

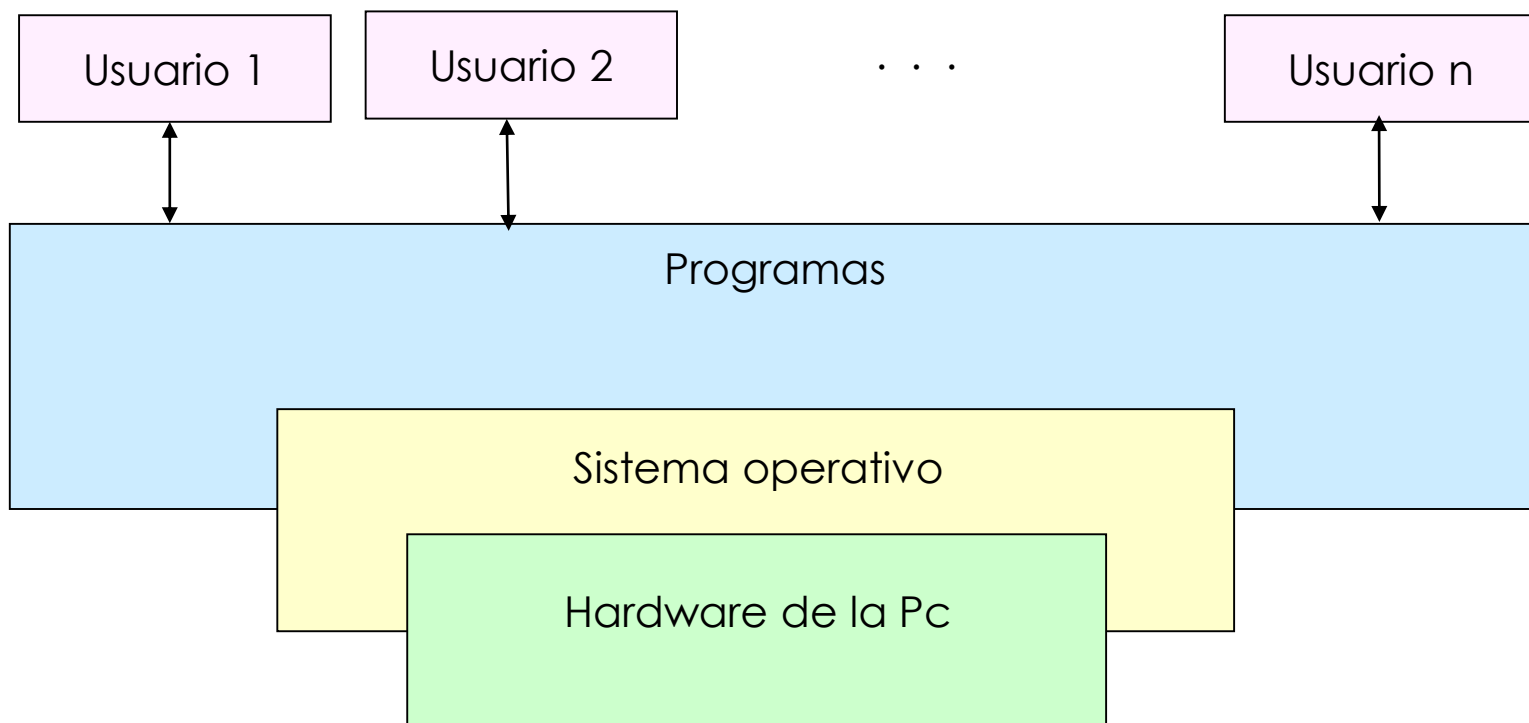
Programas

Programación I

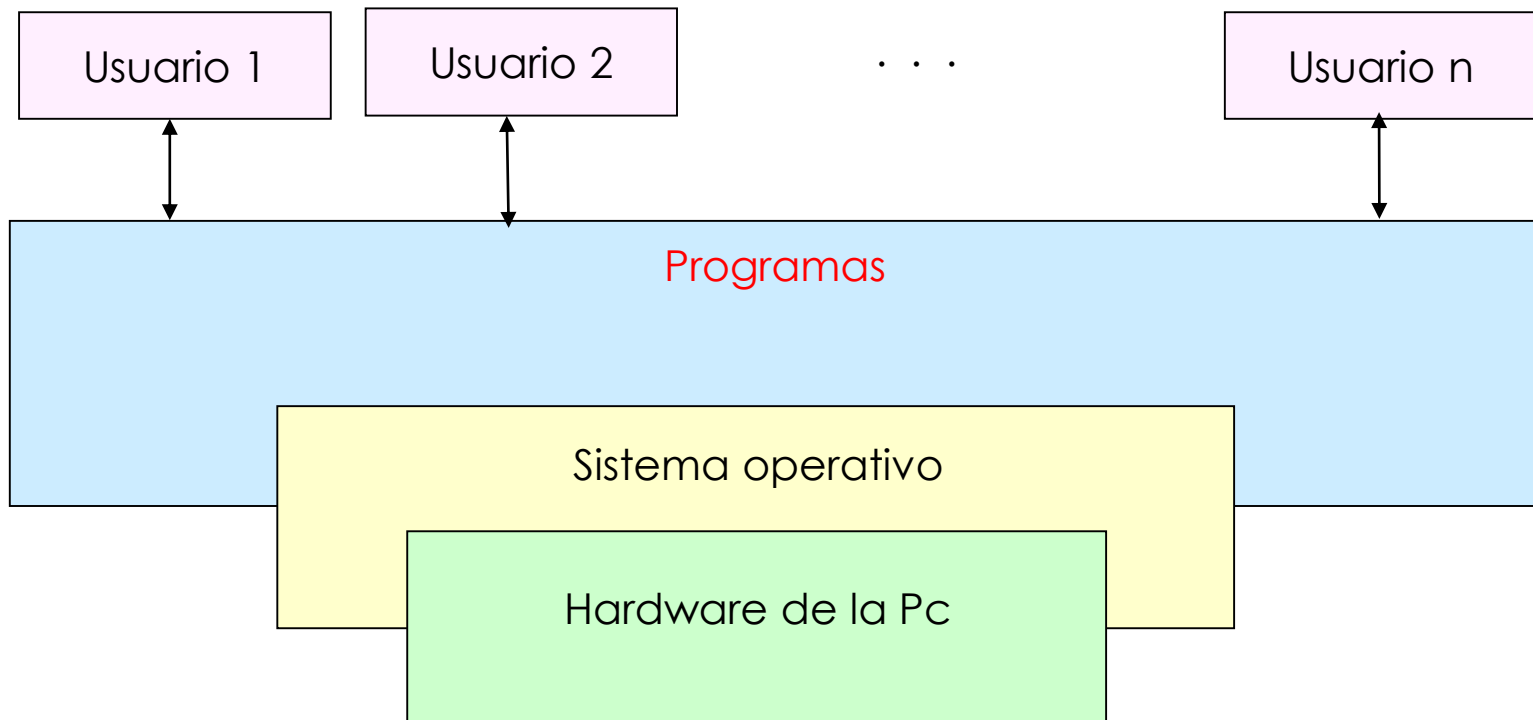
Objetivos del tema

- Comprender el concepto de programa
- Escribir/implementar nuestro primer programa

