

TEMA1

TIPOS DE DATOS

Para enteros: byte,short,long,int

Para tipos de coma flotante: float,double

Para booleanos: boolean

Para carácter: char

CARACTERES ESPECIALES

CARACTER	SIGNIFICADO
\b	Retroceso
\t	Tabulador
\n	Nueva Línea
\r	Retorno de Carro
“	Dobles comillas
‘	Comillas simples
\	Barra inclinada

CASTING

Para poder realizar asignaciones entre tipos distintos

```
public static void main(String[] args) {  
    byte n1=100, n2=100, n3;  
    n3=(byte) (n1*n2/100);  
    /*El resultado de esta operación es un número que está fuera  
    }del rango de byte por lo que debemos de convertirlo a byte|
```

OPERADORES

OPERADOR	DESCRIPCIÓN
+	Suma dos operandos

-	Resta dos operandos
*	Multiplica dos operandos
/	Divide dos operandos
%	Calcula el resto

INCREMENTALES

```

public static void main(String[] args) {
    int x=5, y=5;
    System.out.println(++x);
    //Incrementa en 1 y evalua después
    System.out.println(y++);
    //Incrementa en 1 y evalua antes
    System.out.println(--x);
    //Decrementa en 1 y evalua después
    System.out.println(x--);
    //Decrementa en 1 y evalua antes
}

```

LÓGICOS

```

public static void main(String[] args) {
    boolean llueve,tareas,biblioteca,salir;
    llueve=false;
    tareas=true;
    biblioteca=false;

    salir=(llueve==false && tareas==true)|| (biblioteca ==true);
    System.out.println("¿Puedes salir? "+salir);

    /* Condicion 1 y Condicion 2 Verdaderas: &&, Condicion1 o Condicion 2 Verdaderas: ||
    Condicion falsa: !Condicion
    */
}

```

ASIGNACIÓN

OPERADOR	USO	DESCRIPCIÓN
=	X=Y	Asigna x el valor de y
+=	X+=y	X=x+y
-=	x-=y	X=x-y
=	X=y	X=x*y
/=	x/=y	X=x/y

```

public static void main(String[] args) {
    Scanner teclado = new Scanner(System.in);
    float kgManzanas1, kgPeras1, ingresos1, kgManzanas2, kgPeras2, ingresos2, beneficioAnual;
    final float PRECIOMANZANAS = 2.35f;
    final float PRECIOPERAS = 1.95f;
    //Introduzco los datos por teclado mediante Scanner
    System.out.println(x:"Introduce los kilos de manzanas del primer semestre: ");
    kgManzanas1 = teclado.nextFloat();

    System.out.println(x:"Introduce los kilos de peras del primer semestre: ");
    kgPeras1 = teclado.nextFloat();

    System.out.println(x:"Introduce los kilos de manzanas del segundo semestre: ");
    kgManzanas2 = teclado.nextFloat();

    System.out.println(x:"Introduce los kilos de peras del segundo semestre: ");
    kgPeras2 = teclado.nextFloat();

    //Cálculo de los ingresos por semestres
    ingresos1 = (kgManzanas1 * PRECIOMANZANAS) + (kgPeras1 * PRECIOPERAS);
    ingresos2 = (kgManzanas2 * PRECIOMANZANAS) + (kgPeras2 * PRECIOPERAS);

    //Cálculo beneficios anuales
    beneficioAnual = ingresos1 + ingresos2;
    System.out.println(x:" ");
    //Muestro los resultados de los ingresos de los dos semestres con el ingreso anual
    System.out.printf("Los ingresos del primer semestre son: %.2f euros \nLos ingresos del segundo semestre son: %.2f euros "
        + "\nEl beneficio anual es: %.2f euros", args: ingresos1, args: ingresos2, args: beneficioAnual);
}

```

TERNARIO

Devuelve un valor que se selecciona de dos posibles. Dependerá de expresión racional o lógica, que puede tomar o verdadero o falso

```

public static void main(String[] args) {
    //Ejemplo ternario
    int a,b;
    a=3<5 ? 1:-1;

    System.out.println("a vale "+a);

    b= a==7?10:20;

    System.out.println("b vale "+b);

    //Programa que calcule el valor absoluto de un número
    int num=11;
    int valorAbs=num>0?num:-num;

    System.out.println("El valor absoluto de "+num+" es: "+valorAbs);
}

```

CONSTANTES

Es una variable de solo lectura, una vez que asignemos un valor, no se podrá cambiar

```

final float PRECIOMANZANAS = 2.35f;
final float PRECIOPERAS = 1.95f;

