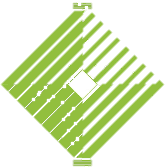
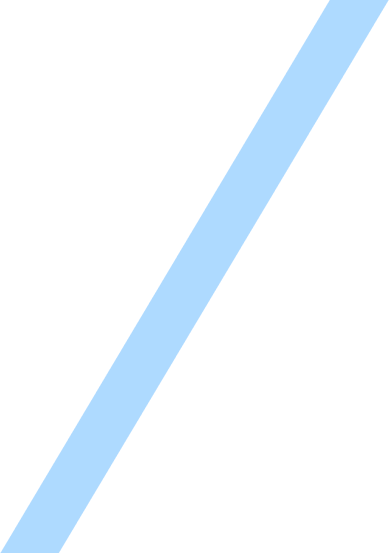
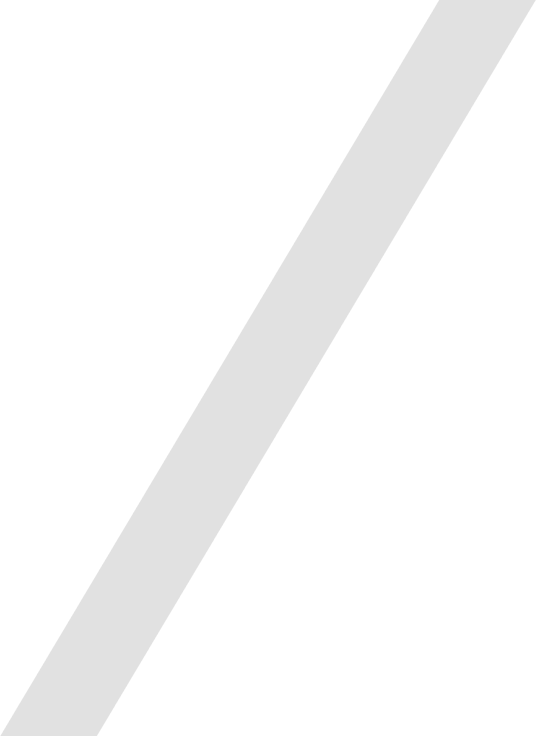
**DIVISIÓN DE TECNOLOGÍAS DE LA INFORMACIÓN Y COMUNICACIÓN**

**CARRERA**

**• DESARROLLO DE SOFTWARE MULTIPLATAFORMA•**



*ACT2-1P-RECURSIVIDAD | ESTRUCTURA DE DATOS*

PROFESORA: MIRIAN MAGALY CANCHE CAAMAL

FECHA: 20 DE SEPTIEMBRE DEL 2024

CUAT. - GRUPO: 4B-DSM

1ER PARCIAL

EDUARDO ALBERTO FLORES HERRERA

DIEGO ARMANDO CEN CABALLERO

ANDRÉS ANDREI PEREZ MENENDES

Un letrero de color blanco

Descripción generada automáticamente con confianza baja

Índice

[Introducción 3](#_Toc177725291)

[**Descripción del problema** 4](#_Toc177725292)

[**Descripción del funcionamiento paso a paso** 4](#_Toc177725293)

[**2. Ejercicio 2: Secuencia Fibonacci:** 6](#_Toc177725294)

[**3. Ejercicio 3: Resolución de la Torre de Hanoi:** 7](#_Toc177725295)

[**4. Ejercicio 4: Cálculo del MCD (Máximo Común Divisor):** 8](#_Toc177725296)

[**5. Ejercicio 5: Cajero Automático para Cambio Mínimo:** 9](#_Toc177725297)

[**5. Ejercicio 6: Dibujar Árbol de Navidad:** 11](#_Toc177725298)

[**Casos de Prueba** 12](#_Toc177725299)

[Conclusión 14](#_Toc177725300)

# Introducción

El siguiente código implementa un menú interactivo basado en consola, desarrollado en JavaScript con la ayuda del módulo readline. El objetivo de este programa es proporcionar al usuario una serie de herramientas útiles y educativas que incluyen ejercicios matemáticos clásicos como el cálculo de factoriales, secuencias de Fibonacci, la resolución de la Torre de Hanoi, y la determinación del máximo común divisor (MCD) entre dos números. Además, el programa incluye funciones prácticas como un simulador de cajero automático para calcular el cambio mínimo en diversas denominaciones y una divertida opción para dibujar un árbol de Navidad utilizando recursividad.

