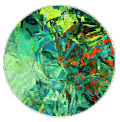


C.F.G.S. Desarrollo de Aplicaciones Web/Multiplataforma

MÓDULO: Programación

IES CAMAS Curso 2025/26

RA1. Reconoce la estructura de un programa informático, identificando y relacionando los elementos propios del lenguaje de programación utilizado.
Criterios de evaluación:
a) Se han identificado los bloques que componen la estructura de un programa informático.
b) Se han creado proyectos de desarrollo de aplicaciones.
c) Se han utilizado entornos integrados de desarrollo.
d) Se han identificado los distintos tipos de variables y la utilidad específica de cada uno.
e) Se ha modificado el código de un programa para crear y utilizar variables.
f) Se han creado y utilizado constantes y literales.
g) Se han clasificado, reconocido y utilizado en expresiones los operadores del lenguaje.
h) Se ha comprobado el funcionamiento de las conversiones de tipo explícitas e implícitas.
i) Se han introducido comentarios en el código.

(2,5 puntos) Ejercicio 1: Un economista te ha encargado un programa para realizar cálculo con el IVA. La aplicación debe solicitar la base imponible y el IVA que se debe aplicar. Muestra en pantalla el importe correspondiente al IVA y al total.

Nota: IVA = 21.0 ; Importe IVA es igual a base imponible por IVA / 100; Total es igual a base imponible más importe IVA

```java
package examen.ra1;
import java.util.Scanner;
public class Ejercicio1 {
    public static void main(String[] args) {
        // 1) Entrada/Salida con Scanner (bloque de E/S)
        Scanner sc = new Scanner(System.in);

        // 2) Constantes y variables
        //    - CONSTANTE: valor que no cambia. Usamos mayúsculas por convención
        final double IVA = 0.21;
        //    - VARIABLES: guardan datos que sí pueden cambiar.
        double baseImponible, importeIva, total;

        System.out.print("Introduce la base imponible (€): ");
        baseImponible = sc.nextDouble(); // tipo primitivo double

        // 3) Calculo
        importeIva = baseImponible * IVA;
        total  = baseImponible + importeIva;

        // 4) Salida con formato
        System.out.println("Base imponible: " + baseImponible + " €");
        System.out.println("TOTAL: " + total + " €");

        sc.close(); // buena práctica
    }
}
```

(2,5 puntos) Ejercicio 2: Sin usar condicionales. Pide dos números al usuario: a y b. Deberá mostrarse **`true`** si ambos números son iguales y **`false`** en caso contrario.

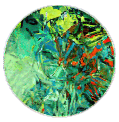
```java
package examen.ra1;
import java.util.Scanner;
public class Ejercicio2 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        // Variables. (también podría ser double; aquí usamos int para claridad)
        int a, b;
        boolean sonIguales;

        // Leemos dos enteros
        System.out.print("Introduce a: ");
        a = sc.nextInt();
        System.out.print("Introduce b: ");
        b = sc.nextInt();

        // Sin condicionales: el operador == devuelve un boolean directamente
        sonIguales = (a == b); // true si a y b tienen el mismo valor

        // Mostramos el resultado booleano tal cual pide el enunciado
        System.out.println(sonIguales);

        sc.close();
    }
}
```

(5 puntos) Ejercicio 3: Selecciona las respuestas correctas

1.1. ¿Cuál de los siguientes identificadores **no** puede emplearse para una variable? a) lenguaje b) $firma **c) int** **d) final**	**1.2.** De todos los tipos primitivos disponibles en Java, selecciona cuál o cuáles son los que tienen un mayor tamaño y, por lo tanto, pueden albergar un mayor número de valores: a) long **b) long y double** c) long y float d) En Java todos los tipos primitivos tienen el mismo tamaño
1.3. Mediante qué símbolos es posible añadir un comentario en nuestro código: **a) //** **b) /* */** c) ++ d) Cualquiera de los anteriores	**1.4.** ¿Qué puede usarse de importe automáticamente en cualquier programa sin necesidad de tener que utilizar una sentencia `import`? a) java.util **b) java.lang** c) java.swing d) javax.sql

1.5. ¿Cuáles de las siguientes instrucciones nos permiten mostrar información por consola?
a) System()
b) System.out()
c) System.out.print()
d) System.out.println()
	1.6. ¿Qué instrucción es equivalente a: `i = i + 1`?
	a) i += 1;
	b) ++i;
	c) i++;
	d) Cualquiera de los anteriores

1.7. Si evalúas la siguiente expresión: `z < 1 || z++ == 1 && z == 1` el resultado de dicha expresión es: a) true **b) false (Nota: Para z→2)** c) 0 d) Ninguna de las anteriores	**1.8.** ¿Qué valor toma la variable a, tras la ejecución de la instrucción: `int a = 1 < 2 ? 3 : 4;` a) 1 b) 2 **c) 3** d) 4
1.9. Selecciona la expresión cuya evaluación resulta 3: a) 3 * 2 / 5 b) 3 + 2 * 6 / 5 c) (3 + 2) * 6 / 5 **d) (3 + 2 * 6) / 5**	**1.10.** En las siguientes conversiones de tipo, ¿cuál de ellas produce un error? a) `int a = (int) 1.23;` **b) `int a = 12.3;`** c) `double a = (double) 123;` d) `double a = 123;`



En varias preguntas hay **más de una correcta**. Se indican la razón:

1.1. Identificadores no válidos
✅ **c) int** y **d) final** (palabras reservadas; no pueden ser nombres de variables).
1.2. Tipos primitivos con mayor tamaño
✅ **b) long y double**
`long` (entero 64 bits) y `double` (coma flotante 64 bits).

1.3. Símbolos para comentarios
✅ **a)** *//* y *b) //*

1.4. Paquete importado automáticamente
✅ **b) java.lang** Se importa por defecto (String, Math, System…).

1.5. Mostrar por consola
✅ **c) System.out.print()** y **d) System.out.println()**
`System()` y `System.out()` solos no imprimen.

1.6. Equivalente a `i = i + 1;`
✅ **d) Cualquiera de los anteriores**
`i += 1;`, `++i;` e `i++;` incrementan 1. (Ojo: el valor **devuelto** por `++i`/`i++` difiere, pero como sentencia aislada, todas incrementan en 1).

1.7. Expresión: `z < 1 || z++ == 1 && z == 1`
👉 Si **z = 2**, resulta **false**.
z < 1 || z++ == 1 && z == 1

2 < 1 || 3 == 1 && 2 == 1

false || false && false

1.8. Ternaria
`1 < 2 ? 3 : 4` → condición verdadera → **3**
✅ **c) 3**

1.9. Expresión que vale 3 (enteros)
✅ **d) (3 + 2 * 6) / 5**

1.10. Conversión que da error
a) `int a = (int) 1.23;` ✅ (cast explícito)
b) `int a = 12.3;` ❌ (falta cast)
c) `double a = (double) 123;` ✅ (innecesario pero válido)
d) `double a = 123;` ✅ (ampliación implícita)
✅ **b) int a = 12.3;**