PROGRAMACIÓN

UT2: Identificación de los elementos de un programa informático

1. Métodos.

Ya hemos visto cómo se definen los elementos básicos de la programación en el tema anterior.

Ahora vamos a ver cómo usarlos correctamente.

Vamos a empezar analizando el programa que usamos anteriormente:

```
public class App {
    public static void main(String[] args) throws Exception {
        System.out.println("Hello, World!");
    }
}
```

Este código sirve para escribir “***Hello World!”*** en la pantalla. Para empezar a entender el código:

♦ La primera línea (`public class App`) declara el nombre de la clase del código. Más adelante se explicará que significa el concepto de clase; por ahora entenderemos que el nombre de la clase es el nombre del programa.

Además, el archivo tiene que llamarse obligatoriamente App.java (ya que así hemos llamado a nuestra clase principal

♦ La línea `public static void main(String[] args)`, sirve para indicar el inicio del método **main**. Este es el método principal, que contiene las instrucciones que se ejecutarán cuando el programa arranque. Es decir, **lo que está tras las llaves del main, es el programa en sí**.

♦ La instrucción `System.out.println` sirve para escribir en pantalla (consola de comandos). Como lo que escribimos es un texto, se encierra entre comillas.

.

Vamos a ver ahora la definición de método.

Los métodos sirven para **agrupar instrucciones de código** y luego este conjunto de instrucciones **pueden ser llamadas** cuantas veces sean necesarias simplemente haciendo la “**llamada al método**“, esto nos permite **reutilizar código** y **resolver problemas cada vez más complejos** gracias al **aumento de abstracción** sobre un problema.

Ya hemos visto el método principal **main,** pero normalmente en cada programa vamos a tener que escribir múltiples métodos, y a estos llamarlos desde el propio main.

Veamos como se escribe un método:

```
public class App {
    public static void main(String[] args) throws Exception {

        // Código del metodo main
        metodo1();
    }

    static void metodo1(){
        System.out.println("hola");
    }
}
```

Como vemos el nuevo método se declara **fuera** del método main, **pero dentro de la clase.**

Este nuevo método podrá se llamado desde el propio main para que se ejecute su código, o desde cualquier otro método **dentro de la clase**, como veremos más adelante.

¿Cómo se estructura un método?

```
    public static void metodo1(){

    }
```

Para escribir un método debemos tener en cuenta lo siguiente:

MODIFICADORES DE ACCESO

Debemos definir si nuestro método va a ser accesible para todo nuestro proyecto, o solo para la clase donde está definido.

- **private**: El modificador de acceso privado especifica campos y métodos de una clase que no son accesible fuera de la unidad donde se declara la clase.
- **public**: El modificador de acceso público denota campos y métodos que son de libre acceso desde cualquier otra parte de un programa.
- **protected**: El modificador de acceso protegido se utiliza para indicar métodos y campos con visibilidad sólo en la clase actual y sus clases derivadas (o subclases)

Si no le indicamos ninguno de los 3, por defecto será público.

A continuación, añadiremos la palabra reservada **static** si el método es estático o no. Esto quiere decir que no necesita que instanciemos un objeto de la clase para poder llamarlo.

De momento la pondremos por defecto hasta que veamos programación más avanzada.

RETORNO (SALIDA)

El siguiente elemento que debemos indicar es qué va a devolver nuestro método, es decir, la salida de este.

Un método puede devolver 1 único objeto de la clase que deseemos, o en caso de no querer que devuelva nada, no devolver nada.

- Si no queremos salida, usaremos la palabra reserva **void**.
- Si, por ejemplo, queremos que devuelva un entero, indicaremos que la salida es un **int**.
  En caso de que la salida no sea void, el método deberá contener en un interior una instrucción de salida.

Esta instrucción consta de la palabra reservada **return** y a continuación el dato que vamos a devolver:

```
static int metodo1(){
        return 5;
    }


static int metodo1(){
        int ejemplo = 5;
        return ejemplo;
    }
```

Lo siguiente será el nombre del método. Este lo usaremos para llamarlo desde otros métodos. Debe empezar por minúsculas, ser una única palabra y NO se puede repetir.

ENTRADA

Lo último será que le indiquemos que parámetros (datos) recibe el método como entrada, por ejemplo, si nuestro método realiza una suma de dos números, le indicaremos que recibe dos números. Un método puede recibir ninguno, uno o múltiples datos de entrada, y no tienen por qué ser del mismo tipo.

```
static void imprimirPalabra(String palabra){
       System.out.println(palabra);
   }


static int metodoSuma(int a, int b){
       return a + b;
   }
```

2. Clases

Una clase es una plantilla para la creación de objetos. Podemos decir que son la esencia de Java.

Dentro de las clases definimos los datos y el código que actúa sobre estos datos.

Las clases contienen la descripción del objeto al que sirven como plantilla y los métodos de este objeto.

Por ejemplo, si tenemos la clase Matemáticas, esta tendrá los métodos Suma, Resta, Multiplicación, División, …

Si tenemos la clase coche, necesitaremos definir sus atributos, número de ruedas, marca, motor, manual o automático, … y sus métodos; acelerar, frenar, cambiar de marcha, ….

