

*Mercado Salinas Fhernando*

- *Máster Universitario en Sistemas Embebidos*
  - Mondragon Unibertsitatea, Basque Country, Spain
- *Ingeniería en Sistemas Computacionales*
  - Tecnológico de Estudios Superiores de Jocotitlán

---

# Programación Lógica y Funcional

---

v 0.0, Advanced Level



---

# Expresión lambda como retorno de métodos

---

¿Cómo podríamos modificar el ejercicio para poder hacerlo con Sobrecarga de Métodos?



---

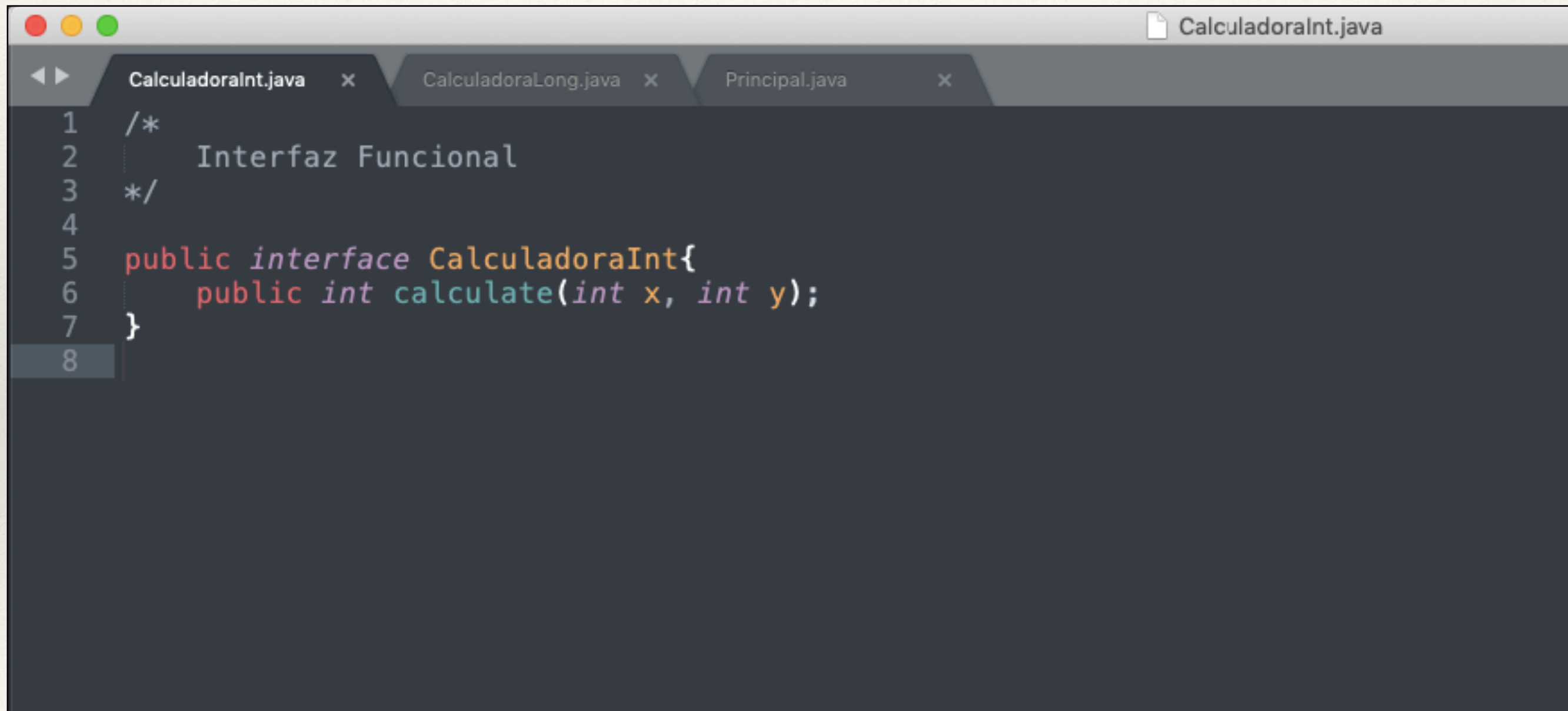
# Expresión lambda como retorno de métodos

---

Solución




# Expresión lambda como retorno de métodos

A screenshot of a Java IDE window titled 'CalculadoraInt.java'. The window has three tabs: 'CalculadoraInt.java', 'CalculadoraLong.java', and 'Principal.java'. The code in the 'CalculadoraInt.java' tab is as follows:

```
1  /*
2     Interfaz Funcional
3  */
4
5  public interface CalculadoraInt{
6      public int calculate(int x, int y);
7  }
8
```