Este tipo de programas no solo resulta educativo, sino que también ofrece una interfaz visual atractiva a través del uso de colores para la consola, lo que mejora la experiencia del usuario. Cada función está diseñada para introducir conceptos importantes en programación como la recursividad, el manejo de errores y la interacción dinámica con el usuario, haciéndolo ideal para quienes están interesados en fortalecer sus habilidades en algoritmos y lógica computacional.

Documentación de Funcionalidad y Pruebas del Programa con Recursividad

## **Descripción del problema**

Este programa tiene como objetivo ofrecer una solución interactiva y automatizada para realizar diversas operaciones matemáticas de manera eficiente y clara. Los ejercicios abarcan desde cálculos sencillos, como el factorial o el MCD, hasta problemas más complejos como la Torre de Hanoi, la secuencia de Fibonacci, y la distribución de cambio con denominaciones específicas. La implementación está diseñada para ofrecer una experiencia visual agradable, con mensajes y tablas estilizadas en la consola.

El problema central se enfoca en proporcionar un menú interactivo donde los usuarios puedan ingresar valores numéricos, seleccionar las operaciones que desean realizar, y recibir los resultados de manera detallada, ya sea en forma de cálculos matemáticos, secuencias numéricas o tablas.

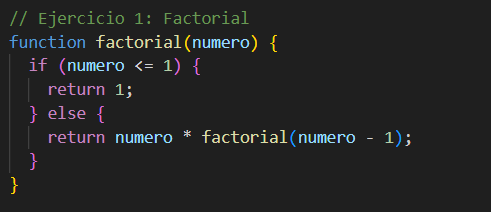
Además, se plantea la validación de los datos de entrada para asegurar que los usuarios ingresen valores correctos y manejar excepciones o errores en caso de valores inválidos, lo que añade robustez al programa. Esto garantiza que el sistema funcione sin interrupciones, proporcionando retroalimentación clara ante errores de usuario.

## **Descripción del funcionamiento paso a paso**

**1. Ejercicio 1: Cálculo de Factorial**

* **Descripción**: El usuario ingresa un número, y el programa devuelve su factorial. Si el número es inválido o negativo, se muestra un mensaje de error.
* **Entrada**: Un número entero positivo.
* **Salida**: El factorial del número.
* **Excepciones**: Si se ingresa un valor no numérico o negativo, el programa generará una advertencia.

**Captura de pantalla aquí:**



**Explicación:**

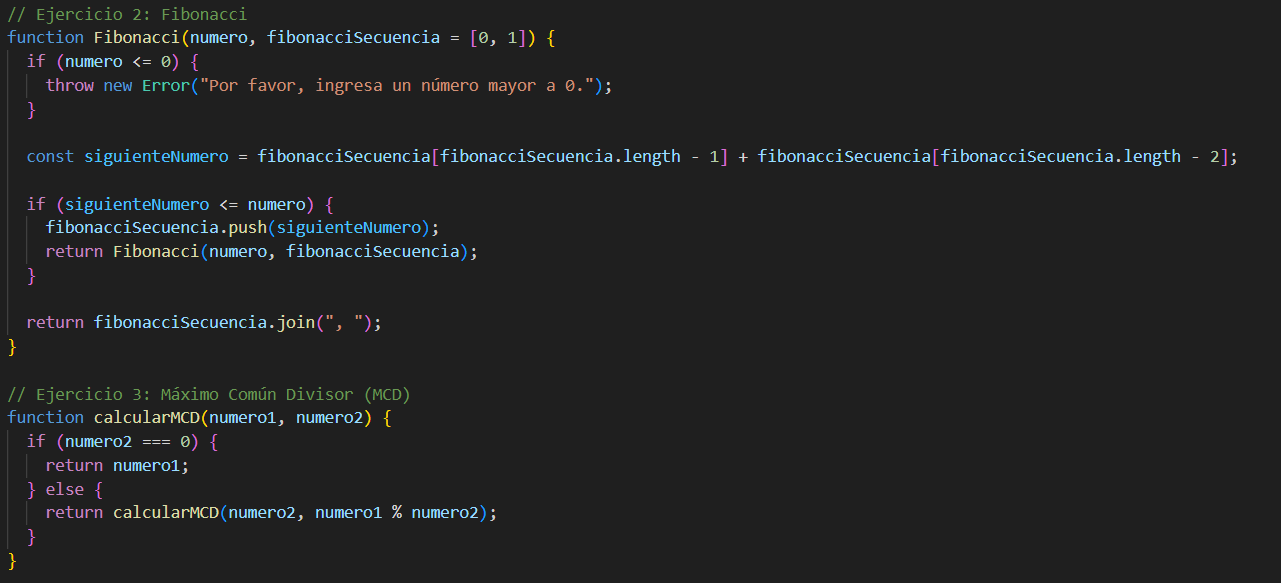
La función factorial utiliza recursividad para calcular el factorial de un número. El caso base se define cuando el número es 1 o menor, retornando 1. De lo contrario, la función se llama a sí misma con el valor de numero - 1 y multiplica ese valor por el número actual. Así, se va descomponiendo el problema hasta que llega al caso base.