```

ORDEN PR

%c: Escribe carácter

%s: Escribe Cadena de texto

%d: Escribe entero

%f: Escribe un número en coma flotante

%e: Escribe un número en coma flotante en notación científica

```
System.out.println(x: " ");
//Muestro los resultados de los ingresos de los dos semestres con el ingreso anual
System.out.printf("Los ingresos del primer semestre son: %.2f euros \nLos ingresos del segundo semestre son: %.2f euros "
    + "\nEl beneficio anual es: %.2f euros", args: ingresos1, args: ingresos2, args: beneficioAnual);
```

LECTURA POR TECLADO

JOPTIONPANE

```
String texto;
int num;
texto = JOptionPane.showInputDialog(message: "Escribe un número");
num = Integer.parseInt(s: texto);
System.out.println("Has introducido el número "+num);
//Mostrando el resultado en un cuadro de diálogo
JOptionPane.showMessageDialog(parentComponent: null, "Has introducido el número "+num);

//Mensaje de advertencia
JOptionPane.showMessageDialog(parentComponent: null, message: "Advertencia", title: "Peligro", messageType: JOptionPane.WARNING_MESSAGE);
```

SCANNER

```
int edad;
String nombre, apellido;
Scanner teclado=new Scanner( source: System.in);

System.out.println(x: "Introduce tu nombre: ");
nombre=teclado.nextLine();

System.out.println(x: "Introduce tus apellidos: ");
apellido=teclado.nextLine();

System.out.println(x: "Introduce tu edad: ");
edad=teclado.nextInt();
System.out.println("Nombre: "+nombre+"\nApellidos: "+apellido+"\nEdad: "+edad);
```

SYSTEM

```
InputStreamReader isr= new InputStreamReader( in: System.in);
BufferedReader br= new BufferedReader( in: isr);
System.out.println(x: "Introduce un número: ");
String cad= br.readLine();
//Conversión de la variable cad a el tipo de dato que quiero
int num1=Integer.parseInt(s: cad);
System.out.println(x: num1);
```

ENUM

```
public class Ejemplo_Enumerados {
    public enum nivel{
        BAJO, MEDIO, ALTO;
    }

    public static void main(String[] args) {
        nivel miNivel=nivel.MEDIO;
        System.out.println("Mi nivel es "+miNivel);
    }
}
```

TEMA2

Constructor con parámetros

```
public Alimento(String nombre, int grasas, int hidratos, boolean origenAnimal) {  
    this.nombre = nombre;  
    this.grasas = grasas;  
    this.hidratos = hidratos;  
    this.origenAnimal = origenAnimal;  
}
```

Constructor sin parámetros o por defecto

```
//Constructor por defecto  
public Circulo() {  
  
}
```

Constructor copia

```
//Creo un constructor copia  
public Taller(Taller t){  
    this.aceite=t.aceite;  
    this.ruedas=t.ruedas;  
    this.contCambiosParciales=t.contCambiosParciales;  
    this.contCambiosTotales=t.contCambiosTotales;  
}
```

Crear objetos

```
Alimento alimento = new Alimento( nombre: "Muslo de cerdo", grasas:10, hidratos:3, origenAnimal:true);
```

Palabra reservada this

Se utiliza para resolver ambigüedades con atributos con el mismo identificador

```
public Alimento(String nombre, int grasas, int hidratos, boolean origenAnimal) {  
    this.nombre = nombre;  
    this.grasas = grasas;  
    this.hidratos = hidratos;  
    this.origenAnimal = origenAnimal;  
}
```

Métodos set y get

Get-> Obtener dato

Set->Asignar dato

```
public int getGrasas() {  
    return grasas;  
}  
  
public void setLargo(int largo) {  
    this.largo = largo;  
}
```

Operador .

Para acceder a los métodos de una clase

```
alimento.mostrar();
```

Definir atributos

```
private int largo;  
private int ancho;
```

Método estático

No hace falta crear un objeto, simplemente llamamos a la clase y llamamos al método

```
public static void nombreClase(){  
    System.out.println("Soy la clase Persona");  
}  
public class Ejemplo{  
    public static void main(String args[]){  
        Persona.nombreClase();  
    }  
}
```

STRINGS

Extracción de un carácter individual

```
if (frase.charAt(index:i) == 'a' || frase.charAt(index:i) == 'e' || frase.charAt(index:i) == 'i'  
    frase_convertida+="o";  
else{  
    frase_convertida+=frase.charAt(index:i);  
}
```

Obtener Longitud de una cadena

```
for (int i = 0; i < frase.length(); i++) {
```

Descomponer una cadena

```
frase_mitad=frase.substring(beginIndex: 0, frase.length()/2);
```

Igualdad de cadenas

Con equals, distingue entre mayúsculas y minúsculas

```
System.out.println("¿Las cadenas son iguales?: "+texto1.equals(texto2));
```