# Expresión lambda como retorno de métodos

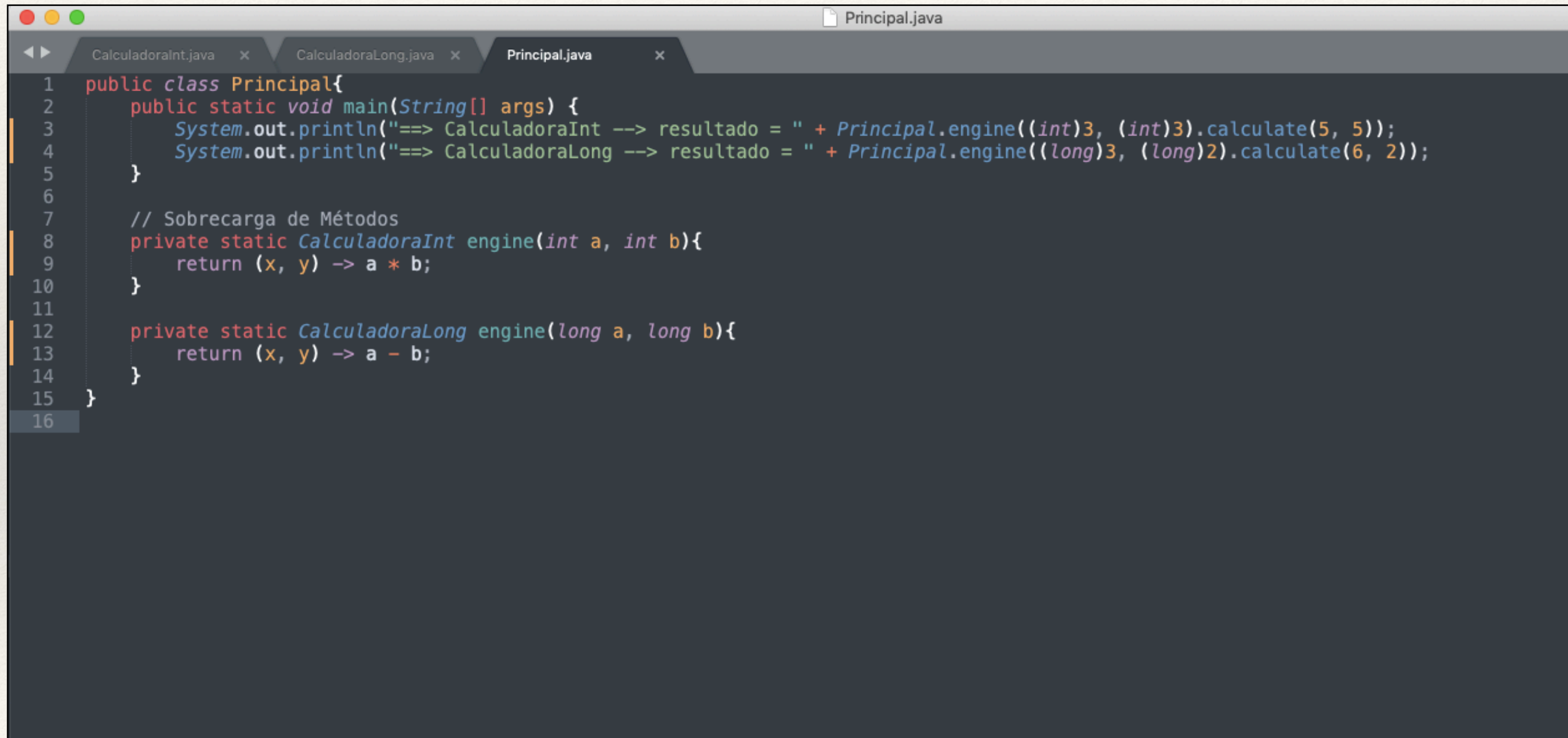


The screenshot shows an IDE window titled 'CalculadoraLong.java'. The code defines a functional interface named 'CalculadoraLong' with a single method 'calculate' that takes two 'long' parameters and returns a 'long' value. The code is as follows:

```
1  /*
2     Interfaz Funcional
3  */
4
5  public interface CalculadoraLong{
6      public long calculate(long x, long y);
7  }
8
```