* **Recursividad**: La función se llama a sí misma con un valor menor (numero - 1).
* **Dónde se usa**: Se utiliza en el else, donde se realiza la llamada recursiva.

## **2. Ejercicio 2: Secuencia Fibonacci:**

* **Descripción**: El usuario ingresa un número, y el programa genera la secuencia de Fibonacci hasta ese número.
* **Entrada**: Un número entero mayor a 0.
* **Salida**: La secuencia de Fibonacci hasta el número ingresado.
* **Excepciones**: Se muestra un mensaje de error si el número ingresado es 0 o negativo.

**Captura de pantalla aquí:**



**Explicación:**

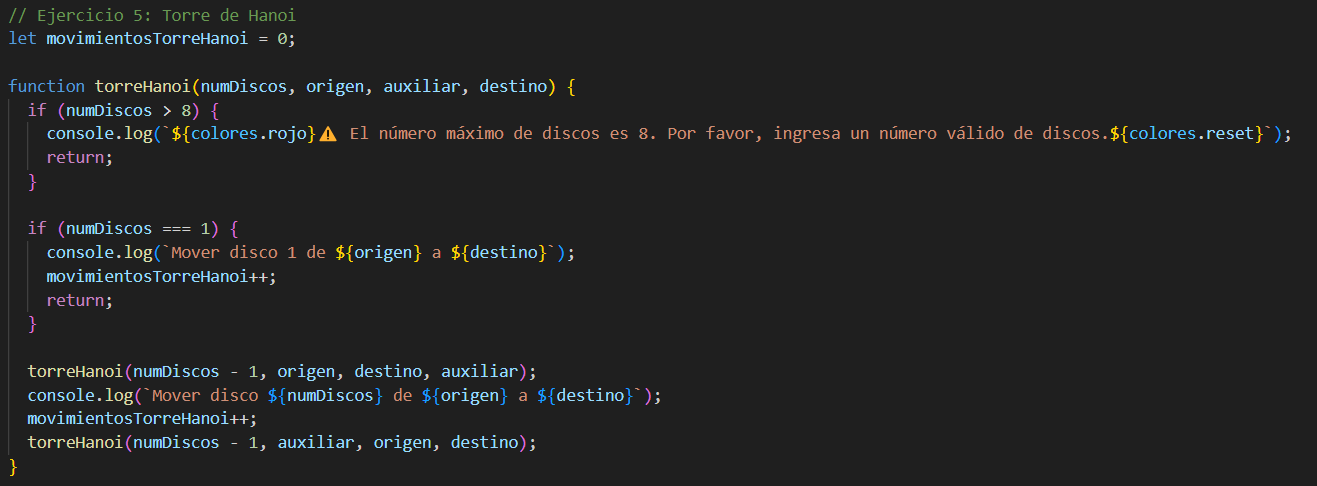
La función Fibonacci genera la secuencia de Fibonacci hasta que el siguiente número supere el valor dado. Aquí, la recursividad ocurre cuando la función se llama a sí misma después de agregar el siguiente número de la secuencia.

* **Recursividad**: La función se llama a sí misma con la secuencia actualizada si el siguiente número es menor o igual al número objetivo.
* **Dónde se usa**: En el bloque if, donde se agrega el siguiente número y luego se llama a la función de nuevo.