Con equalsIgnoreCase, no diferencia entre mayúsculas y minúsculas

```
System.out.println("¿Las cadenas son iguales?: "+texto1.equalsIgnoreCase(texto2));
```

Comparar cadenas con compareTo, devuelve un entero >0,<0 o =0

```
String texto1="Buenos días";  
String texto2=new String("Buenos dias");  
if (texto1.compareTo(texto2)>0)  
    System.out.println("texto1 es superior a texto2");  
else if(texto1.compareTo(texto2)<0)  
    System.out.println("texto1 es inferior a texto2");  
else  
    System.out.println("texto1 y texto2 son iguales");
```

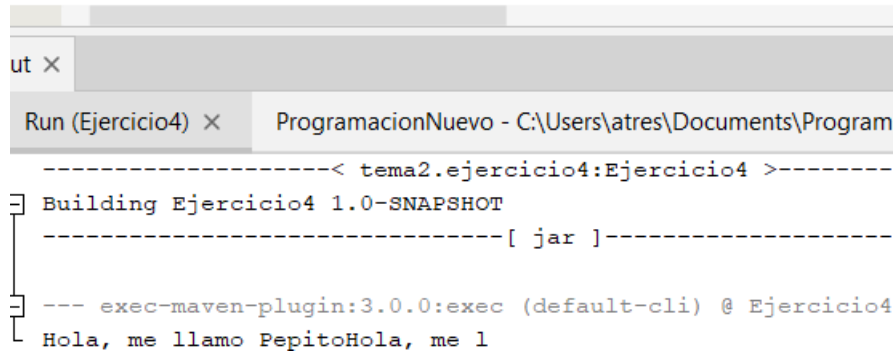
Otras funciones

Comprobar si una cadena empieza o finaliza con un subcadena determinada

```
System.out.println( x: frase.startsWith( prefix: "Ho" ) );
```

Eliminar los espacios en blanco de una cadena que tenga por delante y detrás

```
System.out.println(frase.trim()+frase_mitad);
```



```
-----< tema2.ejercicio4:Ejercicio4 >-----
Building Ejercicio4 1.0-SNAPSHOT
-----[ jar ]-----
--- exec-maven-plugin:3.0.0:exec (default-cli) @ Ejercicio4
Hola, me llamo PepitoHola, me l
```

Cambiar toda la cadena a mayúsculas o minúsculas

```
System.out.println( x: frase.toUpperCase() );
```

Buscar una cadena dentro de otra, si lo encuentra te devuelve la posición, si no, -1
Siempre la primera ocurrencia que encuentre en la cadena

```
System.out.println( x: frase.indexOf( str: "o" ) );
```

También se puede buscar desde una posición inicial

```
System.out.println( x: frase.indexOf( str: "Hola", fromIndex: 5 ) );
```

Reemplazar una cadena por otra

```
frase.replace("es", "no por");
```

Devolver la posición de la ultima ocurrencia de la cadena dada como parámetro

```
System.out.println( x: frase.lastIndexOf( str: "o" ) );
```

TEMA3

IF

```
if (numero >= 1 && numero <= 7) {  
    valido = true;  
} else {  
    System.out.println(x: "Error, has introducido un número fuera del intervalo(1-7)");  
}
```

IF Anidado

```
if (numero > 0) {  
    System.out.println(x: "Es positivo");  
} else if (numero < 0) {  
    System.out.println(x: "Es negativo");  
} else {  
    System.out.println(x: "Es nulo");  
}
```

SWITCH

```
public static String diaSemana(int num) {  
    String diaSemana = " ";  
    switch (num) {  
        case 1 ->  
            diaSemana = "Lunes";  
        case 2 ->  
            diaSemana = "Martes";  
        case 3 ->  
            diaSemana = "Miercoles";  
        case 4 ->  
            diaSemana = "Jueves";  
        case 5 ->  
            diaSemana = "Viernes";  
        case 6 ->  
            diaSemana = "Sábado";  
        case 7 ->  
            diaSemana = "Domingo";  
    }  
    return diaSemana;  
}
```

WHILE


```

while(i <=numero){

    if(i%2!=0){
        int j=9;
        while(j!=0){
            System.out.printf( format: "%d ", args: j);
            j--;
        }
        System.out.println();
    }
    else{
        int j=1;
        while(j!=10){
            System.out.printf( format: "%d ", args: j);
            j++;
        }
        System.out.println();
    }

    i++;
} //while

```

DO WHILE

```

do {
    System.out.println( x: "Introduce un número entre 1 y 7: ");
    numero = teclado.nextInt();
    if (numero >= 1 && numero <= 7) {
        valido = true;
    } else {
        System.out.println( x: "Error, has introducido un número fuera del intervalo(1-7)");
    }
} while (!valido);
return numero;

```

FOR

```

for(int lineas=1; lineas<=numFilas; lineas++){

    System.out.println( x: "*");
}