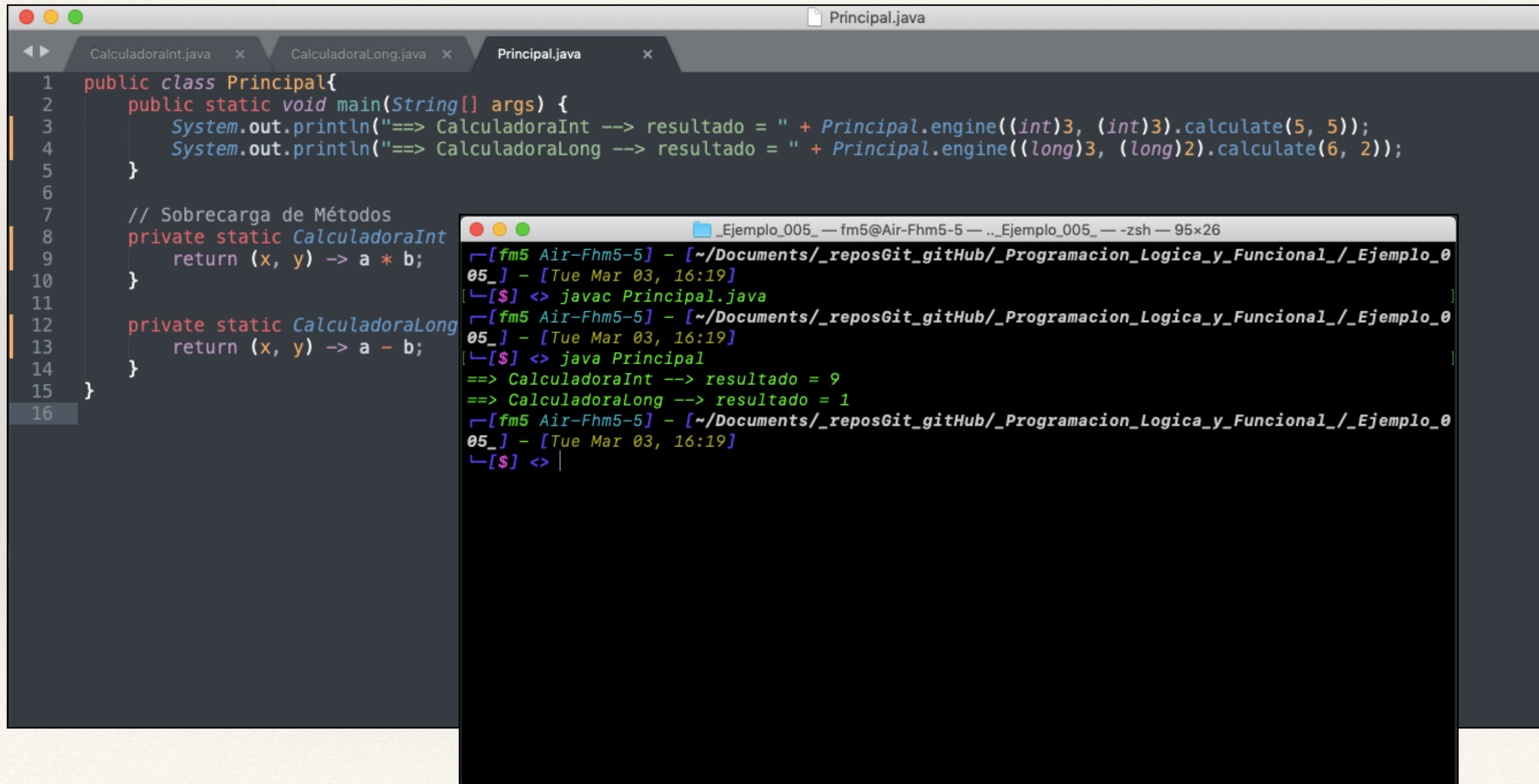


# Expresión lambda como retorno de métodos



```
Principal.java
CalculadoraInt.java x CalculadoraLong.java x Principal.java x
1 public class Principal{
2     public static void main(String[] args) {
3         System.out.println("==> CalculadoraInt --> resultado = " + Principal.engine((int)3, (int)3).calculate(5, 5));
4         System.out.println("==> CalculadoraLong --> resultado = " + Principal.engine((long)3, (long)2).calculate(6, 2));
5     }
6
7     // Sobrecarga de Métodos
8     private static CalculadoraInt engine(int a, int b){
9         return (x, y) -> a * b;
10    }
11
12    private static CalculadoraLong engine(long a, long b){
13        return (x, y) -> a - b;
14    }
15 }
16
```

# Expresión lambda como retorno de métodos



The image shows a Java IDE with three tabs: `CalculadoraInt.java`, `CalculadoraLong.java`, and `Principal.java`. The `Principal.java` tab is active, displaying the following code:

```
1 public class Principal{
2     public static void main(String[] args) {
3         System.out.println("==> CalculadoraInt --> resultado = " + Principal.engine((int)3, (int)3).calculate(5, 5));
4         System.out.println("==> CalculadoraLong --> resultado = " + Principal.engine((long)3, (long)2).calculate(6, 2));
5     }
6
7     // Sobrecarga de Métodos
8     private static CalculadoraInt
9         return (x, y) -> a * b;
10    }
11
12    private static CalculadoraLong
13        return (x, y) -> a - b;
14    }
15 }
16
```

Below the code editor, a terminal window titled `_Ejemplo_005_ — fm5@Air-Fhm5-5 — .._Ejemplo_005_ — -zsh — 95x26` shows the execution of the program:

```
└─[fm5 Air-Fhm5-5] - [~/Documents/_reposGit_github/_Programacion_Logica_y_Funcional/_Ejemplo_005_] - [Tue Mar 03, 16:19]
└─[$] <> javac Principal.java
└─[fm5 Air-Fhm5-5] - [~/Documents/_reposGit_github/_Programacion_Logica_y_Funcional/_Ejemplo_005_] - [Tue Mar 03, 16:19]
└─[$] <> java Principal
==> CalculadoraInt --> resultado = 9
==> CalculadoraLong --> resultado = 1
└─[fm5 Air-Fhm5-5] - [~/Documents/_reposGit_github/_Programacion_Logica_y_Funcional/_Ejemplo_005_] - [Tue Mar 03, 16:19]
└─[$] <> |
```