## **Documentación del Módulo de Cálculo (**modulos/calculo.py**)**

### ****Propósito del Módulo****

El módulo de cálculo proporciona funciones para realizar tanto **operaciones matemáticas básicas** (como suma, resta, multiplicación y división) como **operaciones avanzadas** (como derivadas, integrales, y funciones similares a las de Excel como SUMAR.SI, BDCONTARA, etc.). Este módulo es llamado por el controlador central para ejecutar diversas acciones matemáticas y de manejo de bases de datos.

### ****Funciones del Módulo****

#### 1. realizar\_calculo(tipo\_calculo, valores)

* **Propósito**: Esta es la función principal del módulo. Recibe un tipo de cálculo y los valores necesarios para realizarlo, y delega la tarea al cálculo correspondiente.
* **Parámetros**:
  + **tipo\_calculo** (str): Un string que especifica el tipo de cálculo que se desea realizar. Puede ser 'suma', 'resta', 'multiplicacion', 'division', 'derivada', 'integral', 'sumar\_si', 'bdcontara', 'bdextraer', 'bdmax', 'bdmin'.
  + **valores** (list): Una lista que contiene los datos necesarios para realizar el cálculo. El formato de esta lista varía dependiendo del tipo de cálculo (puede ser una lista de números o una expresión simbólica).
* **Salida**: Devuelve el resultado del cálculo realizado. Puede ser un número, una expresión simbólica, o un mensaje de error.
* **Descripción**: La función recibe la solicitud del tipo de cálculo y, dependiendo del tipo de acción, llama a la función adecuada para ejecutar el cálculo correspondiente.

**Ejemplo de uso**:

resultado = realizar\_calculo('suma', [10, 20, 30])

# resultado = 60

#### 2. suma(valores)

* **Propósito**: Realiza la suma de todos los valores contenidos en la lista valores.
* **Parámetros**:
  + **valores** (list): Lista de números que se desean sumar.
* **Salida**: El resultado de la suma de todos los valores en la lista.
* **Descripción**: Suma todos los valores de la lista usando la función sum() de Python.

**Ejemplo de uso**:

resultado = suma([10, 20, 30])

# resultado = 60

#### 3. resta(valores)

* **Propósito**: Realiza la resta de los valores de la lista valores. La operación resta el primer valor con los valores subsecuentes.
* **Parámetros**:
  + **valores** (list): Lista de números de los que se realizará la resta.
* **Salida**: El resultado de la resta de los valores en la lista.
* **Descripción**: Realiza una resta sucesiva: el primer valor menos la suma de los demás.

**Ejemplo de uso**:

resultado = resta([100, 20, 10])

# resultado = 70 (100 - 20 - 10)

#### 4. multiplicacion(valores)

* **Propósito**: Realiza la multiplicación de todos los valores de la lista valores.
* **Parámetros**:
  + **valores** (list): Lista de números que se desean multiplicar.
* **Salida**: El resultado de la multiplicación de todos los valores en la lista.
* **Descripción**: Multiplica los valores de la lista en una secuencia.

**Ejemplo de uso**:

resultado = multiplicacion([2, 3, 4])

# resultado = 24 (2 \* 3 \* 4)

#### 5. division(valores)

* **Propósito**: Realiza la división de los valores de la lista valores. La operación divide el primer valor por los valores subsecuentes.
* **Parámetros**:
  + **valores** (list): Lista de números en los que se realizará la división.
* **Salida**: El resultado de la división sucesiva de los valores.
* **Descripción**: Realiza una división sucesiva: el primer valor dividido por el siguiente y así sucesivamente.

**Ejemplo de uso**:

resultado = division([100, 2, 5])

# resultado = 10 (100 / 2 / 5)

#### 6. derivada(valores)

* **Propósito**: Calcula la derivada de una expresión simbólica.
* **Parámetros**:
  + **valores** (list): Una lista que contiene una expresión matemática (en formato de texto) para la cual se calculará la derivada.
* **Salida**: La derivada de la expresión.
* **Descripción**: Utiliza la librería sympy para calcular la derivada de una expresión simbólica respecto a una variable. La variable predeterminada es x.

