Módulo 4. POO: Subrutinas, subprogramas: con y sin pasaje de parámetros

CÓRDOBA CREA FUTURO

1. Subrutinas o subprogramas

En POO en Java, los «subprogramas» se refieren a los métodos, que son bloques de código que realizan una tarea específica dentro de una clase y pueden ser llamados desde otros lugares del programa. Estos métodos son fundamentales para la encapsulación, la reutilización de código y el polimorfismo en Java.

Aunque el término «subprograma» también puede referirse a funciones y procedimientos, en Java, la unidad básica de comportamiento es el método de una clase.

¿Qué son los subprogramas en Java?

El concepto de programación «divide y vencerás» es una estrategia algorítmica que consiste en resolver un problema complejo dividiéndolo en subproblemas más pequeños y manejables, resolviendo cada subproblema de forma recursiva y, finalmente, combinando las soluciones para obtener la solución final. Este enfoque se basa en tres fases:

dividir, conquistar (resolver los subproblemas) y combinar las soluciones.

El principio de Rubik sostiene que los problemas complejos pueden resolverse con mayor facilidad si se dividen en partes cada vez más pequeñas y, por lo tanto, más simples. Esto implica que cada incidencia o tarea debe asumir una única responsabilidad, lo que facilita su explicación. En esencia, la solución consiste en descomponer el problema general en pequeños fragmentos manejables.

Los subprogramas son un conjunto de instrucciones que realizan una labor específica y se comportan de manera independiente en un programa.

Los subprogramas facilitan lo siguiente.

- Descomponer la complejidad del problema.
- Reutilización del código.
- Mayor legibilidad.

- Facilitar para añadir nueva funcionalidad.

Los subprogramas pueden ser invocados varias veces desde diferentes partes del programa.

El programa principal y los subprogramas se comunican mediante el paso de parámetros o argumentos. A nivel algorítmico, los subprogramas se clasifican en funciones y procedimientos.

Funciones

Un procedimiento es un subprograma que puede recibir «n» parámetros y devuelve o regresa cero, uno o más valores.