Componentes de un sistema de computación



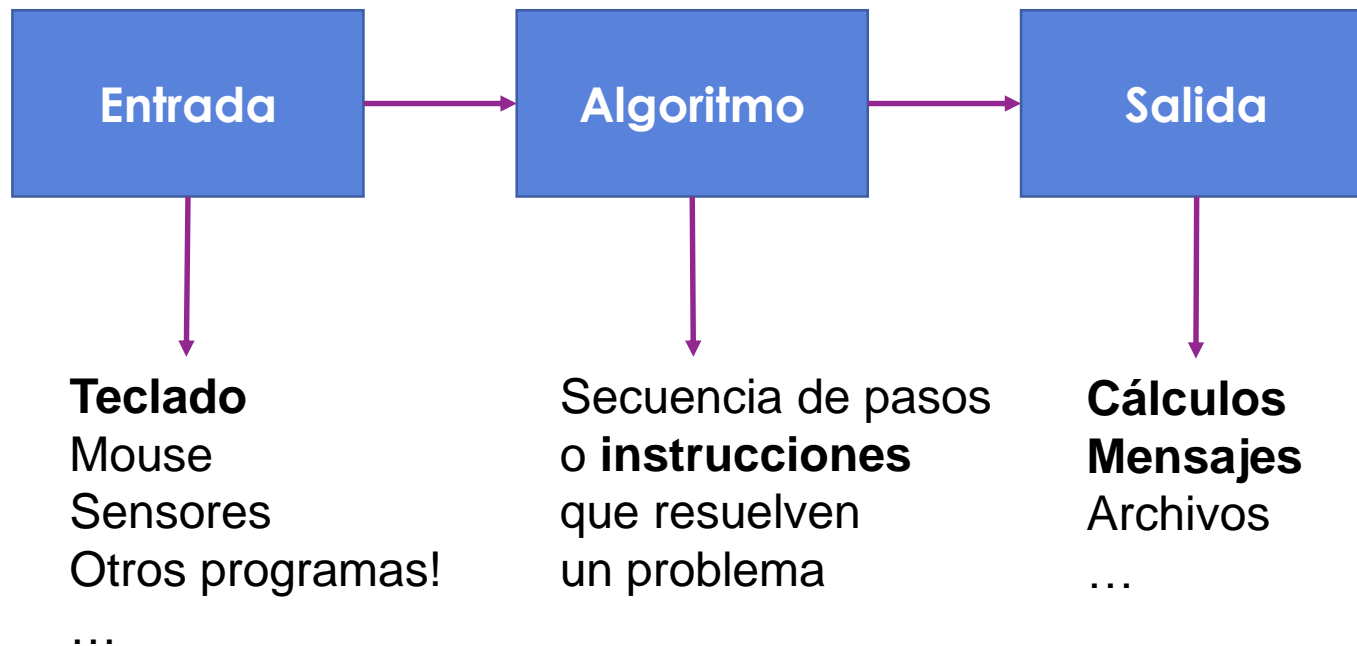
Componentes de un sistema de computación



Programas: lo que vamos a aprender a escribir en Programación!

