

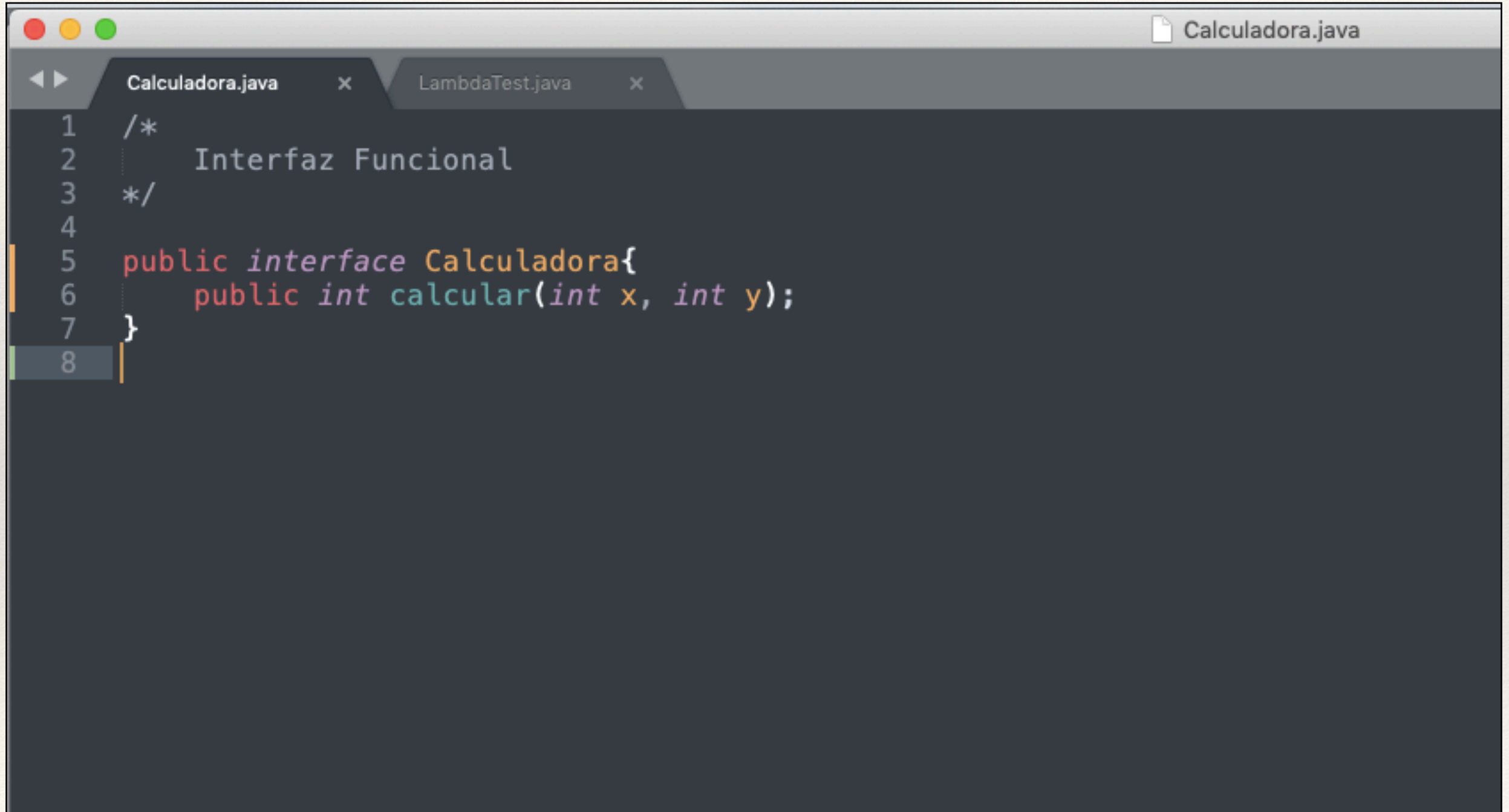
Mercado Salinas Fhernando

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

Programación Lógica y Funcional

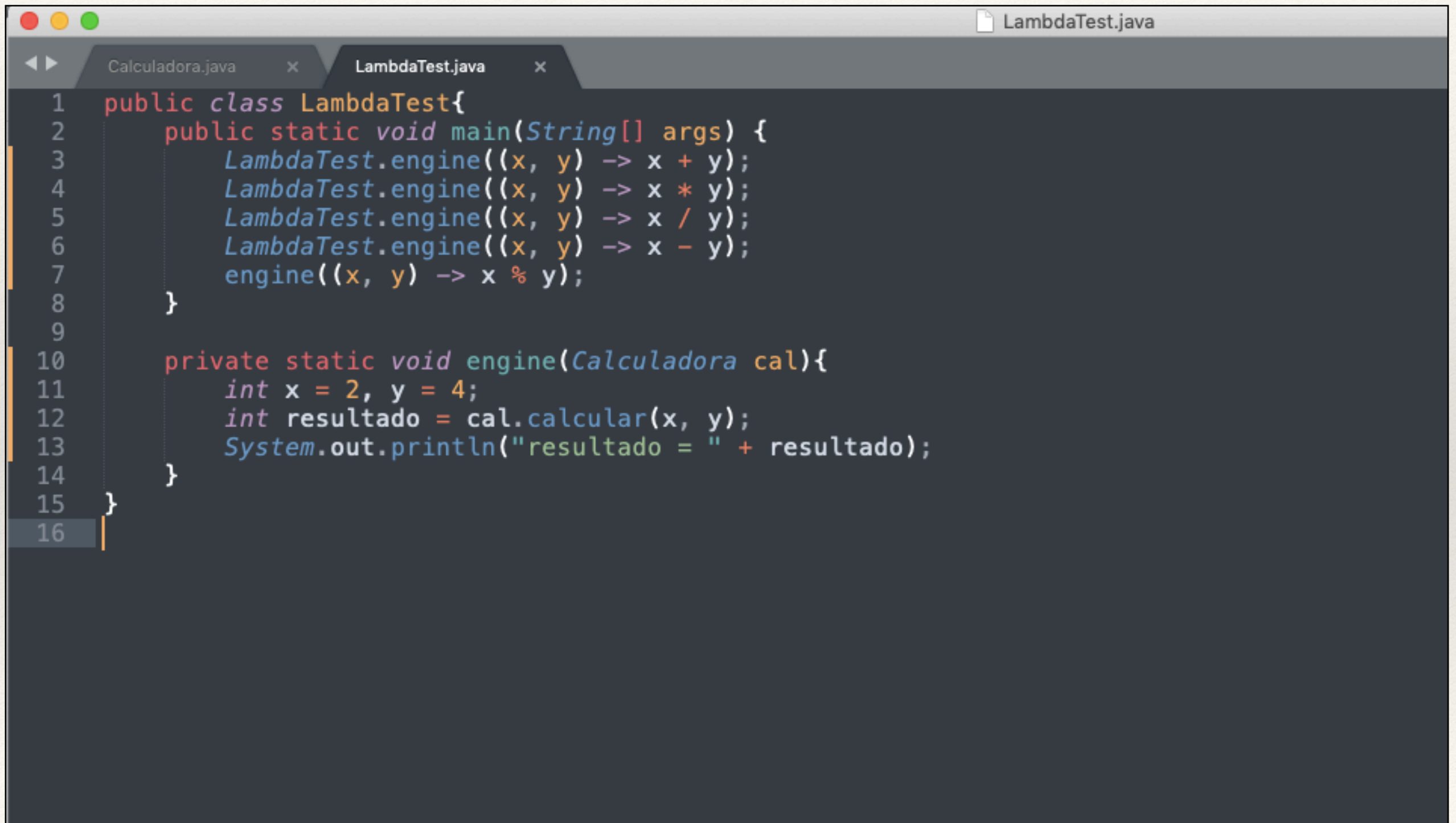
v 0.0, Advanced Level

Expresiones lambda con devolución



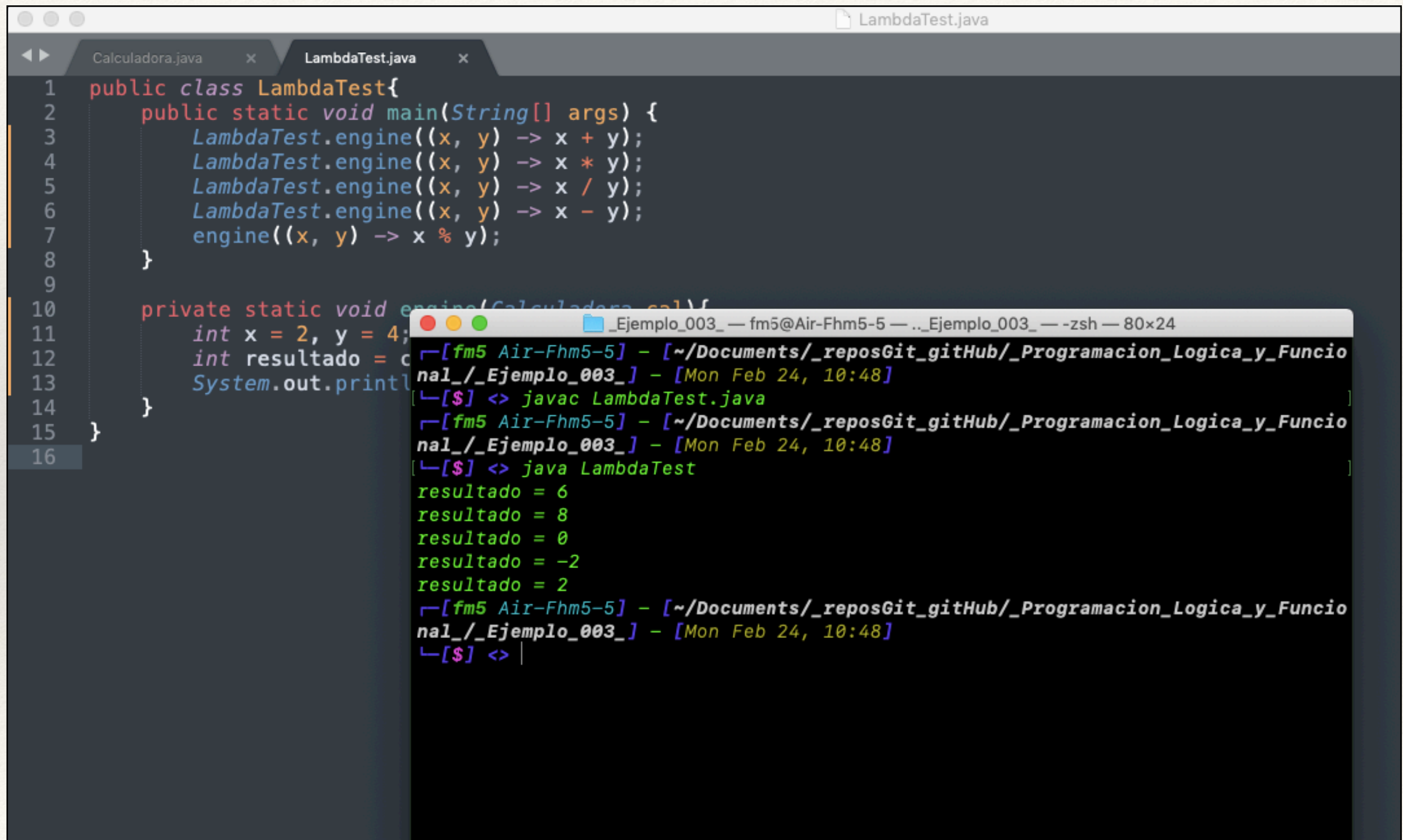
```
1  /*
2  Interfaz Funcional
3  */
4
5  public interface Calculadora{
6      public int calcular(int x, int y);
7  }
8
```

Expresiones lambda con devolución



```
1 public class LambdaTest{
2     public static void main(String[] args) {
3         LambdaTest.engine((x, y) -> x + y);
4         LambdaTest.engine((x, y) -> x * y);
5         LambdaTest.engine((x, y) -> x / y);
6         LambdaTest.engine((x, y) -> x - y);
7         engine((x, y) -> x % y);
8     }
9
10    private static void engine(Calculadora cal){
11        int x = 2, y = 4;
12        int resultado = cal.calcular(x, y);
13        System.out.println("resultado = " + resultado);
14    }
15 }
16
```


