Universidad de Granada

Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación

PRÁCTICA 2
Apache Thrift

Desarrollo de Sistemas Distribuidos

Autor:
Raquel Molina Reche

Abril 2021

1. Introduction

Para la realización de esta práctica he tenido en cuenta 4 grupos de operaciones según la estructura de datos que estas manejan:

- Operaciones Básicas.
  Suma, resta, producto, división, logaritmo y potencia.
- Operaciones con vectores de cualquier tamaño.
  Suma y resta de vectores.
- Operaciones con vectores 3D.
  Producto vectorial y producto escalar de vectores de 3 dimensiones.
- Operaciones con matrices cuadradas.
  Suma, resta y producto de matrices.

Todo esto haciendo uso de la estructura cliente servidor mediante Apache Thrift. De forma que a partir del compilador se genera el código para la implementación de clientes y servidores que se comunican mediante RPC y con la posibilidad de hacerlo en diferentes lenguajes de programación. En este caso se ha usado Python y Java, en Python se ha generado un cliente y un servidor y en Java un cliente que se comunica con el servidor de Python.

2. Planteamiento de la solución

El programa del fichero IDL *calculadora.thrift* se muestra en la Figura 1, tiene 13 métodos que equivalen a las distintas operaciones que se ofrecen.

```
service Calculadora{
   void ping(),
   i32 suma(1:i32 num1, 2:i32 num2),
   i32 resta(1:i32 num1, 2:i32 num2),
   double multiplicacion(1:double num1, 2:double num2),
   double division(1:double num1, 2:double num2),
   double logaritmo(1:double num1, 2:double num2),
   double potencia(1:double num1, 2:i32 num2),
   list<double> sumavectores(1:list<double> v1, 2:list<double> v2),
   list<double> restavectores(1:list<double> v1, 2:list<double> v2),
   double productoescalar3d(1:Vector3D v3D1, 2:Vector3D v3D2),
   Vector3D productovectorial3d(1:Vector3D v3D1, 2:Vector3D v3D2),
   Matriz sumamatrices(1:Matriz m1, 2:Matriz m2),
   Matriz restamatrices(1:Matriz m1, 2:Matriz m2),
   Matriz productomatrices(1:Matriz m1, 2:Matriz m2),
}
```

Figura 1: Fichero IDL métodos

2.1. Estructuras de datos complejas

Para el uso de estructuras de datos complejas se han tenido en cuenta las categorías: vectores de 3 dimensiones y matrices.

Para las operaciones con vectores 3D, se ha tenido en cuenta la estructura de la Figura 2, *Vector3D* como una estructura que contiene el valor de las 3 coordenadas (x y z) del mismo.

```
//---------Estructura que compone un vector 3D-----------
struct  Vector3D{
   1:  required  double  x;   //Componente x 3D
   2:  required  double  y;   //Componente y 3D
   3:  required  double  z;   //Componente z 3D
}
```

Figura 2: Estructura Vector 3D

En estas operaciones la entrada son datos de este tipo (Vector3D). Para el producto escalar el resultado es un dato de tipo double pero en el caso de producto vectorial, el resultado que se devuelve es otro vector de 3 dimensiones.

Para las operaciones con matrices cuadradas, se ha usado la estructura de la Figura 3, *Matriz* como una estructura que contiene la información de la misma, es decir la estructura almacena el número de filas, el número de columnas y un dato de tipo *list*. Esta lista tendrá de tamaño filas*columnas. El acceso a los elementos de la matriz a partir de la lista se realiza de la siguiente manera:

*(fila a la que se quiere acceder * número de columnas de la matriz) + columna a la que se quiere acceder*

```
//---------Estructura que compone una matriz-----------
struct  Matriz{
   1:  required  i32  f;   //filas
   2:  required  i32  c;   //columnas
   3:  required  list<double> m; //lista que contiene los elementos de la matriz
}
```

Figura 3: Estructura para una Matriz

Con esta estructura se podrían manejar matrices que no fueran cuadradas ya que se almacena tanto el valor de las filas como el de las columnas pero se ha restringido en este caso a matrices cuadradas por la comodidad de las operaciones. En estas operaciones la entrada y la salida son datos de este tipo (Matriz).