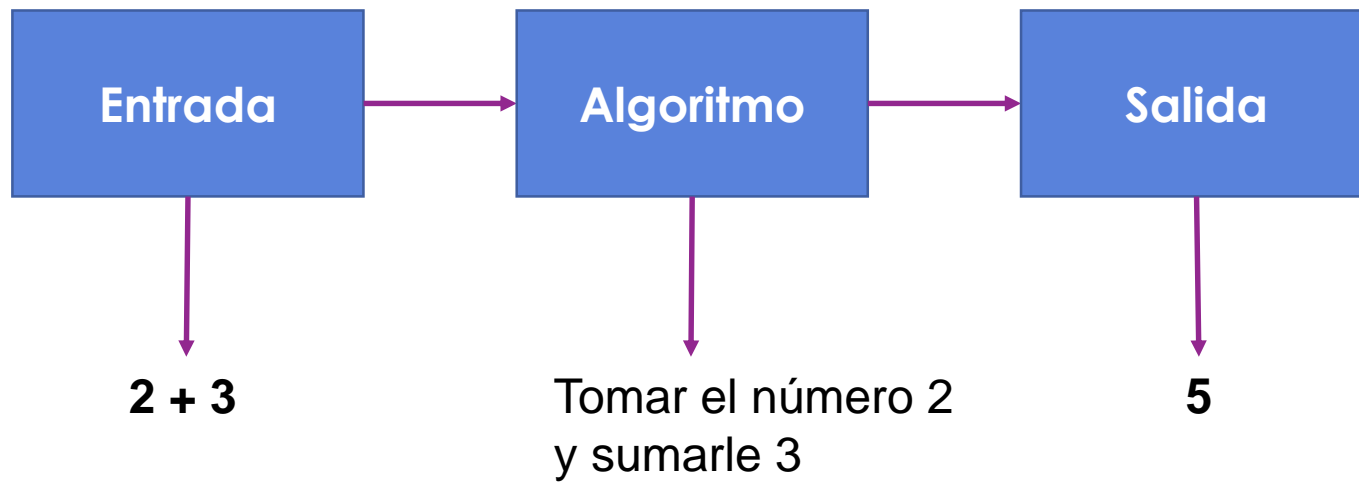
Programa de usuario

- Conjunto de **instrucciones** escritas en un **lenguaje** de programación que pueden ser **ejecutadas** por una computadora para realizar una determinada función de acuerdo a una entrada de datos.



Programa de usuario

- Calculadora



Lenguaje (Java)



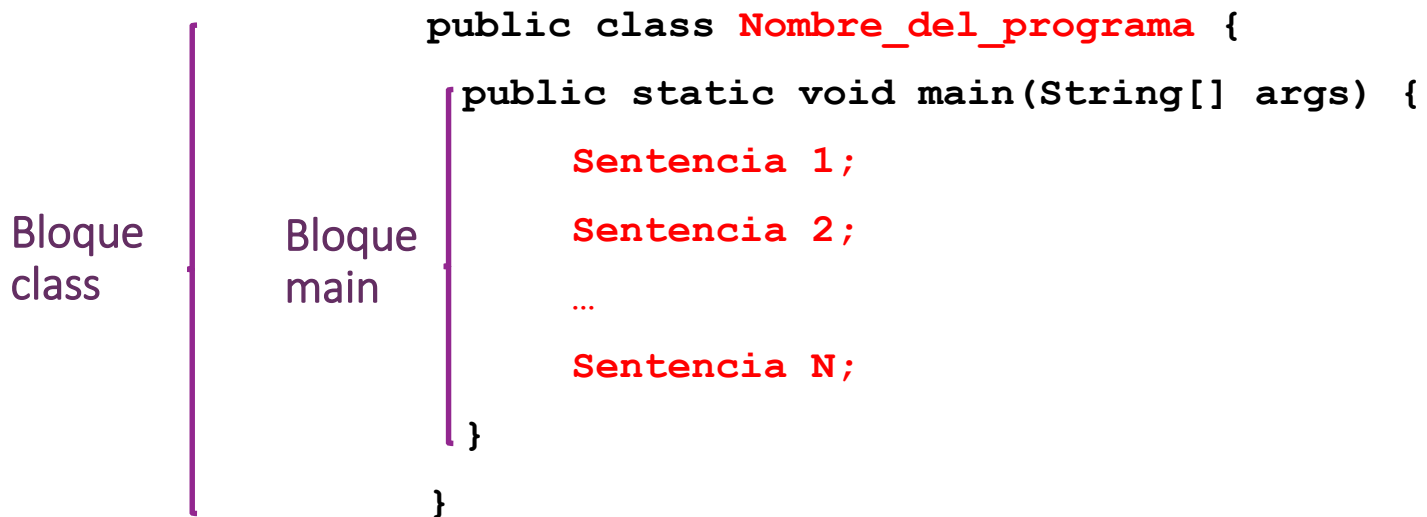
- Es un lenguaje de programación que define un conjunto de instrucciones mediante las cuales les podemos dar órdenes a la computadora
 - Mostrar un mensaje
 - Sumar/restar/dividir/multiplicar números
 - ...
- En estos tipos de lenguaje tenemos que escribir las instrucciones sin errores
 - Tendremos un chequeador de formato que nos ayudará a detectar esos errores
 - Cada programa se guarda en uno o más archivos de formato .java
- Por qué Java?
 - Muy usado en la comunidad (podemos consultar a mucha más gente!)
 - Es relativamente fácil encontrar errores en nuestro código
 - Otros aspectos técnicos que por ahora no son relevantes

Instrucciones (nuestro primer programa!)