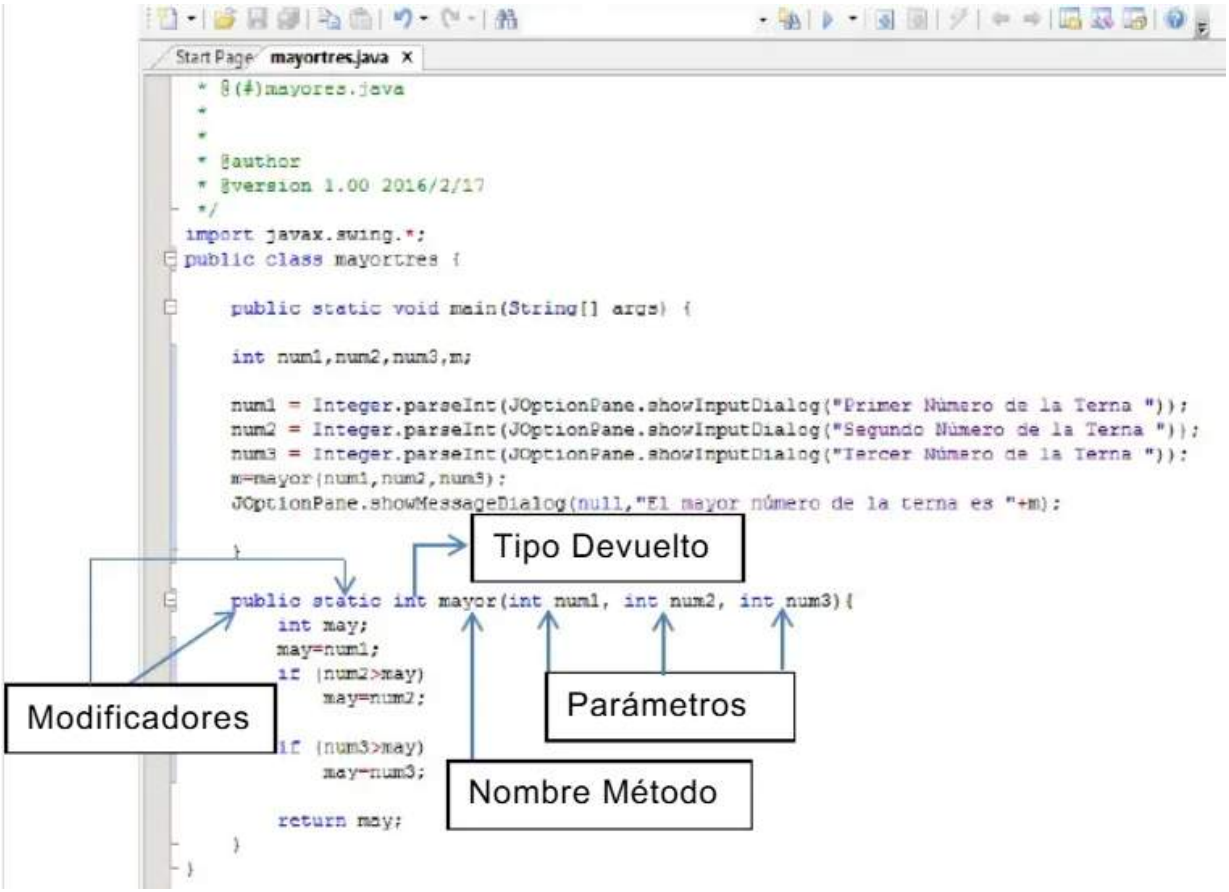
Procedimientos

Un procedimiento es un subprograma que pueden recibir «n» parámetros y devuelve o regresa, cero o más valores.

En el lenguaje Java los subprogramas/funciones o procedimientos) se implementan por medio de los métodos.

En la siguiente imagen se puede observar un método definido como subprograma.

Figura 1: Parámetros de un subprograma



Fuente: elaboración propia.

A continuación, explicaremos los recuadros que aparecen en la figura 1.

Modificadores: está indicando que es público y puede ser invocado sin mayores problemas.

- **Tipo devuelto**: indica que si la innovación al método, en este caso, es «mayor», este devolverá un valor entero (int). En el ejemplo de la figura 1, se lleva a cabo cuando se ejecuta la última línea: «*return may*», donde el atributo «*may*» está declarado como entero (int).

- **Nombre método**: nombre con el que será invocado.

- **Parámetros**: cada uno de ellos recibe el valor con el que es invocado en el método, es decir, desde alguna parte del programa debió ser llamado, por ejemplo, del siguiente modo: «mayor(1,3,5)».

