# 0. Presentación del Informe

## Título del Informe:

## Tarea Obligatoria Nro. 1: Operaciones.

## Autor:

Rodrigo Ciosek.

## Materia / Asignatura:

Algoritmos y Estructuras de Datos 1.

Profesor:

Carlos Mascheroni.

## Fecha de entrega:

8 de abril de 2025

# Introducción

Desarrollar una función que, dada cierta operación entre dos valores devuelva el resultado correspondiente utilizando únicamente sumas y restas. No está permitido el uso de operadores matemáticos directos distintos de + y – (adición y sustracción).

La operación por realizar se indicará por medio de un tercer parámetro. Si los valores no son adecuados para la operación solicitada, la función deberá manejar dicha situación de forma apropiada.

# 2. Análisis

## 2.1 Definición y entendimiento del problema

Crear una función o procedimiento que retorne el resultado de operaciones aritméticas utilizando solamente la adición y la sustracción. Se pasarán tres valores como argumentos, dos valores enteros y un carácter que define la operación aritmética.

## 2.2 Entradas y salidas:

### 2.2.1 Entradas:

* Un **valor entero “a”**.
* Un **valor entero “b”**.
* Un carácter “op” que represente un símbolo aritmético. Puede ser ‘+’, ‘-‘, ‘\*’, ‘/’ .

### 2.2.2 Salidas:

* Un numero flotante que representa el resultado de una operación aritmética con los valores **a**, **b** y el **operador**.

## 2.3 Pre y Post condiciones

### 2.3.1 Pre condiciones

* “a”, “b”, “op”, no sea nulo ni vacío
* Si “op” = 0, “num1” o “num2” != 0
* op == + o op == - o op == + o op == +

### 2.3.2 Post condiciones

* Retorna un numero entero o flotante como resultado de la operación aritmética entre a y b.
* Si alguno de los parámetros está vacío devolverá un mensaje de error.
* Si se intenta realizar una división por 0 devolverá un mensaje de error.

# 3. Diseño

### 3.2 Pseudocódigo

Clase Algoritmo.java

Funcion Calculadora(a, op, b) : Flotante

Si op != '-' Y op != '+' Y op != '/' Y op != '\*'

Retornar "NaN"

FinSi

Segun op Hacer

Caso '-':

Retornar Utilidades.Resta(a,b)

Caso '+':

Retornar Utilidades.Suma(a,b)

Caso '/':

Retornar Utilidades.Division(a,b)

Caso '\*':

Retornar Utilidades.Multiplicacion(a,b)

FinFuncion

Clase Utilidades.Java

Funcion suma(a,b) : Entero

Retornar a + b

FinFuncion

Funcion resta(a,b) : Entero

Retornar a - b

FinFuncion

Funcion multiplicacion(a,b) : Entero

resultado = 0

absA = FuncionValorAbsoluto(a)

absB = FuncionValorAbsoluto(b)

Para i = 0 Hasta i=absB Hacer

resultado = resultado + absA;

FinPara

Si a < 0 Y b < 0

Retornar FuncionValorAbsoluto(resultado)

FinSi

Si a < 0 O b < 0

Retornar resultado \* -1

FinSi

Retornar resultado

FinFuncion

Funcion division(a,b) : Flotante

resultado = 0

resto = 0

cociente = 0

absA = ValorAbsoluto(a)

absB = ValorAbsoluto(b)