**Ejemplo de uso**:

resultado = derivada(['x\*\*2 + 3\*x + 2'])

# resultado = 2\*x + 3

#### 7. integral(valores)

* **Propósito**: Calcula la integral de una expresión simbólica.
* **Parámetros**:
  + **valores** (list): Una lista que contiene una expresión matemática (en formato de texto) para la cual se calculará la integral.
* **Salida**: La integral de la expresión.
* **Descripción**: Utiliza sympy para calcular la integral indefinida de una expresión simbólica respecto a una variable. La variable predeterminada es x.

**Ejemplo de uso**:

resultado = integral(['x\*\*2 + 3\*x + 2'])

# resultado = x\*\*3/3 + 3\*x\*\*2/2 + 2\*x

#### 8. sumar\_si(valores)

* **Propósito**: Suma los valores de un rango que cumplan con un criterio específico.
* **Parámetros**:
  + **valores** (list): Una lista donde el primer elemento es un rango de valores (como una lista de precios) y el segundo es el criterio de suma.
* **Salida**: La suma de los valores que cumplen con el criterio especificado.
* **Descripción**: Utiliza pandas para aplicar un filtro al rango y luego sumar los valores que cumplen con el criterio.

**Ejemplo de uso**:

resultado = sumar\_si([ [100, 200, 150], 150 ])

# resultado = 350 (100 + 200)

#### 9. bdcontara(valores)

* **Propósito**: Cuenta el número de celdas no vacías en una base de datos.
* **Parámetros**:
  + **valores** (list): Lista de datos (puede ser una lista de celdas).
* **Salida**: El número de celdas no vacías en los datos proporcionados.
* **Descripción**: Utiliza pandas para contar las celdas no vacías.

**Ejemplo de uso**:

resultado = bdcontara([1, 2, None, 4, 5])

# resultado = 4

#### 10. bdextraer(valores)

* **Propósito**: Extrae un único registro de una base de datos que cumple con los criterios especificados.
* **Parámetros**:
  + **valores** (list): Lista donde el primer elemento es un conjunto de datos (como un DataFrame) y los otros son criterios de búsqueda.
* **Salida**: Un único registro que cumple con los criterios especificados.
* **Descripción**: Utiliza pandas para filtrar los datos y extraer un único registro que cumpla con los criterios.

**Ejemplo de uso**:

datos = [{"nombre": "Juan", "edad": 30}, {"nombre": "Ana", "edad": 25}]

resultado = bdextraer([datos, 'nombre', 'Juan'])

# resultado = {"nombre": "Juan", "edad": 30}

#### 11. bdmax(valores)

* **Propósito**: Devuelve el valor máximo de las entradas seleccionadas de la base de datos.
* **Parámetros**:
  + **valores** (list): Una lista de datos.
* **Salida**: El valor máximo en los datos proporcionados.
* **Descripción**: Utiliza pandas para encontrar el valor máximo.

**Ejemplo de uso**:

resultado = bdmax([10, 20, 30, 40])

# resultado = 40

#### 12. bdmin(valores)

* **Propósito**: Devuelve el valor mínimo de las entradas seleccionadas de la base de datos.
* **Parámetros**:
  + **valores** (list): Una lista de datos.
* **Salida**: El valor mínimo en los datos proporcionados.
* **Descripción**: Utiliza pandas para encontrar el valor mínimo.

**Ejemplo de uso**:

resultado = bdmin([10, 20, 30, 40])

# resultado = 10

### ****Conclusión****

Este módulo de cálculo abarca una amplia gama de operaciones, tanto **básicas** como **avanzadas**, y es fácilmente extensible para agregar más funcionalidades. Cada función está diseñada para ser modular y reutilizable, permitiendo que el controlador central ejecute diversas operaciones matemáticas y funciones tipo Excel de manera sencilla.

¡Este módulo está listo para ser integrado con el resto del sistema y comenzar a realizar cálculos y análisis de datos!