De momento nos centraremos en la clase principal que se crea al iniciar nuestro proyecto, y que tiene el mismo proyecto que este.

Esta clase debe contener el método main, que es el encargado de ejecutar el programa, haciendo a esta clase la principal.

```
public class App {
    public static void main(String[] args) throws Exception {
    }
    static void metodo1(){
    }
    static int metodoSuma(int a, int b){
        return a + b;
    }
    static void imprimirPalabra(String palabra){
    }
}
```

Todos los elementos de la clase deben ir dentro de las llaves principales, sino no podremos usarlos.

3. Partes de un programa.

Cada programa se compone de una o más clases y obligatoriamente main() debe ser uno de los métodos de la clase principal.

Un programa en java puede incluir:

PAQUETES

- Nombre del paquete al que pertenece.

Un **package** es una agrupación de clases. Es parecido a una "caja" que contiene las clases que queramos mantener en un solo lugar. También podría decirse que los packages es el equivalente a las librerías en otros lenguajes. Esta parte del código no es obligatoria, es sólo si lo necesita usar nuestro programa

Se escriben al principio de todo. Son las primeras líneas de código.

IMPORTACIONES

- Declaraciones para importar clases o paquetes.

En ocasiones nuestros programas requerirán utilizar clases existentes en otros Packages, y esto se puede lograr con Import. En el ejemplo vemos un import que es el de la clase que contiene el método para imprimir por pantalla, la clase System que pertenece a la librería java.lang (ésta es la librería estándar de Java y no es necesario importarla)

```
import java.lang.Math;
```

Se escriben debajo de los packages y antes de declarar la clase principal.

CLASES

- Declaraciones de clases.

Java puede crear diferentes tipos de clases: privadas, públicas y protegidas; y se utilizan de acuerdo con conveniencia de la estructura de nuestro programa. El nombre de la clase pública debe coincidir con el nombre del archivo, y se pone con la primera letra en mayúscula.

```
public class App {
...
}
```

METODO MAIN

- El método main( ).

Un programa en Java se puede considerar una colección de clases, en la que al menos una de ellas incluya de manera obligatoria al método main(), también indica el comienzo del programa y requiere la sintaxis:

```
public static void main(String[] args) throws Exception {
...
}
```

VARIABLES

- Zona de declaración de variables.

Java maneja tres tipos de variables: de instancia, de clase y locales.

- Las variables de instancia son las que se usan para guardar valores o atributos de un objeto en particular.
- Las variables de clase son las que guardan valores o atributos de la clase. **Se pueden usar en toda la clase**.
- Las variables locales son las que se **declaran en una función o método y solamente las puede utilizar esa función o método**, de ahí el nombre de locales.

OTROS

- Comentarios.

Éstos pueden incluirse en cualquier parte del código. Sus líneas serán completamente ignoradas por el compilador, o sea que no afectarán para nada nuestro programa.

- Métodos definidos por el usuario dentro de las clases.

En Java los métodos son los que utilizamos para realizar alguna tarea en específico. Van dentro de la clase.

4. Identificadores

En los lenguajes de programación, los identificadores (como su nombre lo indica) se utilizan con fines de identificación. En Java, un identificador puede ser un nombre de clase, un nombre de método o un nombre de variable

Por ejemplo, en este código java, tenemos 5 identificadores, a saber:

```
public class App {

    public static void main(String[] args){

        int ejemplo = 29;

    }

}
```

App: nombre de clase.

main: nombre del método.

String: tipo de objeto predefinido, indica de que clase será “args”.

args: nombre de la variable de entrada.

ejemplo: nombre de la variable local.

REGLAS PARA DEFINIR IDENTIFICADORES

- Los únicos caracteres permitidos para los identificadores son todos los caracteres alfanuméricos ([AZ], [az], [0-9]), “$” (signo de dólar) y ‘_’ (guión bajo).
- Los identificadores no deben comenzar con dígitos ([0-9]).
- Los identificadores de Java distinguen entre mayúsculas y minúsculas.
- No hay límite en la longitud del identificador, pero es aconsejable usar solamente una longitud óptima de 4 a 15 caracteres.
- Las palabras reservadas no se pueden usar como un identificador, ni los valores lógicos true o false.
- No pueden ser iguales a otro identificador declarado en el mismo ámbito.

- **Es de vital importancia elegir nombres adecuados para los identificadores. El motivo es facilitar la lectura del código todo lo posible. Identificadores como a, b, c, objeto, uno, etc. no son adecuados porque no indican nada acerca del elemento identificad.**
- IMPORTANTE: Por convenio:
    - Los nombres de las variables y los métodos deberían empezar por una letra minúscula y los de las clases por mayúscula.
    - Si el identificador está formado por varias palabras, la primera se escribe en minúsculas (excepto para las clases) y el resto de las palabras se hace empezar por mayúscula (por ejemplo: anyoDeCreación).
    - Estas reglas no son obligatorias, pero son convenientes ya que ayudan al proceso de codificación de un programa, así como a su legibilidad. Es más sencillo distinguir entre clases y métodos o variables.