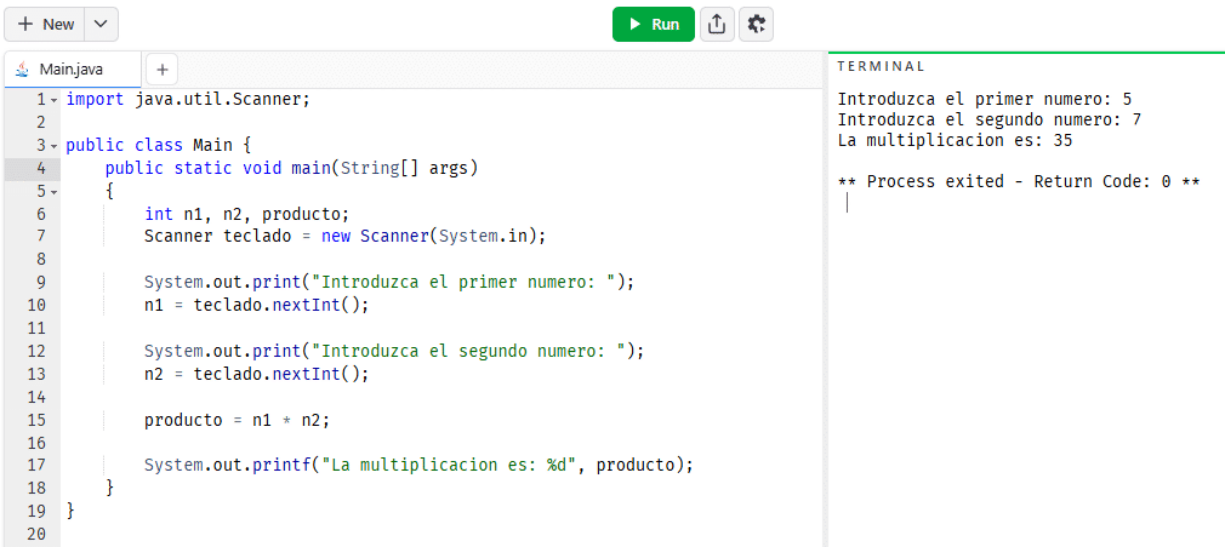
CONTINUAR

2. Funciones

Sin invocar subprogramas

Se ingresan por teclado dos números enteros, luego se multiplican y, finalmente, se imprime el resultado. El alumno puede ejecutar el código de la figura 2 haciendo clic en el siguiente enlace: https://www.online-java.com/9TBbcwrwFB

Figura 2: Sin invocar subprogramas




```java
import java.util.Scanner;

public class Main {
    public static void main(String[] args)
    {
        int n1, n2, producto;
        Scanner teclado = new Scanner(System.in);

        System.out.print("Introduzca el primer numero: ");
        n1 = teclado.nextInt();

        System.out.print("Introduzca el segundo numero: ");
        n2 = teclado.nextInt();

        producto = n1 * n2;

        System.out.printf("La multiplicacion es: %d", producto);
    }
}
```

Fuente: elaboración propia.

Funciones con subprograma público y con parámetros

No existe obligatoriedad de usar parámetros, es decir, puede haber un método que no lo tenga.

Vemos el siguiente ejemplo que el alumno podrá ejecutar haciendo clic en el siguiente enlace: https://www.online-java.com/OgqQQHMBij

Figura 3: Funciones con subprograma público y con parámetros

Fuente: elaboración propia.

Explicación del código

public static int sumar(int num1, int num2): esta es la definición del subprograma o método.

- *public static*: modificadores de acceso que permiten que el método sea llamado desde cualquier otra clase sin necesidad de crear un objeto.
- int: el tipo de dato que el método devolverá al finalizar su ejecución (en este caso, un número entero).
- sumar: el nombre del método.
- (int num1, int num2): los parámetros que el método recibe para trabajar.

int resultado = num1 + num2: dentro del método, se realiza la operación de suma.

***return* resultado**: el resultado de la operación se devuelve al punto donde se llamó el método.

***public static void main*(String[] args):** este es el punto de entrada principal de la aplicación Java. Aquí es donde se llama a nuestro subprograma.

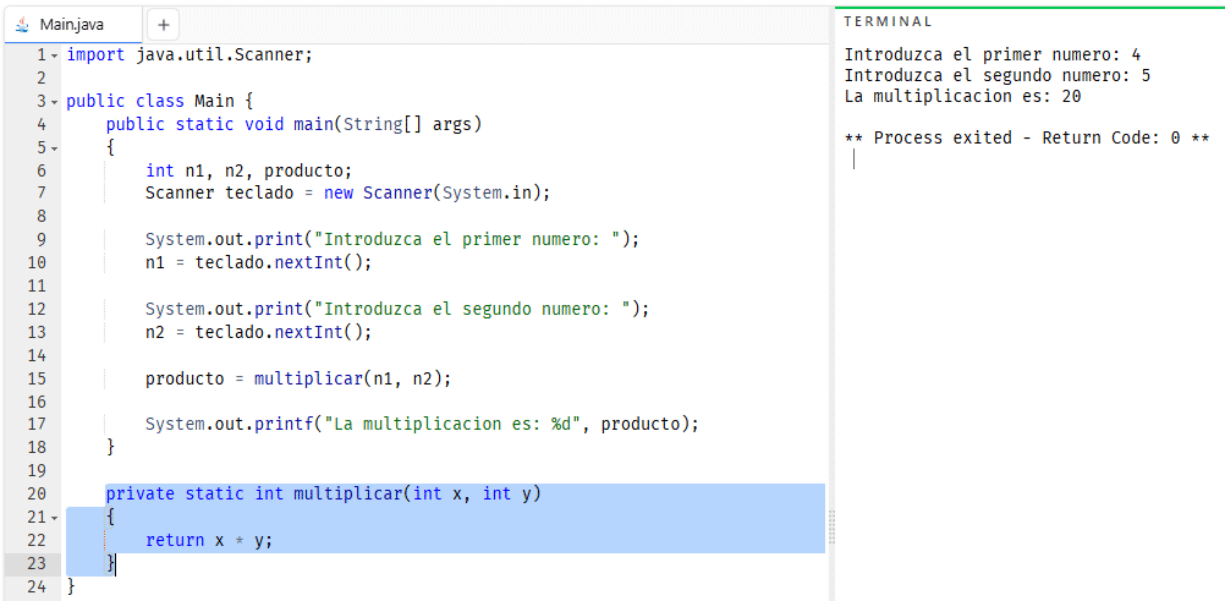
int sumaTotal = sumar(a, b): esta línea ejecuta (llama) al subprograma sumar pasando los valores de a y b. El valor devuelto por el subprograma se almacena en la variable sumaTotal.