También se realizan operaciones con vectores pero en este caso no requieren de ninguna estructura pues para Thrift se puede usar el tipo list.

Para el uso de estas estructuras por parte del cliente y del servidor es necesario que se importe o incluya en los mismos al archivo que genera el compilador con las estructuras, tiene el sufijo *ttypes*.

3. Generación ficheros fuente

3.1. Python

$$\$thrift\ -gen\ py\ calculadora.thrift$$

En este caso ya se generan todos los ficheros necesarios para la ejecución del cliente y el servidor.

3.2. Java

$$\$thrift\ -r\ --gen\ java\ calculadora.thrift$$

En el caso de java se ha usado como IDE *Intellij* y se han incorporado de forma sencilla los archivos .jar que son requeridos. Están todos los que se han usado en la carpeta *jar*.

4. Pasos para la implementación del cliente y el servidor

4.1. Implementar el Servidor

1. Instalar/importar el paquete thrift en el lenguaje que se esté programando el servicio.
2. Importar clases generadas en la compilación del fichero IDL. (Importante importar las clases con las estructuras de datos necesarias).
3. Generar la clase handler e implementar las operaciones.
4. Crear el objeto server e iniciarlo.

4.2. Implementar el Cliente

1. Instalar/importar paquete thrift en el lenguaje que se esté programando el servicio.
2. Importar clases generadas en la compilación del fichero IDL. (Importante importar las clases con las estructuras de datos necesarias).
3. Crear el objeto cliente y llamar a los métodos que realizan las operaciones.

5. Ejemplos de ejecución

Para la ejecución, en el cliente se han manejado las operaciones a partir de un menú en el que se van introduciendo las distintas peticiones y cuya implementación llama a los métodos necesarios para obtener los resultados. A continuación se muestran imagenes con la ejecución.

IMPORTANTE: Los operandos de la suma y la resta y el exponente de la potencia son enteros, el resto son número decimales. Además en el caso de Java los decimales deben introducirse separando la parte decimal de la entera con una coma ‘,’, en el caso de Python se separa con punto ‘.’.

5.1. Servidor

$ python3 servidor.py

5.2. Clientes

Python : $ python3 cliente.py

Java : La compilación y ejecución del cliente se ha realizado con el IDE *Intellij*. Para ello solo hace falta que en el proyecto se encuentren los archivos *Cliente.java Calculadora.java Vector3D.java y Matriz.java* y se incluyan los archivos de la carpeta jar. Se ejecuta el main de *Cliente*.

5.3. Ejemplos

Operaciones Básicas

```
Opciones disponibles:
  1: Operaciones Básicas
  2: Operaciones con vectores
  3: Operaciones con vectores 3D
  4: Operaciones con matrices cuadradas
  5: Salir

--Introducce una opción: 1

        ----OPERACIONES BÁSICAS----
        Operaciones disponibles:
            1: Suma
            2: Resta
            3: Multiplicacion
            4: Division
            5: Logaritmo
            6: Potencia
            7: Salir

            --Introducce una opción: 1
            Introduce el primer operando (entero): 55
            Introduce el segundo operando (entero): 6

------------------------------------------------------------------------

  El resultado de la operación  55  +  6  =  61

------------------------------------------------------------------------
```

Figura 4: Suma (Cliente Python)