```

BUCLE ANIDADOS

```

for(int lineas=1; lineas<=numFilas; lineas++){

    //asteriscos
    for(int asteriscos=1; asteriscos<=(2*lineas)-1; asteriscos++){

        System.out.print( s: "*");
    }
    System.out.println();
}

```

TEMA 4

FECHAS

DATETIMEFORMATTER

```
DateTimeFormatter f = DateTimeFormatter.ofPattern(pattern: "dd-MM-yyyy");  
this.fechaVencimiento = LocalDate.parse(text: new Scanner(source: System.in).nextLine(), formatter: f);
```

IS BEFORE

```
return fechaVencimiento.isBefore(other: LocalDate.now());
```

PLUS DAYS

```
this.fechaVencimiento.plusDays(daysToAdd: dias);
```

PERIOD

```
int diasFaltan = Period.between(startDateInclusive: fechaVencimiento, endDateExclusive: LocalDate.now()).getDays();
```

NEW LOCALE

```
DateTimeFormatter f=DateTimeFormatter.ofPattern(pattern: "dd-MMMM-yyyy").withLocale(new Locale(language: "es", country: "ES"));
```

ARRAYS

Asignacion

```
this.libros = new Libro[nLibros];
```

Recorrer vector

```
for(int i=0;i<numeros.length;i++){  
    numeros[i]=(int) (Math.floor(Math.random()*10)+1);  
    System.out.print(numeros[i]+" ");  
}
```

ARRYS MULTIDIMENSIONALES

```
int[][] array = new int[10][10];
```

RECORRER ARRAYS MULTIDIMENSIONALES

```
for (int i = 0; i < array.length; i++) {  
    for (int b = 0; b < array[i].length; b++) {
```

ARRAYS DE OBJETOS

```
private Ciudad[] ciudades;
```

CLASES DE ARRAYS

Arrays fill

```
Arrays.fill(a: notas, val: 0);
```

Array sort(ordena de menor a mayor)

```
Arrays.sort(a: notas);
```

TEMA5

CASTING DE CLASES

```
boolean encontrado=false;
for (Empleado p : empleados) {
    if (p != null) {
        if (p instanceof Programador programaux) {
            if (programaux.clasificacion().equalsIgnoreCase("Intermedio")) {
```

Excepciones

FileNotFoundException: archivo no encontrado

ClassNotFoundException: no existe la clase

ArrayIndexOutOfBoundsException: posición del array inexistente

NullPointerException: null;

IOException: generaliza a muchas excepciones

InputMismatchException: formato

RuntimeException.

Excepción personalizada

```
public class EmpleadoNOEncontradoException extends Exception {

    public EmpleadoNOEncontradoException(String mensaje1) {
        super(mensaje1);
    }
}
```

HASH CODE

```
@Override
public int hashCode() {
    int hash = 7;
    hash = 53 * hash + Objects.hashCode(this.nombre);
    hash = 53 * hash + Objects.hashCode(this.ciudad);
    hash = 53 * hash + this.nacimiento;
    return hash;
}
```

Equals

```

@Override
public boolean equals(Object obj) {
    if (this == obj) {
        return true;
    }
    if (obj == null) {
        return false;
    }
    if (getClass() != obj.getClass()) {
        return false;
    }
    final Pintor other = (Pintor) obj;
    if (this.nacimiento != other.nacimiento) {
        return false;
    }
    if (!Objects.equals(this.nombre, other.nombre)) {
        return false;
    }
    return Objects.equals(this.ciudad, other.ciudad);
}

```

Compare

```

@Override
public int compareTo(Pintor o) {
    int comparacion = Integer.compare(o.nacimiento, this.nacimiento);
    if (comparacion != 0) {
        return comparacion;
    } else {
        return this.nombre.compareTo(o.nombre);
    }
}

```

el metodo compareTo debe seguir la regla de que si a.compareTo(b) devuelve un número negativo

a debería estar antes de b en la secuencia

si devuelve cero, a y b son iguales

si devuelve un número positivo, a debería estar después de b en la ordenacion

RANDOM

```

int x= (int) Math.floor(Math.random()*N);
// x será un número entre 0 y N-1
int x= (int) Math.floor(Math.random()*N)+1;
// x será un número entre 1 y N
int x= (int) Math.floor(Math.random()*(M-N+1))+N;
//x será un número entre M y N ambos incluidos y siendo M mayor que N

```

```
Random r = new Random();  
int y= r.nextInt(N);  
// y será un número entre 0 y N-1  
int y= r.nextInt(N)+1;  
// y será un número entre 1 y N  
int y= rand.nextInt(M-N+1)+N;  
//y será un número entre M y N ambos incluidos y siendo M mayor que N
```