.nextLine());
10
11          for (int i = 0; i < casos; i++) {
12
13              /* Se declaran las variable a usar en la solución del problema */
14
15              int segundos;
16              int respuesta;
17
18              /* Se obtiene el número de segundos del caso de
19               prueba y se guarda en la variable "segundos" */
20
21              segundos = Integer.parseInt(entrada.nextLine());
22
23              /* Se opera con la variable "segundos" para obtener la diferencia
24               entre 60 y su valor y almacenarlo en la variable "respuesta" */
25
26              respuesta = 60 - segundos;
27
28              /* Mediante el uso de consola se imprime la respuesta */
29
30              System.out.println(respuesta);
31          }
32      }
33
34  }
```

De esta manera nuestro código de solución está listo para ser presentado al **JUEZ** de **CODERPERU**, solo hay un detalle mas que debes tener en cuanta y es que: Uses el **LENGUAJE DE PROGRAMACIÓN** que uses, siempre debes usar la clase principal o clase "**Main**" de tal lenguaje para que la compilación del código sea correcto, en el caso de **JAVA** observa en la parte superior del código la sentencia "**public class Main**", esta sintaxis está correcta.

Paso final

Si has comprendido a la perfección esta sesión, estás listo para resolver problemas en **CODERPERU**, intenta ingresando a la sección de "**EJERCICIOS**" del menú principal de nuestra web y resuelve los primeros tres **PROBLEMAS PROPUESTOS**.