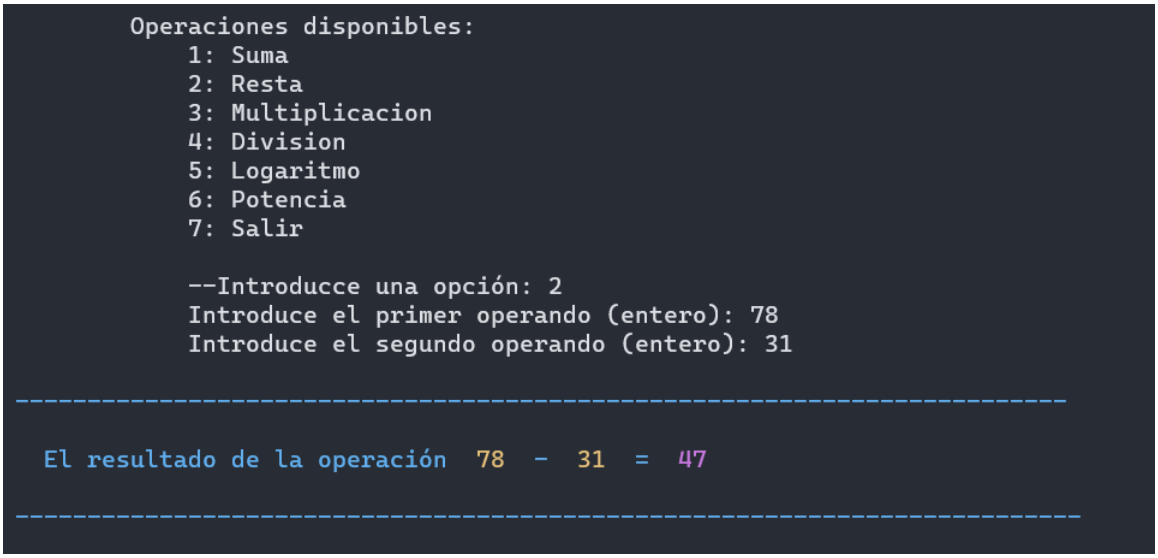

Figura 5: Resta (Cliente Python)

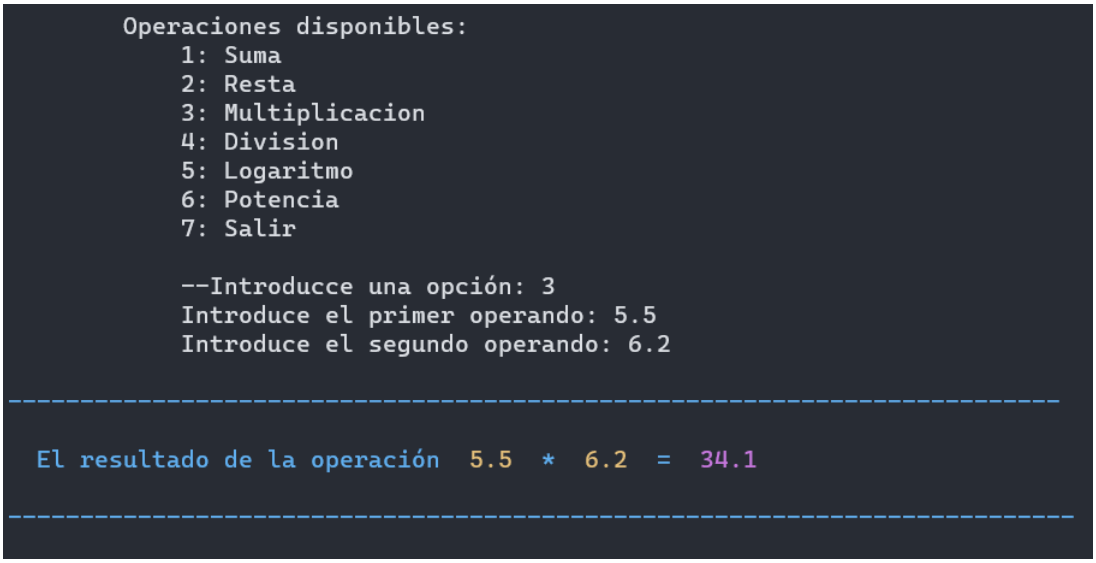

Figura 6: Multiplicación (Cliente Python)