## **3. Ejercicio 3: Resolución de la Torre de Hanoi:**

* **Descripción**: El usuario elige la cantidad de discos para resolver la Torre de Hanoi, mostrando el número total de movimientos.
* **Entrada**: Un número de discos (máximo 8).
* **Salida**: El paso a paso de los movimientos para resolver la Torre de Hanoi.
* **Excepciones**: Si se ingresa un número mayor a 8 o un valor inválido, se muestra un mensaje de advertencia.
* **Imagen**: Captura mostrando un número válido de discos y una excepción.

**Captura de pantalla aquí:**



**Explicación:**

La función torreHanoi es un ejemplo clásico de recursividad. Para mover los discos entre torres, la función se llama a sí misma para mover numDiscos - 1 discos al poste auxiliar, luego mueve el disco más grande al destino y vuelve a llamar a la función para mover los discos restantes.

* **Recursividad**: La función se llama dos veces dentro de sí misma, primero para mover los discos a la torre auxiliar y luego al destino.
* **Dónde se usa**: En las dos llamadas recursivas, una antes y otra después de mover el disco más grande.

## **4. Ejercicio 4: Cálculo del MCD (Máximo Común Divisor):**

* **Descripción**: El programa pide dos números y calcula su MCD.
* **Entrada**: Dos números enteros.
* **Salida**: El MCD de ambos números.
* **Excepciones**: Mensaje de error si alguno de los números es negativo o no válido.

**Captura de pantalla aquí:**

Texto

Descripción generada automáticamente

**Explicación:**

Este algoritmo utiliza el método de Euclides para calcular el MCD de dos números. Si el segundo número es 0, devuelve el primer número (caso base). De lo contrario, la función se llama a sí misma con numero2 y el residuo de numero1 % numero2.

* **Recursividad**: La función se llama a sí misma con los valores actualizados hasta que el segundo número sea 0.
* **Dónde se usa**: En el else, donde la función realiza una llamada recursiva con los nuevos parámetros.

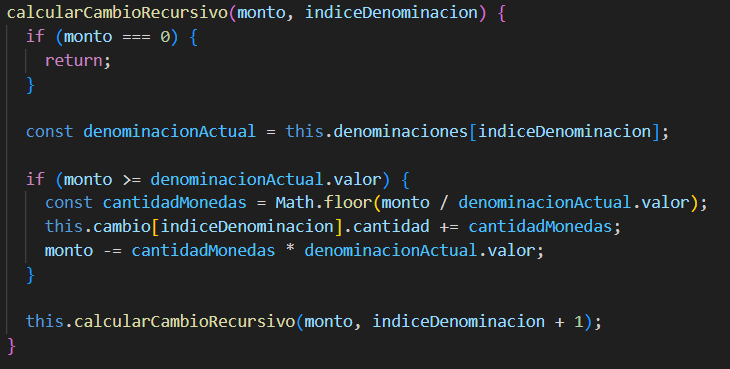
## **5. Ejercicio 5: Cajero Automático para Cambio Mínimo:**

* **Descripción**: El programa calcula el cambio mínimo utilizando denominaciones específicas.
* **Entrada**: Monto de compra y pago.
* **Salida**: Tabla de cambio detallada.
* **Excepciones**: Si el monto de pago es insuficiente, se muestra una advertencia.

**Captura de pantalla aquí:**

Texto

Descripción generada automáticamente



**Explicación:**

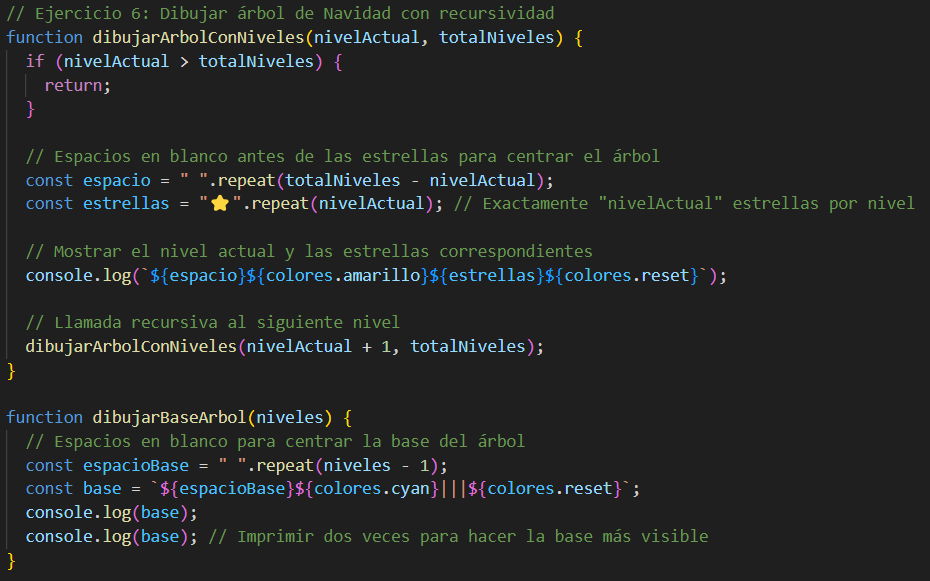
Este método de CajeroAutomatico calcula el cambio utilizando recursividad. En cada llamada, se reduce el monto según la denominación actual y luego la función se llama a sí misma con la siguiente denominación.