- **Todos los programas** que vamos a escribir en Programación I tendrán la siguiente estructura:

```
public class Nombre_del_programa {  
    public static void main(String[] args) {  
        Sentencia 1;  
        Sentencia 2;  
        ...  
        Sentencia N;  
    }  
}
```


Instrucciones (nuestro primer programa!)



```
public class Nombre_del_programa {  
    public static void main(String[] args) {  
        Sentencia 1;  
        Sentencia 2;  
        ...  
        Sentencia N;  
    }  
}
```

The diagram illustrates the structure of a Java program. It shows two nested blocks: an outer 'Bloque class' and an inner 'Bloque main'. The 'Bloque class' is represented by a vertical line on the left, and the 'Bloque main' is represented by a vertical line on the right. The code is indented to show the nesting: the class definition is the outermost, followed by the main method, and then the statements inside the main method.

- Java nos obliga a definir dos bloques principales delimitados por llaves (`{}`), donde deberemos realizar el código de nuestros programas.
 - **public class**: propia de la sintaxis de Java (se explica más adelante), y seguido se coloca el nombre a nuestro programa (por **convención***, empieza con mayúscula)
 - **public static void main(String[] args)**: es por donde comienza a ejecutarse nuestro programa.
- **Siempre los usaremos de la misma manera!**

* la mayoría de los programadores de Java lo usa así

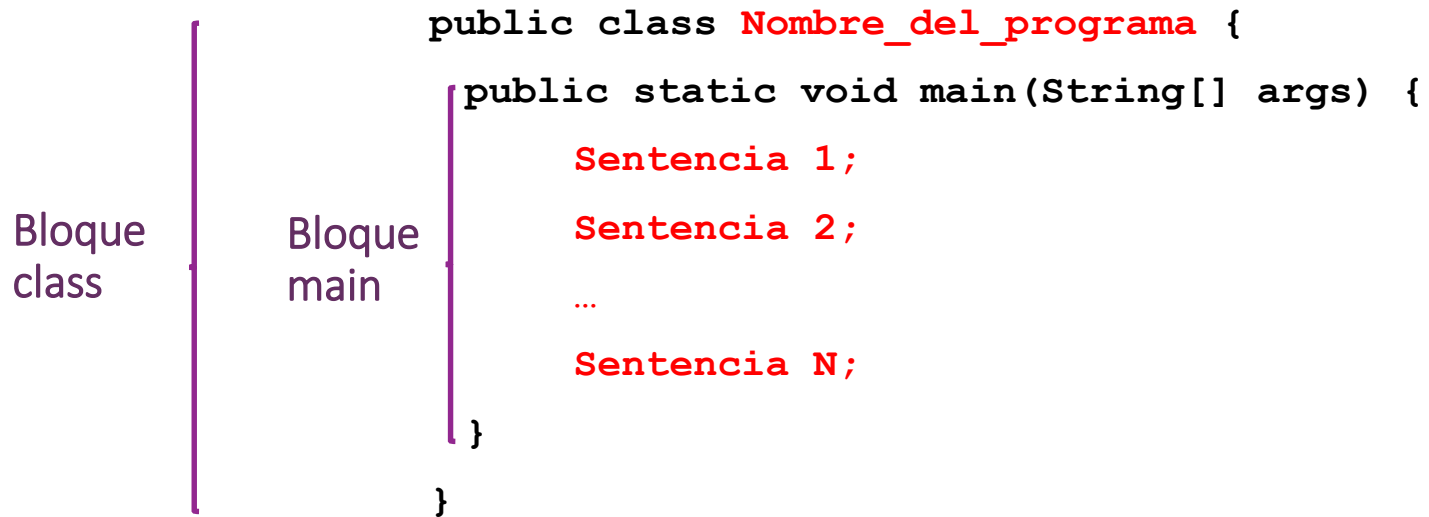
Instrucciones (nuestro primer programa!)

```
public class Nombre_del_programa {  
    public static void main(String[] args) {  
        Sentencia 1;  
        Sentencia 2;  
        ...  
        Sentencia N;  
    }  
}
```