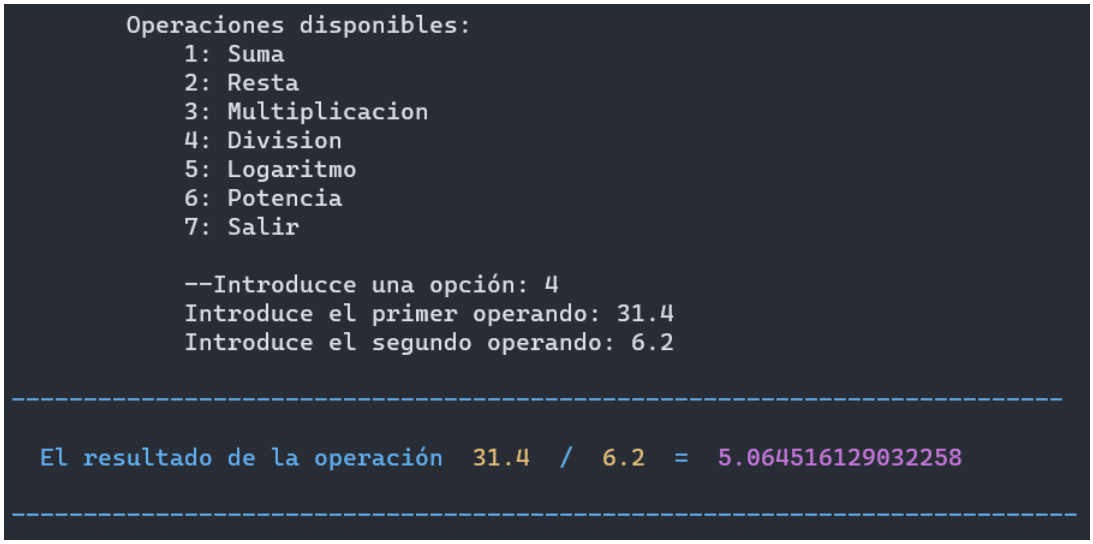

Figura 7: División (Cliente Python)

