



Carátula para entrega de prácticas

Facultad de Ingeniería

Laboratorios de docencia

Laboratorio de Computación Salas A y B

Profesor(a): René Adrián Dávila Pérez

Asignatura: Programación Orientada a Objetos

Grupo: 1

No de Práctica(s): 1, 2

Integrante(s): 322184022

322184022

321570789

322069093

322249723

*No. de lista o
brigada:* 5

Semestre: 2026-1

Fecha de entrega: 29/08/2025

Observaciones:

CALIFICACIÓN: _____

Índice

1. Introducción	2
2. Marco Teórico	2
3. Desarrollo	3
3.1. Menú	3
3.2. Factorial	3
3.3. Fibonacci	3
3.4. Collatz	3
4. Resultados	4
5. Conclusión	7
6. Referencias	7

1 Introducción

En esta práctica se exploró el trabajo de la recursividad a través de la implementación de tres algoritmos matemáticos: El factorial, la secuencia de Fibonacci y Collatz. La práctica consistió en elaborar un código que no solo resolviera cada problema eficientemente utilizando funciones recursivas en ciertos casos, sino también, mostrar una interfaz de usuario interactiva, para que quien lo usa pueda seleccionar y ejecutar cada una de las funciones de forma independiente.

Objetivos

- Implementar funciones recursivas
- Aplicar los conceptos matemáticos factorial, Fibonacci y Collatz
- Comprender la diferencia entre el uso de estructuras recursivas e iterativas para resolver un mismo problema.

2 Marco Teórico

Previo a la práctica, fue necesario conocer y entender conceptos como algoritmo: es un conjunto de pasos (finitos y no ambiguos) ordenados que nos permiten resolver un problema. Utilizamos estructuras de control, que permiten decidir o repetir ciertas instrucciones, por ejemplo, los condicionales (if-else) y los ciclos (do-while). Por otra parte, implementamos funciones/métodos: bloques de código diseñados para cumplir instrucciones específicas.

- Método recursivo
Sucede cuando una función se llama a si misma para resolver un problema en partes más pequeñas. Debe existir un caso base, que es la condición donde la función deja de llamarse a si misma y termina la ejecución (para no tener un caso infinito).
- Factorial
Es el producto de un número por todos los enteros positivos menores que él.
- Serie de Fibonacci
Es una sucesión de números donde cada número es la suma de los dos anteriores, empezando por 1 y 1.
- Collatz
Secuencia definida de la siguiente forma: si el número es par se divide entre 2 y si el número es impar se multiplica por 3 y se le suma 1, se repite el proceso hasta llegar al número 1. [1]

3 Desarrollo

3.1 Menú

Primero se importó el “Scanner” para poder solicitar datos de entrada al usuario. Después se desarrolló un menú con un ciclo “do-while” el cual se ejecuta siempre y cuando el usuario no introduzca el número 4. Dentro del ciclo “do-while” se encuentra un “switch case” el cual cuenta con 4 opciones y un caso “default”. El caso uno contiene el Factorial, el caso dos la secuencia de Fibonacci, el caso tres Collatz, el cuatro la opción de salir y el “default” por si el usuario introduce una opción inexistente.

3.2 Factorial

Para el Factorial, primero se define la función y el número entero que va a recibir. Se trabaja con long ya que los números factoriales pueden ser muy grandes. Se crea una condición para los casos de 0 y 1, si se cumple se devuelve 1, en caso contrario se devuelve $n * \text{factorial}(n-1)$. En el switch se le pide al usuario que introduzca un número entero y se vuelve a crear otra condición, si el usuario introduce un número negativo, menor a cero, el sistema le regresa un mensaje advirtiéndole que no es posible. En caso contrario se llama a la función y se imprime el resultado. Luego finaliza el case con "break".

3.3 Fibonacci

Se define la función la cual también regresará un número long y se le pasa el dato de entrada. Se definen dos condiciones, si el número dado es 0 devuelve 0 y si es 1 devuelve 1. En caso contrario, se calcula recursivamente utilizando la fórmula de Fibonacci. En el switch primero el usuario ingresa el número hasta el cual se desea calcular la secuencia, después se evalúa con una condición en caso de haber introducido un numero menor a cero y se le advierte al usuario que los números ingresados deben ser mayores a 0 ya que la secuencia no está definida para números negativos. Si el número cumple entonces se utiliza un ciclo “for” para imprimir la secuencia desde el número 0 hasta el solicitado. Luego finaliza el case con "break".

3.4 Collatz

Primero se define la función, en este caso se utiliza “void” ya que no será una función recursiva, será una función que no devuelve nada. Después se imprime el número actual y se definen tres condiciones, la primera es el caso base y dice que si el número dado por el usuario es igual a 1 se termina la función, para estos se utiliza return, evitando que se ejecute infinitamente, la segunda condición comprueba si es par, dice que si el número dado es divisible entre dos, entonces, el número será dividido entre dos, y en caso de ser impar, el número será multiplicado por tres y sumado uno. En el switch se le solicita al usuario que introduzca un número, se llama la función y se imprime en pantalla el resultado. Luego finaliza el case con "break".

