Multiplicación de enteros grandes

Esta práctica nos permite confirmar lo que ya sabíamos y es que el paradigma de dividir y vencer nos genera algoritmos muy eficaces. Para poder comprobar lo anterior hemos realizado ya varias prácticas y considero que este es un ejemplo sencillo y comprobable de lo anterior, es sobre algo básico de la vida cotidiana que es sobre hacer multiplicaciones.

Al desarrollar esta práctica me encontré con un dilema, en ocasiones la multiplicación de dos int puede dar un long, podemos multiplicar un int y un long un long y un BigInteger, pero decidí no adentrarme en ese problema ya que implicaría sobrecargar los métodos más veces y eso es algo no adecuado. Así que decidí que la mejor sería decirle al usuario aquí debes ingresar dos int después, dos long y después dos BigInteger, así nos quitamos de complicaciones siguiendo un consejo que se maneja en Ingeniería de Software de no complicarse si no es necesario.

Después de lo anterior y de haber programado la clase MultiplicacionEntera la cual es nuestro main de la practica pensé, como puedo garantizar que el algoritmo para la multiplicación entera es realmente eficiente y es menos costoso que el normal, entonces decidí hacer la multiplicación como lo hace Java, es decir, con el símbolo '*' en el caso de BigInteger es con multiply y compararlo con el algoritmo que se está estudiando, la comparación se hace en el tiempo de ejecución de cada uno.

Lo anterior nos permite ver que tan eficiente es nuestra forma de multiplicar, el ejemplo de ejecución lo vemos en la Imagen 1.

```
Multiplicacion usando el dato primitivo: int
Ingrese el primer numero a multiplicar
12345
Ingrese el segundo numero a multiplicar
12345
Multiplicacion como lo hace Java: 152399025
Tiempo de ejecucion en milisegundos:0.142
Multiplicacion usando dividir y vencer: 152399025
Tiempo de ejecucion en milisegundos:0.0585
**********************************************************************
Multiplicacion usando el dato primitivo: long
Ingrese el primer numero a multiplicar
1234567
Ingrese el segundo numero a multiplicar
1234567
Multiplicacion como lo hace Java: 1524155677489
Tiempo de ejecucion en milisegundos:0.1037
Multiplicacion usando dividir y vencer: 1524155677489
Tiempo de ejecucion en milisegundos:0.0376
**********************************************************************
DMultiplicacion usando el dato primitivo: BigInteger
Ingrese el primer numero a multiplicar
123456789
Ingrese el segundo numero a multiplicar
123456789
Multiplicacion como lo hace la clase BigInteger: 15241578750190521
Tiempo de ejecucion en milisegundos:0.6601
Multiplicacion usando dividir y vencer: 15241578750190521
Tiempo de ejecucion en milisegundos:0.0797
BUILD SUCCESSFUL (total time: 22 seconds)
```

Imagen 1. Ejemplo de ejecución del main.

Sanchez Mendez Edmundo Josué
Análisis de Algoritmos

Si vemos la imagen 1 podemos darnos cuenta de cómo hay una gran diferencia entre usar la operación que nos proporciona Java y la clase BigInteger es menos eficiente y por lo tanto rápida que nuestro algoritmo usando dividir y vencer, es importante mencionar que si en alguna multiplicación se genera o ingresa datos mayores al tipo de dato que se solicita el programa fallara o tendríamos que realizar lo anterior, pero nuestro objetivo principal es ver la eficiencia del algoritmo con lo cual se decidió no hacer lo que se menciona.

Podemos concluir que el algoritmo de la multiplicación usando dividir y vencer es más efectivo que usar el algoritmo normal de multiplicación o el que nos proporciona Java o la clase BigInteger, esto se comprueba mediante el tiempo de ejecución que utiliza para realizar la multiplicación y aun a pesar de ser diferentes tipos de datos, mencionar también que la diferencia se nota más cuando el número es de gran tamaño. Finalmente podemos decir que el paradigma dividir y vencer nos sirve para hacer algoritmos más eficientes permitiéndonos el uso de datos muy grandes y no comprometer el tiempo de ejecución.