System.out.println(...): imprime el resultado en la consola.

Funciones con subprograma privado, estático y con parámetros

No obstante, si se quisiera considerar la realización de la multiplicación como una función, teniendo en cuenta que en Java las funciones se implementan como métodos, la solución podría ser la siguiente (el alumno puede ejecutar el código fuente que se aporta haciendo clic en el siguiente enlace: https://www.online-java.com/4J1TK6OF6U).

Figura 4: Funciones con subprograma privado, estático y con parámetros



Fuente: elaboración propia.

Véase que, se ha llamado al método multiplicar, pasándole los parámetros actuales n1 y n2 (línea 15 de la figura 4):

producto = multiplicar(n1, n2).

Al hacer la llamada a multiplicar, los parámetros formales x e y (línea 20 de la figura 4) reciben los valores (datos) de n1 y n2. Debemos prestar atención a que son del mismo tipo entero (int, int), haciéndose una copia de dichos datos. Por lo

que, aunque este no sea el caso, si los valores de **x** e **y** fuesen modificados dentro de multiplicar, los valores de n1 y n2 no se verían afectados.

> ***private static int multiplicar*(int x, int y)** (línea 20 de la figura 4)
>
> **{**
>
> ***return* x * y;** (retorna la multiplicación de dos variables enteras: línea 22 de la figura 4)
>
> **}**

Respecto al método **multiplicar**, observa que se ha declarado lo siguiente.

De tipo *integer* (int): debido a que, devuelve el resultado de multiplicar x * y, que es un número entero. Para indicar esto, se ha utilizado la palabra reservada *return*.