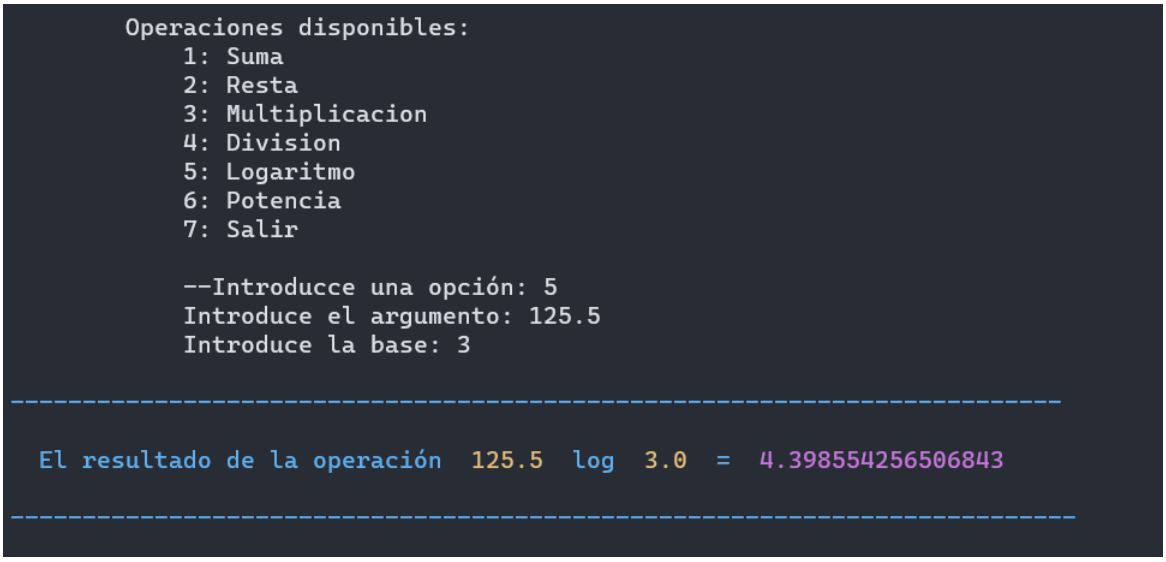

Figura 8: Logaritmo (Cliente Python)

```
        Operaciones disponibles:
            1: Suma
            2: Resta
            3: Multiplicacion
            4: Division
            5: Logaritmo
            6: Potencia
            7: Salir

            --Introducce una opción: 6
            Introduce la base: 2
            Introduce el exponente (entero): 9
------------------------------------------------------------------------

  El resultado de la operación  2.0  ^  9  =  512.0

------------------------------------------------------------------------

        Operaciones disponibles:
            1: Suma
            2: Resta
            3: Multiplicacion
            4: Division
            5: Logaritmo
            6: Potencia
            7: Salir

            --Introducce una opción: 7
            Saliendo...
```

Figura 9: Potencia y salir del menú de Operaciones Básicas (Cliente Python)

5.3.1. Operaciones con vectores de cualquier tamaño

```
Opciones disponibles:
  1: Operaciones Básicas
  2: Operaciones con vectores
  3: Operaciones con vectores 3D
  4: Operaciones con matrices cuadradas
  5: Salir

--Introducce una opción: 2

        ----OPERACIONES CON VECTORES----
        Introduce el tamaño de los vectores: 6
        Contenido del primer vector (v1):
        v1[0]: 2.3
        v1[1]: 5.6
        v1[2]: 9.5
        v1[3]: 1.2
        v1[4]: 6.2
        v1[5]: 3.8
        Contenido del segundo vector (v2):
        v2[0]: 8.3
        v2[1]: 8.2
        v2[2]: 9.5
        v2[3]: 7.3
        v2[4]: 7.4
        v2[5]: 5
        Operaciones disponibles:
            1: Suma
            2: Resta
            3: Salir

            --Introducce una opción: 1

------------------------------------------------------------------------
  El resultado de la operación:
        2.3  5.6  9.5  1.2  6.2  3.8
                +
        8.3  8.2  9.5  7.3  7.4  5.0
                  =
        10.600000000000001  13.799999999999999  19.0  8.5  13.600000000000001  8.8
------------------------------------------------------------------------

        Operaciones disponibles:
            1: Suma
            2: Resta
            3: Salir

            --Introducce una opción: 2

------------------------------------------------------------------------
  El resultado de la operación:
        2.3  5.6  9.5  1.2  6.2  3.8
                -
        8.3  8.2  9.5  7.3  7.4  5.0
                  =
        -6.000000000000001  -2.5999999999999996  0.0  -6.1  -1.2000000000000002  -1.2000000000000002
------------------------------------------------------------------------

        Operaciones disponibles:
            1: Suma
            2: Resta
            3: Salir

            --Introducce una opción: 3
            Saliendo...
```


Figura 10: Suma y resta de vectores (Cliente Python)

5.3.2. Operaciones con vectores 3D

```
Opciones disponibles:
  1: Operaciones Básicas
  2: Operaciones con vectores
  3: Operaciones con vectores 3D
  4: Operaciones con matrices cuadradas
  5: Salir

--Introducce una opción: 3

        ----OPERACIONES CON  VECTORES 3D----
        Contenido del primer vector (v1):
              x:-15.5
              y:7.8
              z:0.0
        Contenido del segundo vector (v2):
              x:5.6
              y:-1.0
              z:50.0
        Operaciones disponibles:
            1: Producto escalar
            2: Producto vectorial
            3: Salir

            --Introducce una opción: 1

--------------------------------------------------------------------------
  El resultado de la operación:

        (-15.5,7.8, 0.0) * (5.6,-1.0, 50.0) =  -94.6

--------------------------------------------------------------------------

        Operaciones disponibles:
            1: Producto escalar
            2: Producto vectorial
            3: Salir

            --Introducce una opción: 2

--------------------------------------------------------------------------
  El resultado de la operación:

        (-15.5, 7.8, 0.0) · (5.6, -1.0, 50.0) = (390.0, -775.0, -28.18)

--------------------------------------------------------------------------

        Operaciones disponibles:
            1: Producto escalar
            2: Producto vectorial
            3: Salir

            --Introducce una opción: 3
            Saliendo...
```