* **Recursividad**: La función se llama a sí misma con el siguiente índice de denominación.
* **Dónde se usa**: Al final de la función, la llamada recursiva se realiza para manejar el siguiente nivel de denominación.

## **5. Ejercicio 6: Dibujar Árbol de Navidad:**

* **Descripción**: Se dibuja un árbol de Navidad en la consola, con un número de niveles definidos por el usuario.
* **Entrada**: El número de niveles del árbol.
* **Salida**: Un árbol dibujado con estrellas y base.
* **Excepciones**: Se valida que el número de niveles sea positivo.

**Captura de pantalla aquí:**



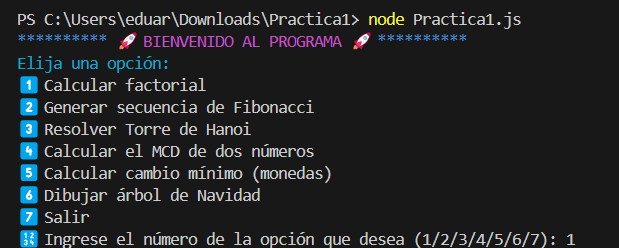
**Explicación:**

Esta función utiliza recursividad para dibujar un árbol de Navidad en la consola, donde cada nivel del árbol se dibuja con un número creciente de estrellas. La recursividad termina cuando el nivelActual supera el número de niveles totales.

* **Recursividad**: La función se llama a sí misma con el siguiente nivel hasta que el nivel actual exceda el total de niveles.
* **Dónde se usa**: Al final de la función, donde se llama a sí misma con el siguiente nivel (nivelActual + 1).

## **Casos de Prueba**

Para garantizar que los resultados sean precisos y que los usuarios proporcionen datos válidos, se han implementado varias validaciones:



1. **Ejercicio1:**

**Captura de pantalla aquí:**



1. **Ejercicio2:**

**Captura de pantalla aquí:**



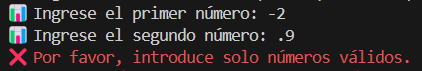
1. **Ejercicio3:**

**Captura de pantalla aquí:**



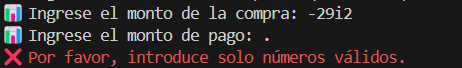
1. **Ejercicio4:**

**Captura de pantalla aquí:**



1. **Ejercicio5:**

**Captura de pantalla aquí:**



1. **Ejercicio6:**

**Captura de pantalla aquí:**



## **Conclusión**

Este programa de consola es una excelente muestra de cómo se pueden integrar diferentes técnicas de programación para ofrecer una experiencia interactiva y educativa. Al implementar funciones que cubren desde problemas matemáticos clásicos hasta simulaciones prácticas como un cajero automático, se brinda al usuario la oportunidad de aprender y practicar conceptos clave de la programación, como la recursividad y el manejo de errores. La incorporación de una interfaz visual colorida en la consola hace que la interacción sea más amena y atractiva, logrando que el aprendizaje sea más accesible y entretenido.

Además, la capacidad del programa de manejar múltiples opciones y brindar resultados instantáneos fomenta un ambiente dinámico de prueba y error, lo que refuerza las habilidades de los usuarios en la resolución de problemas. En resumen, este proyecto es una herramienta versátil y efectiva tanto para principiantes como para aquellos que buscan perfeccionar su lógica de programación, combinando funcionalidad con creatividad.