CONTINUAR

3. Procedimientos en Java

Menú de opciones: sin subprograma

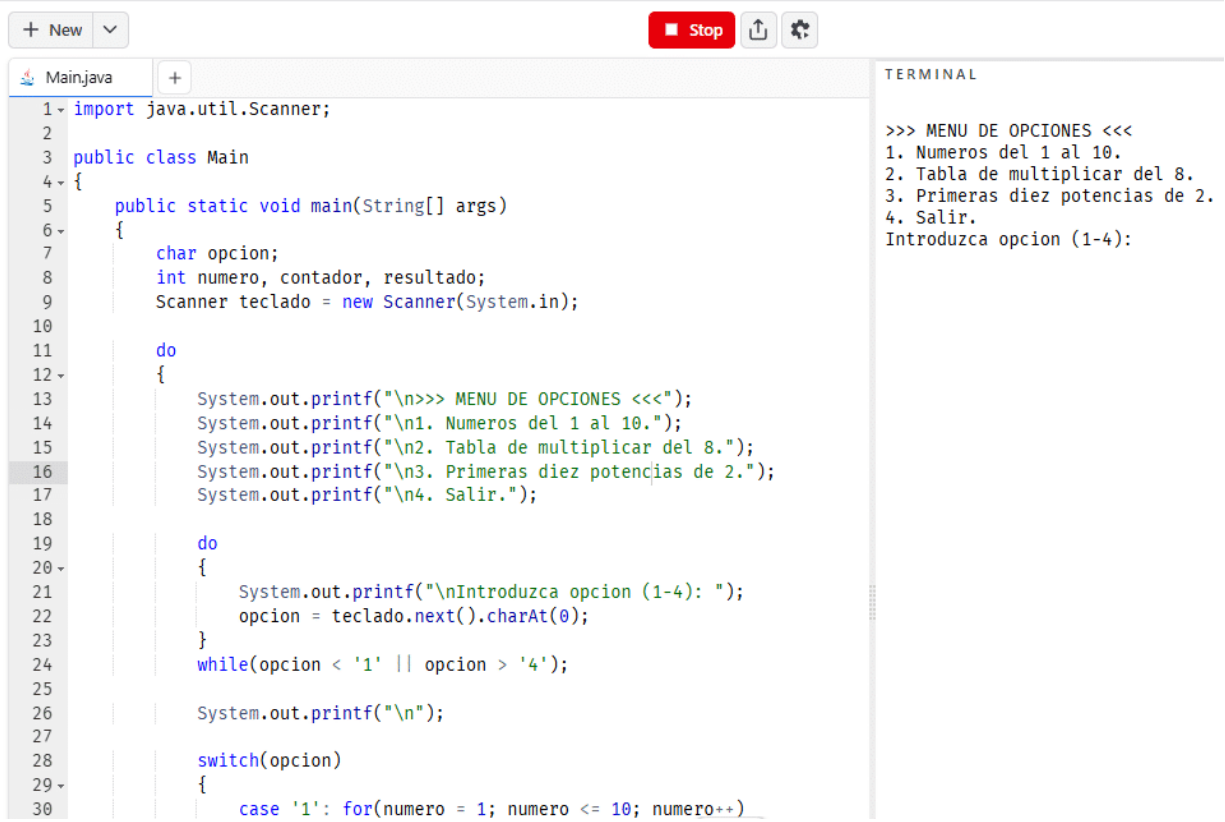
Ejemplo {MenuDeOpciones}. Se quiere escribir un programa que muestre en la pantalla un menú con 4 opciones. Para ello, debemos realizar lo siguiente.

- Visualizar los números enteros del 1 al 10.
- Visualizar la tabla de multiplicar del número 8.
- Visualizar las primeras diez potencias del número 2 (del 2^1 al 2^{10}).
- Salir.

Seguidamente, el programa debe solicitar por teclado al usuario que introduzca la opción deseada y ejecutarla. De tal forma que, por pantalla, se podrá ver algo similar a lo que muestra la figura 5.