Figura 11: Producto escalar y producto vectorial de vectores 3D (Cliente Python)

5.3.3. Operaciones con matrices cuadradas

```
Opciones disponibles:
  1: Operaciones Básicas
  2: Operaciones con vectores
  3: Operaciones con vectores 3D
  4: Operaciones con matrices cuadradas
  5: Salir
  --Introducce una opción: 4

    ----OPERACIONES CON MATRICES CUADRADAS----
    Introduce un tamaño positivo mayor que 0: 3
    Contenido de la primera matriz (m1) (decimal con coma):
    m1[0][0]: 1
    m1[0][1]: 4
    m1[0][2]: 7
    m1[1][0]: 2
    m1[1][1]: 5
    m1[1][2]: 8
    m1[2][0]: 3
    m1[2][1]: 6
    m1[2][2]: 9
    Contenido del segunda matriz (m2) (decimal con coma):
    m2[0][0]: 1
    m2[0][1]: -1
    m2[0][2]: 2
    m2[1][0]: 2
    m2[1][1]: -1
    m2[1][2]: 2
    m2[2][0]: 3
    m2[2][1]: -3
    m2[2][2]: 0
```

Figura 12: Incluir datos de las matrices cuadradas (Cliente Java)

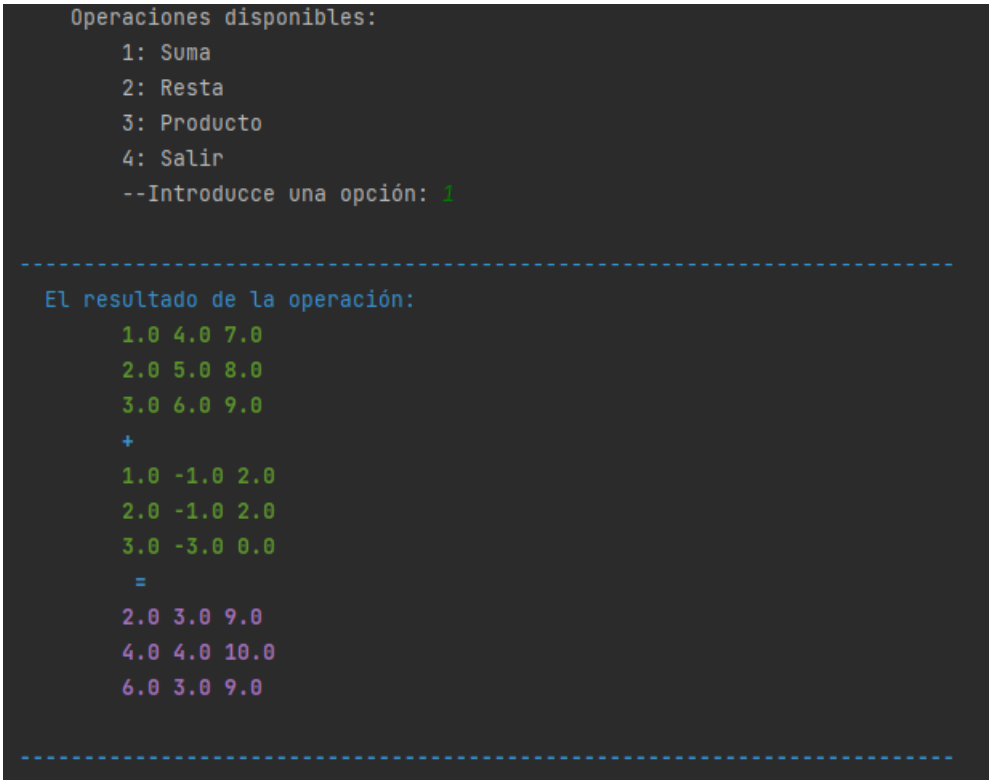

Figura 13: Suma de matrices cuadradas (Cliente Java)