The diagram illustrates the structure of a Java program using nested blocks. A vertical purple line on the left is labeled 'Bloque class'. Inside this block, another vertical purple line is labeled 'Bloque main'. The code is indented to show that the 'main' method is contained within the 'class' block. The code includes a class declaration, a static main method, and several statements (Sentencia 1, Sentencia 2, ..., Sentencia N) within the main method.

- Las sentencias son órdenes (instrucciones) que se le dan a la computadora
 - mostrar valores, realizar cálculos, pedir datos, etc.
- Se ejecutan de manera **secuencial y ordenada**
 - Es decir primero se ejecuta la Sentencia 1, luego la Sentencia 2 y así sucesivamente hasta la Sentencia N.
 - Luego de ejecutarse la Sentencia N, el programa sale del **bloque main** y termina su **ejecución**

Instrucciones (nuestro primer programa!)

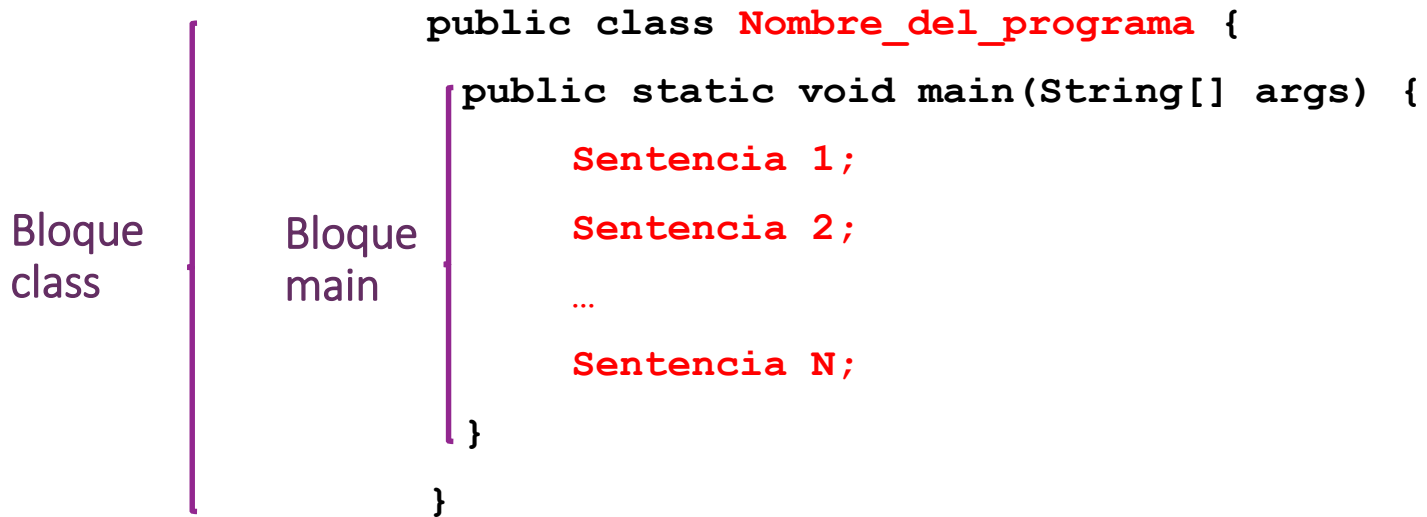


The diagram illustrates the structure of a Java program using nested vertical lines to represent blocks. On the left, a vertical line is labeled "Bloque class". To its right, another vertical line is labeled "Bloque main". The code is indented to match these blocks:

```
public class Nombre_del_programa {  
    public static void main(String[] args) {  
        Sentencia 1;  
        Sentencia 2;  
        ...  
        Sentencia N;  
    }  
}
```

- Nuestros programas **siempre** seguirán esta estructura.