Menú de opciones sin subprograma: https://www.online-java.com/YRGOrgdWrO

Figura 5: Menú de opciones sin subprograma



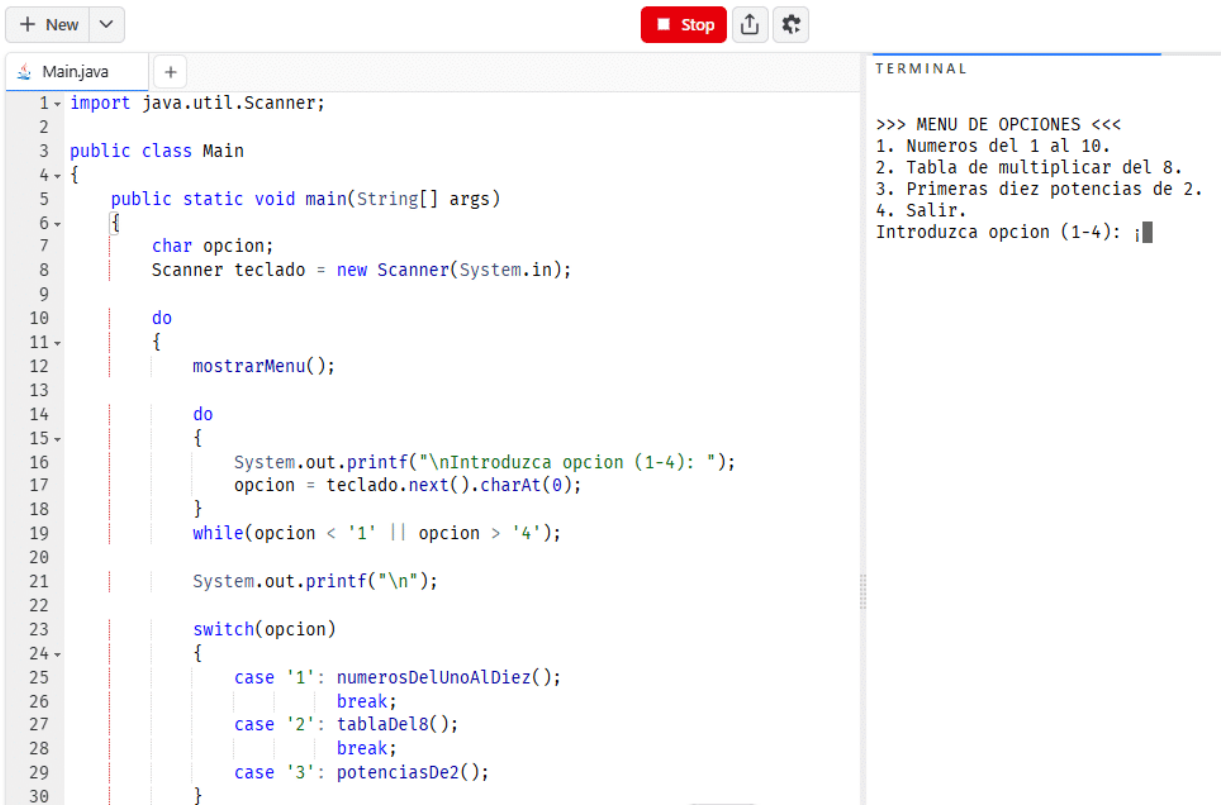

```
1  import java.util.Scanner;
2
3  public class Main
4  {
5      public static void main(String[] args)
6      {
7          char opcion;
8          int numero, contador, resultado;
9          Scanner teclado = new Scanner(System.in);
10
11         do
12         {
13             System.out.printf("\n>>> MENU DE OPCIONES <<<");
14             System.out.printf("\n1. Numeros del 1 al 10.");
15             System.out.printf("\n2. Tabla de multiplicar del 8.");
16             System.out.printf("\n3. Primeras diez potencias de 2.");
17             System.out.printf("\n4. Salir.");
18
19             do
20             {
21                 System.out.printf("\nIntroduzca opcion (1-4): ");
22                 opcion = teclado.next().charAt(0);
23             }
24             while(opcion < '1' || opcion > '4');
25
26             System.out.printf("\n");
27
28             switch(opcion)
29             {
30                 case '1': for(numero = 1; numero <= 10; numero++)
```

Fuente: elaboración propia.

Menú de opciones: con subprograma

No obstante, considerando cada una de las opciones del menú como un subproblema a resolver utilizando un procedimiento y, por otra parte, teniendo en cuenta que en Java los procedimientos se implementan como métodos, el código fuente podría ser como se muestra en la figura 6 o como se ve al hacer clic en el siguiente enlace: https://www.online-java.com/L20VKb53ah

Figura 6: Menú de opciones con subprograma




```java
import java.util.Scanner;

public class Main
{
    public static void main(String[] args)
    {
        char opcion;
        Scanner teclado = new Scanner(System.in);

        do
        {
            mostrarMenu();

            do
            {
                System.out.printf("\nIntroduzca opcion (1-4): ");
                opcion = teclado.next().charAt(0);
            }
            while(opcion < '1' || opcion > '4');

            System.out.printf("\n");

            switch(opcion)
            {
                case '1': numerosDelUnoAlDiez();
                          break;
                case '2': tablaDel8();
                          break;
                case '3': potenciasDe2();
            }
```

Fuente: elaboración propia.

Observa que, además de los métodos numerosDelUnoAlDiez (línea 25 invoca y línea 44 se implementa el subprograma, como vemos en la figura 6), tablaDel8 (línea 27 invoca y línea 53 se implementa el subprograma, como vemos en la figura 6), y potenciasDe2 (línea 29 invoca y línea 64 se implementa el subprograma, como vemos en la figura 6), también se ha creado mostrarMenu (línea 12, como vemos en la figura 6) para mostrar el menú de opciones. Por otra parte, podemos apreciar lo siguiente.

- Todos ellos son de tipo ***void***, ya que no devuelven ningún valor.
- A ninguno se les pasa parámetros, aunque sí podrían recibirlos.

CONTINUAR