restoAux = 0

restoDivLarga = 0

decimal = 0

// Calcular cociente

Mientras restaA >= absB Hacer

restaA = restaA - absB

cociente = cociente + 1

FinMientras

resto = restaA

resultado = cociente

Si resto = 0

Retornar resultado

FinSi

// Division larga para conseguir el resto en decimal

// 1. Mutliplicar el resto \* 10

restoAux = resto

Para i = 1 hasta i = 10 Hacer

resto = resto + restoAux

FinPara

// 2. Conseguir cuantas veces entra el divisor en el resto

Mientras resto >= absA Hacer

resto = resto - absA

restoDivLarga = restoDivLarga + 1

FinMientras

// 3. Convertir restoDivLarga a decimal para sumarlo al resultado

Para i = 0 Hasta i = restoDivLarga

decimal = decimal + 0.1

FinPara

// 4. Sumarlo a resultado

resultado = resultado + decimal

Retornar resultado

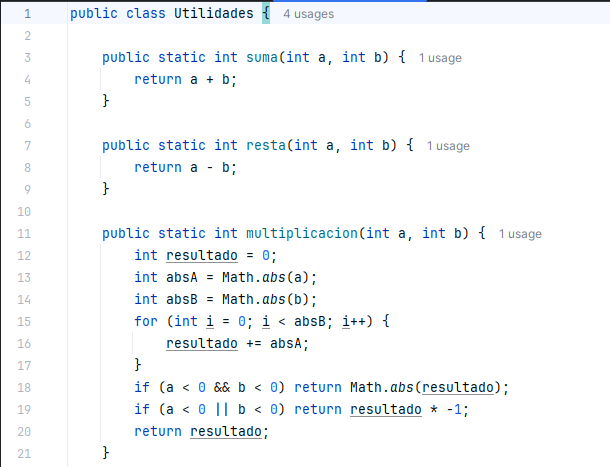
FinFuncion

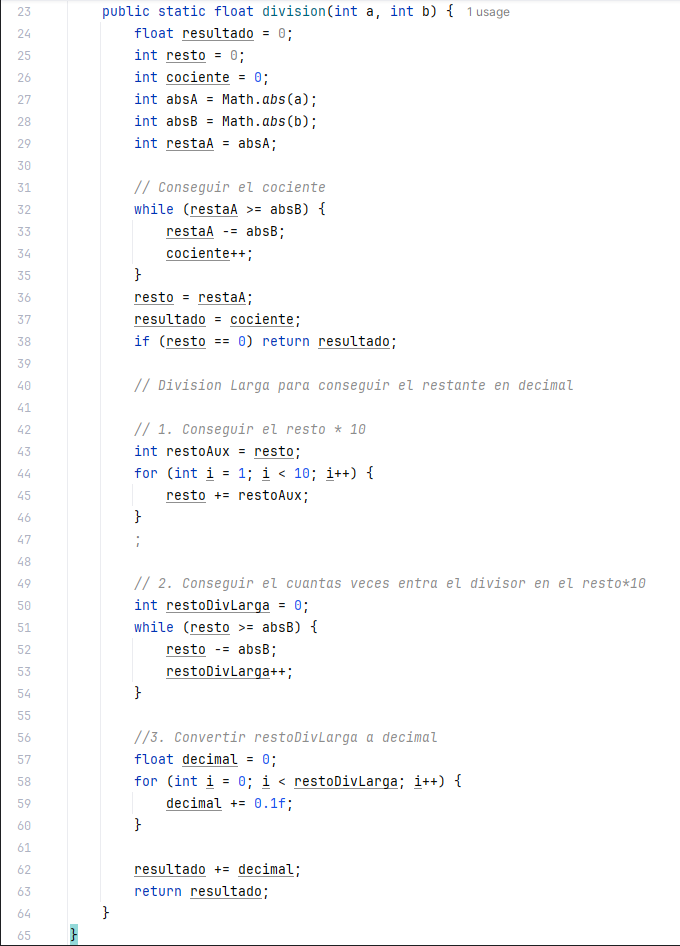
\*/

# 4. Implementación

Algoritmo.java

# 

Utilidades.java



# 5. Verificación y Validación

### Pruebas

* **Caso básico 1:**
  + (-1, '+' , 5) → Resultado esperado: 4.0
* **Caso básico 2:**
  + (1, '+' , -5) → Resultado esperado: -4.0
* **Caso básico 3:**
  + (6, '-' , 5) → Resultado esperado: 1.0
* **Caso básico 4:**
  + (5, '-' , 6) → Resultado esperado: -1.0
* **Caso básico 5:**
  + (-5, '-' , -6) → Resultado esperado: 1.0
* **Caso básico 6:**
  + (1, '\*' , 5) → Resultado esperado: 5.0
* **Caso básico 7:**
  + (6, '\*' , -2) → Resultado esperado: -12.0
* **Caso básico 8:**
  + (0, '\*' , 10) → Resultado esperado: 0.0
* **Caso básico 9:**
  + (10, '\*' , 0) → Resultado esperado: 0.0
* **Caso básico 10:**
  + (1, '/' , -5) → Resultado esperado: -0.2
* **Caso básico 11:**
  + (5, '/' , 0) → Resultado esperado: NaN
* **Caso básico 12:**
  + (0, '/' , 5) → Resultado esperado: 0.0
* **Caso básico 13:**
  + (0, ' ' , 5) → Resultado esperado: NaN
* **Caso básico 14:**
  + (0, '&' , 5) → Resultado esperado: NaN

### Resultados:

* **Caso básico 1:** 4.0 (Correcto)
* **Caso básico 2:** -4.0 (Correcto)
* **Caso básico 3:** 1.0 (Correcto)
* **Caso básico 4:** -1.0 (Correcto)
* **Caso básico 5:** 1.0 (Correcto)
* **Caso básico 6:** 5.0 (Correcto)
* **Caso básico 7:** -12.0 (Correcto)
* **Caso básico 8:** 0.0 (Correcto)
* **Caso básico 9:** 0.0 (Correcto)
* **Caso básico 10:** -0.2 (Correcto)
* **Caso básico 11:** NaN (Correcto)
* **Caso básico 12:** 0.0 (Correcto)
* **Caso básico 13:** NaN (Correcto)
* **Caso básico 14:** NaN (Correcto)

# 6. Conclusiones

Buen ejercicio para poder entender como funciona la multiplicación y la división con mas profundidad y aprender a aplicarlo en código. Me resulto complicada la división, me ayudo a investigar como podría conseguir que me devolviera el resultado con resto y aprendí un concepto que se llama división larga, asique me llevo un concepto nuevo.