

Estructura de una clase:

```
package paquete; // imports public class Clase{
    public static void main(String[] args) {
        // código que se ejecuta en la app
    }
}
```

Firma de una función: `public static TIPORETORNO nombreFuncion(.... Parámetros)`
// en la firma de la función se incluye para cada parámetro el tipo y el nombre

Invocar a una función en otra clase y paquete
// es necesario hacer el import del paquete
`Clase.nombreFuncion();`

Estructuras repetitivas

```
for( inicialización;
    condición;incremento){..}
for(int i=0; i< 3; i++)
{ // bloque de código }
```

```
while(condicion){
    // bloque de código }
```

```
do{
    // bloque de código
}while(condicion);
```

Sintaxis de Scanner

```
Scanner lectorTeclado = new Scanner(System.in);
lectorTeclado.nextInt();
lectorTeclado.nextLine();
lectorTeclado.nextDouble();
```

Sintaxis de switch

```
switch (variable) {
    case 1:
        // ...
        break;
    case 2:
        // ...
        break;
    default:
        // ...
}
```

Sintaxis de if

```
if (condición) {
    // ...
} else if (otraCondición) {
    // ...
} else {
    // ...
}
```

String - métodos para conocer

`int length()`: Retorna la longitud de la cadena.

`boolean isEmpty()`: Retorna true si la cadena está vacía (longitud cero), de lo contrario retorna false.

`char charAt(int index)`: Devuelve el carácter en la posición especificada.

`int indexOf(String str)`: Devuelve el índice de la primera aparición de una subcadena. Si no se encuentra, devuelve -1.

`int lastIndexOf(String str)`: Devuelve el índice de la última aparición de una subcadena.

`boolean equals(Object obj)`: Compara el valor de dos cadenas, respetando las mayúsculas y minúsculas.

`boolean equalsIgnoreCase(String anotherString)`: Compara dos cadenas ignorando diferencias entre mayúsculas y minúsculas.

`int compareTo(String anotherString)`: Compara dos cadenas lexicográficamente. Retorna un valor negativo, cero o positivo si esta cadena es menor, igual o mayor que anotherString.

`boolean startsWith(String prefix)`: Verifica si la cadena comienza con el prefijo especificado.

`boolean endsWith(String suffix)`: Verifica si la cadena termina con el sufijo especificado. `String substring(int beginIndex)`: Retorna una subcadena que comienza en el índice `beginIndex` hasta el final de la cadena.

`String substring(int beginIndex, int endIndex)`: Retorna una subcadena desde `beginIndex` hasta `endIndex`.

`String toLowerCase()`: Convierte todos los caracteres a minúsculas.

`String toUpperCase()`: Convierte todos los caracteres a mayúsculas.

`String trim()`: Elimina los espacios en blanco iniciales y finales de la cadena.

`String replace(char oldChar, char newChar)`: Reemplaza todas las ocurrencias de `oldChar` por `newChar`.

`String replaceAll(String regex, String replacement)`: Reemplaza todas las ocurrencias que coincidan con una expresión regular por la cadena dada.

`String[] split(String regex)`: Divide la cadena en un arreglo usando el delimitador basado en una expresión regular. `char[] toCharArray()`: Convierte la cadena en un arreglo de caracteres.

`static String.valueOf(Object obj)`: Convierte varios tipos de datos, como enteros o caracteres, en una cadena. También funciona para objetos.

Math. Métodos para conocer

`int abs(int a)` Devuelve el valor absoluto de un número entero.

`double abs(double a)` Devuelve el valor absoluto de un número de punto flotante.

`int max(int a, int b)` Devuelve el mayor de dos números enteros.

`int min(int a, int b)` Devuelve el menor de dos números enteros.

`double max(double a, double b)` Devuelve el mayor de dos números de punto flotante.

`double min(double a, double b)` Devuelve el menor de dos números de punto flotante.

`long round(double a)` Redondea el valor de punto flotante al entero más cercano.

`double pow(double a, double b)` Devuelve el resultado de elevar `a` a la potencia de `b`.

`double sqrt(double a)` Devuelve la raíz cuadrada de un número.

`double random()` Devuelve un número aleatorio en el rango [0.0, 1.0).

`double ceil(double a)` Devuelve el menor número entero mayor o igual a `a`.

`double floor(double a)` Devuelve el mayor número entero menor o igual a `a`.

`double rint(double a)` Devuelve el número entero más cercano a `a`, como un valor de punto flotante.

Random – números aleatorios

```
Random r = new Random();
```

```
int n = r.nextInt();    // cualquier int
```

```
int n2 = r.nextInt(10); // 0 a 9
```

```
double d = r.nextDouble(); // 0.0 a 1.0
```

```
boolean b = r.nextBoolean();
```

Conversión char ↔ int

```
char c = 'A';
```

```
int codigo = (int)c;    // → 65
```

```
int x = 66;
```

```
char letra = (char)x;    // → 'B'
```

Casting entre primitivos

```
int x = (int) 5.89;    // → 5
```

```
double d = (double) 10; // → 10.0
```

String → número

```
int n = Integer.parseInt("123");
```

```
double dd = Double.parseDouble("3.14");
```

Número → String

```
String s = String.valueOf(123);
```