Instituto Politécnico Nacional
Centro de Estudios Científicos y Tecnológicos No 9
“Juan de Dios Bátiz”

Ingeniería Inversa

Integrantes del Equipo:

- García Gutiérrez Mariana

-García Xalpa Noé

-Ventura Ramírez José Manuel

-Velázquez Rico María Fernanda

Nombre del Profesor:

Juan Manuel Cruz Mendoza

Unidad de Aprendizaje:

Soporte de Software

Grupo:

6IM8

Entender el propósito del código

El código es un programa Java que implementa una calculadora básica. Permite al usuario realizar operaciones matemáticas simples como suma, resta, multiplicación, división, potenciación y cálculo de porcentaje. Utiliza la entrada del usuario a través de la clase `Scanner` para solicitar los números.

Diagramas

Diagrama de Grafos

10 Jump 20

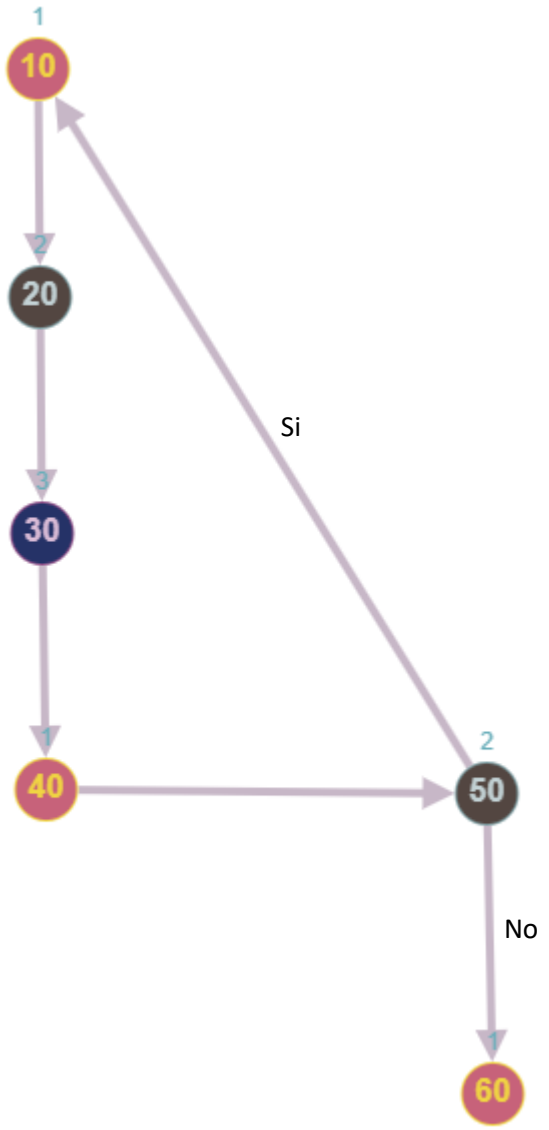
20 P1

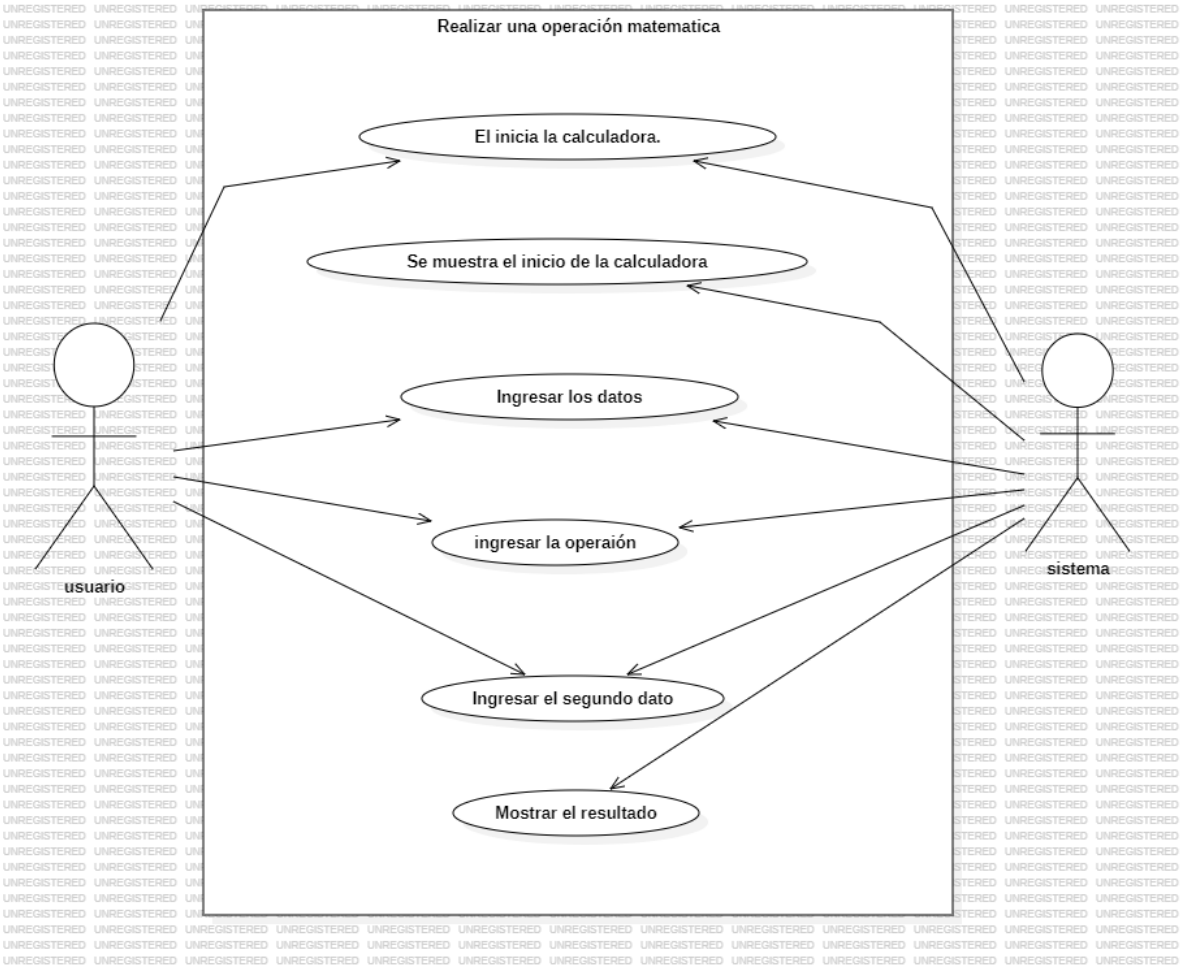
30 Jump 30

40 P2

50 If No (C1) End /
If Si (C1) Jump 10

60 End





Análisis estático del código

El código se divide en secciones claramente definidas que manejan diferentes aspectos de la interacción con el usuario y la ejecución de las operaciones matemáticas. Se utilizan comentarios para explicar la funcionalidad de cada sección y aclarar los pasos del proceso.

1. Se importa la clase Scanner para permitir la entrada de datos desde la consola.
2. Se define la clase TestCalculadora.
3. El método main es el punto de entrada del programa.
4. Se crea un objeto Scanner para leer la entrada del usuario.
5. Se inicializan variables para almacenar el resultado de la operación, la operación seleccionada y un indicador para verificar la entrada del usuario.

6. Se utiliza un bucle do-while para permitir al usuario realizar múltiples operaciones.
7. Se solicita al usuario que ingrese el primer número y se verifica que la entrada sea válida dando un numero decimal.
8. Se solicita al usuario que ingrese la operación deseada y se verifica que sea una de las operaciones permitidas (suma, resta, multiplicación, división, potencia y porcentaje).
9. Se solicita al usuario que ingrese el segundo número y se verifica que la entrada sea válida dando un numero decimal.
10. Se realiza la operación seleccionada por el usuario utilizando un switch y se almacena el resultado.
11. Tiene como especificación de si el usuario intenta dividir por cero, se solicita un nuevo segundo número hasta que sea diferente de cero.