Instrucciones (nuestro primer programa!)



```
public class Nombre_del_programa {  
    public static void main(String[] args) {  
        Sentencia 1;  
        Sentencia 2;  
        ...  
        Sentencia N;  
    }  
}
```

The diagram illustrates the structure of a Java program using nested blocks. A vertical purple line on the left is labeled 'Bloque class'. Inside this block, another vertical purple line is labeled 'Bloque main'. The code to the right of these lines shows the corresponding Java syntax: a public class named 'Nombre_del_programa' containing a public static void main method with several statements (Sentencia 1, Sentencia 2, ..., Sentencia N) enclosed in curly braces.

- Debemos introducir un Nombre de programa (**sin espacios ni caracteres especiales**).
 - Ej1_TP2
 - CalculadoraCuantica
- **Las sentencias que resuelvan el problema en particular** (objetivo de la materia)
 - Instrucciones propias de Java que **aprenderemos con el uso**.
 - La mayoría se pueden trasladar a otros lenguajes sin demasiadas modificaciones

Instrucciones (comentarios)

- Java los ignora, podemos escribir lo que queramos y como queramos
- entre `/*` y `*/` si ocupa más de 1 línea
- después de `//` si es de 1 sola línea
- **Buena práctica** para hacer todo tipo de anotaciones sobre el código que hicimos

```
/* Este es mi primer programa!
*/
public class Nombre_del_programa {
    public static void main(String[] args) {
        Sentencia 1; //mi primera sentencia!
        Sentencia 2;
        ...
        Sentencia N;
    }
}
```

Instrucciones (nuestro primer programa!)

- **Nuestro primer programa: Hola mundo!**
- **Nombre de programa:** HolaMundo (puede ser cualquier otro!)
- **Una única sentencia** que imprime un mensaje por consola
- **Atención:** todas las sentencias en Java deben finalizar con ";". Si no lo ponemos, se nos informará un error!

```
//Mi primer programa!  
public class HolaMundo{  
    public static void main(String[] args) {  
        System.out.println("Hola mundo!");  
    }  
}
```

Instrucciones (nuestro primer programa!)

- **Nuestro primer programa: Hola mundo!**
- **System.out.println()** es una instrucción provista por Java para imprimir (mostrar) un mensaje en la consola
- Lo que escribamos entre "" **dentro de los ()** se mostrará **textualmente** en la pantalla.
- Puede imprimir otras cosas que veremos más adelante

```
//Mi primer programa!  
public class HolaMundo{  
    public static void main(String[] args) {  
        System.out.println("Hola mundo!");  
    }  
}
```

Instrucciones (nuestro primer programa!)

- **Nuestro primer programa: Hola mundo!**
- El programa **siempre** comienza a ejecutarse **por la primera línea del bloque main**, continuando por las siguientes en orden
- En este caso es la única, así que luego de imprimir por pantalla el mensaje Hola mundo!, el programa finalizará.

```
//Mi primer programa!  
public class HolaMundo{  
    public static void main(String[] args) {  
        —→ System.out.println("Hola mundo!");  
    }  
}
```


Instrucciones (nuestro segundo programa!)

- **Nuestro segundo programa: Hola mundo! Estoy programando!**
- El programa **siempre** comienza a ejecutarse por **la primera línea del bloque main**, continuando por las siguientes en orden.
- En este caso tenemos **dos sentencias**, separadas por ";", por lo que se ejecutará la impresión del mensaje **Hola, mundo!** y al concluir esa sentencia se mostrará el mensaje **Estoy programando!**

```
//Mi segundo programa!
```

```
public class HolaMundo{
```

```
    public static void main(String[] args) {
```

```
        System.out.println("Hola mundo!");
```

```
        System.out.println("Estoy programando!");
```

```
    }
```

```
}
```