4 Resultados

```
1 import java.util.Scanner;
2
3 public class Menu {
4
5     Run | Debug
6     public static void main(String[] args) {
7         Scanner scanner = new Scanner(System.in);
8         int opcion;
9
10        do {
11            // Mostrar menú
12            System.out.println("\nMENÚ:");
13            System.out.println("1. Factorial");
14            System.out.println("2. Fibonacci");
15            System.out.println("3. Secuencia de Collatz");
16            System.out.println("4. Salir");
17            System.out.print("Seleccione una opción (1-4): ");
18
19            opcion = scanner.nextInt();
20
21            switch (opcion) {
22                case 1:
23
24                case 2:
25
26                case 3:
27
28                case 4:
29
30            default:
31                System.out.println("Opción no válida. Por favor, seleccione una opción del 1 al 4.");
32            }
33        } while (opcion != 4);
34
35        scanner.close();
36    }
37
38
39
40 }
```

Figura 1: Importación del "Scanner", estructura del código y menú

```
MENÚ:
1. Factorial
2. Fibonacci
3. Secuencia de Collatz
4. Salir
Seleccione una opción (1-4): 1
```

1.1. Menú en la terminal

```

66  public static long factorial(int n) {
67      if (n == 0 || n == 1) { // Casos base
68          return 1;
69      }
70      return n * factorial(n - 1); // aplica recursividad con n-1
71  }

```

Figura 2: Función del factorial

```

21  case 1:
22      System.out.println("Ingrese el número para calcular su factorial: ");
23      int numFactorial = scanner.nextInt();
24      if (numFactorial < 0) {
25          System.out.println("Error: No puede calcularse para numeros negativos");
26      } else {
27          long resultadoFactorial = factorial(numFactorial);
28          System.out.println("El factorial de " + numFactorial + " es: " + resultadoFactorial);
29      }
30      break;
31

```

2.1. Factorial dentro del switch

```

MENÚ:
1. Factorial
2. Fibonacci
3. Secuencia de Collatz
4. Salir
Seleccione una opción (1-4): 1
Ingrese el número para calcular su factorial:
5
El factorial de 5 es: 120

```

2.2. Resultado del factorial

```

73 public static long fibonacci(int n) {
74     // Casos base 0 y 1
75     if (n == 0) {
76         return 0;
77     } else if (n == 1) {
78         return 1;
79     }
80     //caso promedio
81     return fibonacci(n - 1) + fibonacci(n - 2); // aplicando recursividad con las reglas de fibonacci
82 }

```

Figura 3: Función Fibonacci

```

32 case 2:
33     System.out.print("Ingrese la posición en la secuencia de Fibonacci: ");
34     int numFibonacci = scanner.nextInt();
35
36     if (numFibonacci < 0) { // Control de excepciones:
37         System.out.println("Error: La posición debe ser un número positivo");
38     } else {
39         //Implementación de for para mostrar secuencia completa
40         for(int i=0;i<numFibonacci;i++){
41             System.out.print(fibonacci(i)+ " ");
42         }
43     }
44     break;

```

3.1. Fibonacci dentro del switch

```

MENÚ:
1. Factorial
2. Fibonacci
3. Secuencia de Collatz
4. Salir
Seleccione una opción (1-4): 2
Ingrese la posición en la secuencia de Fibonacci: 8
0 1 1 2 3 5 8 13

```

3.2. Resultado de Fibonacci

```

84     public static void collatz(int n){
85         System.out.print(n + " ");
86         if (n == 1) { // Caso base
87             return;
88         } else if (n%2==0) { // si es par, se divide entre 2
89             collatz(n/2);
90         } else { // si es impar, se mult. por 3 y se le suma 1
91             collatz(3*n+1);
92         }
93     }
94 }
95

```

Figura 4: Función Collatz

```

46     case 3:
47         System.out.print("Ingrese un numero para secuencia de Collatz: ");
48         int numCollatz = scanner.nextInt();
49         collatz(numCollatz);
50         System.out.println();
51         break;

```

4.1. Collatz dentro del switch

```

MENÚ:
1. Factorial
2. Fibonacci
3. Secuencia de Collatz
4. Salir
Seleccione una opción (1-4): 3
Ingrese un numero para secuencia de Collatz: 7
7 22 11 34 17 52 26 13 40 20 10 5 16 8 4 2 1

```

4.2. Resultado de Collatz

5 Conclusión

Logramos comprobar que un mismo problema matemático puede resolverse con un enfoque recursivo o de forma iterativa. La recursión nos permite realizar un código más simple y cercano a la definición matemática de cada función, aunque sus desventajas implican un mayor consumo de memoria y menor eficiencia en comparación con la forma iterativa, la elección por el "mejor" método depende del problema y de los recursos con los que contamos.

6 Referencias

- [1] Eric W. Weisstein. Collatz problem. <https://mathworld.wolfram.com/CollatzProblem.html>, 2023. Consultado: 25-08-2025.