12. Se muestra el resultado de la operación.
13. Se pregunta al usuario si desea realizar otra operación.
14. Se verifica que la entrada del usuario sea válida (s para sí, n para no) y se repite el bucle según la respuesta.

Identificación de patrones y algoritmos

El código sigue un patrón de entrada-salida donde solicita números y operaciones al usuario, realiza cálculos según la operación ingresada y muestra el resultado. Utiliza estructuras de control de flujo como bucles `do-while` y condicionales `switch` para manejar la entrada del usuario y ejecutar las operaciones matemáticas correspondientes.

Análisis dinámico del código

Al ejecutar el programa, se puede observar cómo interactúa con el usuario, solicita números y operaciones, y muestra los resultados de las operaciones realizadas. Durante la ejecución, se puede probar diferentes combinaciones de números y operaciones para verificar cómo responde el programa a diferentes entradas.

Experimentación y prueba

Se pueden realizar pruebas adicionales ejecutando el programa con diferentes conjuntos de datos de entrada y observando cómo se comporta. Se pueden probar diferentes

operaciones matemáticas y números para verificar la precisión de los cálculos y cómo maneja el programa casos especiales como la división por cero.

- Prueba de suma:
  Entrada: Primer número: 5, Operación: +, Segundo número: 3
  Resultado esperado: La suma de 5 y 3 es 8.
- Prueba de resta:
  Entrada: Primer número: 10, Operación: -, Segundo número: 7
  Resultado esperado: La resta de 10 y 7 es 3.
- Prueba de multiplicación:
  Entrada: Primer número: 4.5, Operación: x, Segundo número: 2
  Resultado esperado: El producto de 4.5 y 2 es 9.0.
- Prueba de división:
  Entrada: Primer número: 8, Operación: /, Segundo número: 2
  Resultado esperado: La división de 8 entre 2 es 4.0.
- Prueba de potenciación:
  Entrada: Primer número: 3, Operación: *, Segundo número: 3
  Resultado esperado: 3 elevado a la potencia de 3 es 27.0.
- Prueba de residuo:
  Entrada: Primer número: 10, Operación: %, Segundo número: 3
  Resultado esperado: El residuo de 10 dividido por 3 es 1.0.
- Prueba de división por cero:
  Entrada: Primer número: 6, Operación: /, Segundo número: 0
  Resultado esperado: El programa debe solicitar al usuario que ingrese un número diferente de cero como segundo número para evitar errores.