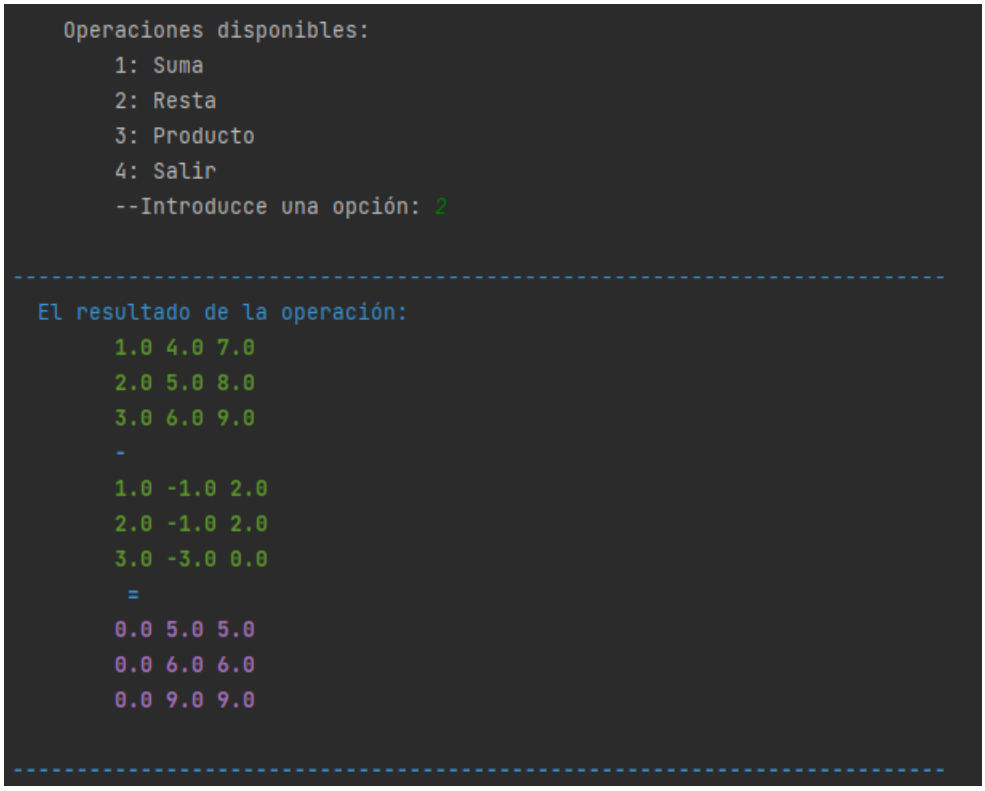

Figura 14: Resta de matrices cuadradas (Cliente Java)

```
    Operaciones disponibles:
        1: Suma
        2: Resta
        3: Producto
        4: Salir
        --Introducce una opción: 3

--------------------------------------------------------------------------------
  El resultado de la operación:
        1.0 4.0 7.0
        2.0 5.0 8.0
        3.0 6.0 9.0
        *
        1.0 -1.0 2.0
        2.0 -1.0 2.0
        3.0 -3.0 0.0
         =
        30.0 -26.0 10.0
        36.0 -31.0 14.0
        42.0 -36.0 18.0

--------------------------------------------------------------------------------
```

Figura 15: Producto de matrices cuadradas (Cliente Java)