Expresiones lambda con devolución



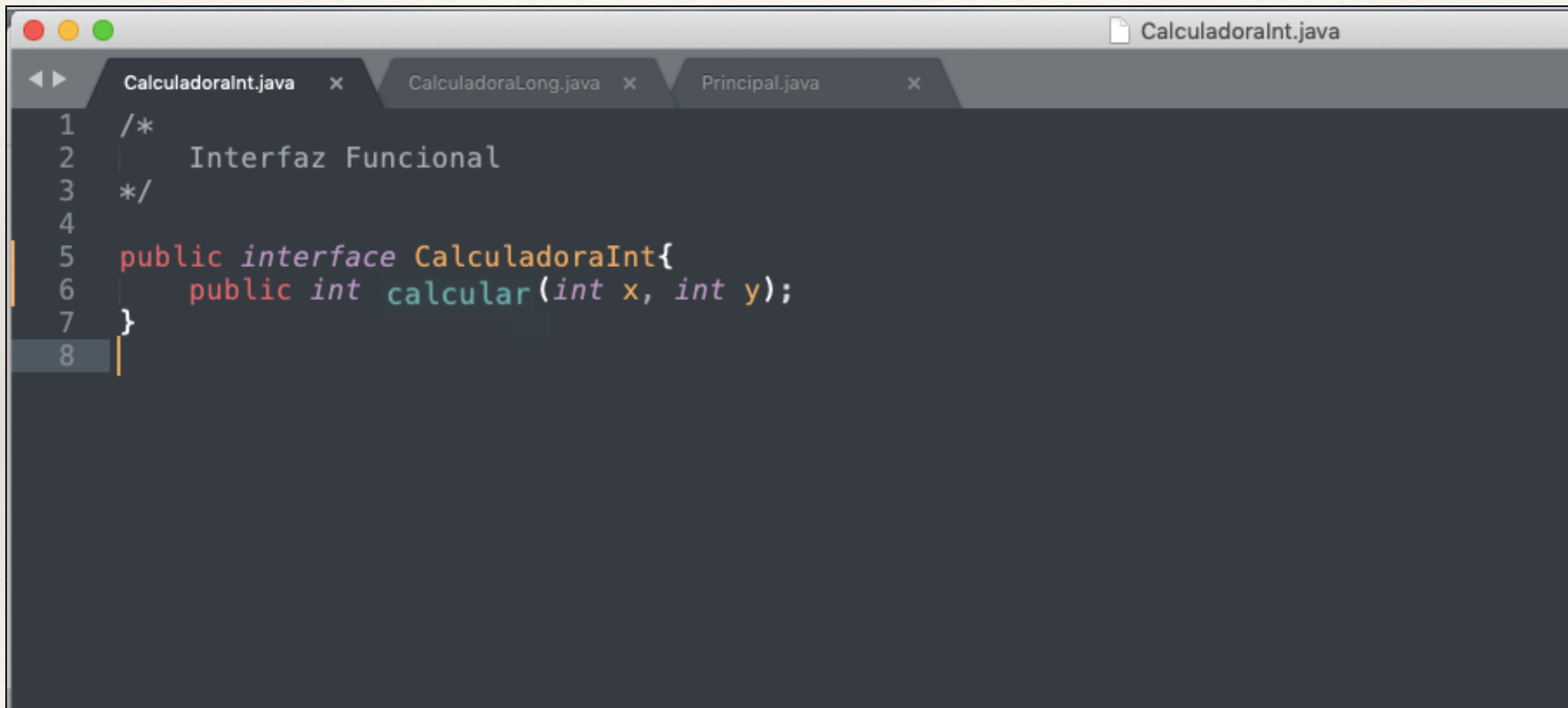
The screenshot displays a Java IDE with two tabs: 'Calculadora.java' and 'LambdaTest.java'. The 'LambdaTest.java' tab is active, showing the following code:

```
1 public class LambdaTest{
2     public static void main(String[] args) {
3         LambdaTest.engine((x, y) -> x + y);
4         LambdaTest.engine((x, y) -> x * y);
5         LambdaTest.engine((x, y) -> x / y);
6         LambdaTest.engine((x, y) -> x - y);
7         engine((x, y) -> x % y);
8     }
9
10    private static void engine(Calculadora cal){
11        int x = 2, y = 4;
12        int resultado = cal.getResultado();
13        System.out.println(resultado);
14    }
15 }
16
```

Below the code editor, a terminal window titled '_Ejemplo_003_ — fm5@Air-Fhm5-5 — ..Ejemplo_003_ — -zsh — 80x24' shows the execution of the program:

```
└─[fm5 Air-Fhm5-5] - [~/Documents/_reposGit_github/_Programacion_Logica_y_Funcio
nal/_Ejemplo_003_] - [Mon Feb 24, 10:48]
└─[$] <> javac LambdaTest.java
└─[fm5 Air-Fhm5-5] - [~/Documents/_reposGit_github/_Programacion_Logica_y_Funcio
nal/_Ejemplo_003_] - [Mon Feb 24, 10:48]
└─[$] <> java LambdaTest
resultado = 6
resultado = 8
resultado = 0
resultado = -2
resultado = 2
└─[fm5 Air-Fhm5-5] - [~/Documents/_reposGit_github/_Programacion_Logica_y_Funcio
nal/_Ejemplo_003_] - [Mon Feb 24, 10:48]
└─[$] <> |
```

Ambigüedad de tipo en expresiones lambda

A screenshot of a Java IDE window titled 'CalculadoraInt.java'. The window contains three tabs: 'CalculadoraInt.java', 'CalculadoraLong.java', and 'Principal.java'. The 'CalculadoraInt.java' tab is active, showing the following code:

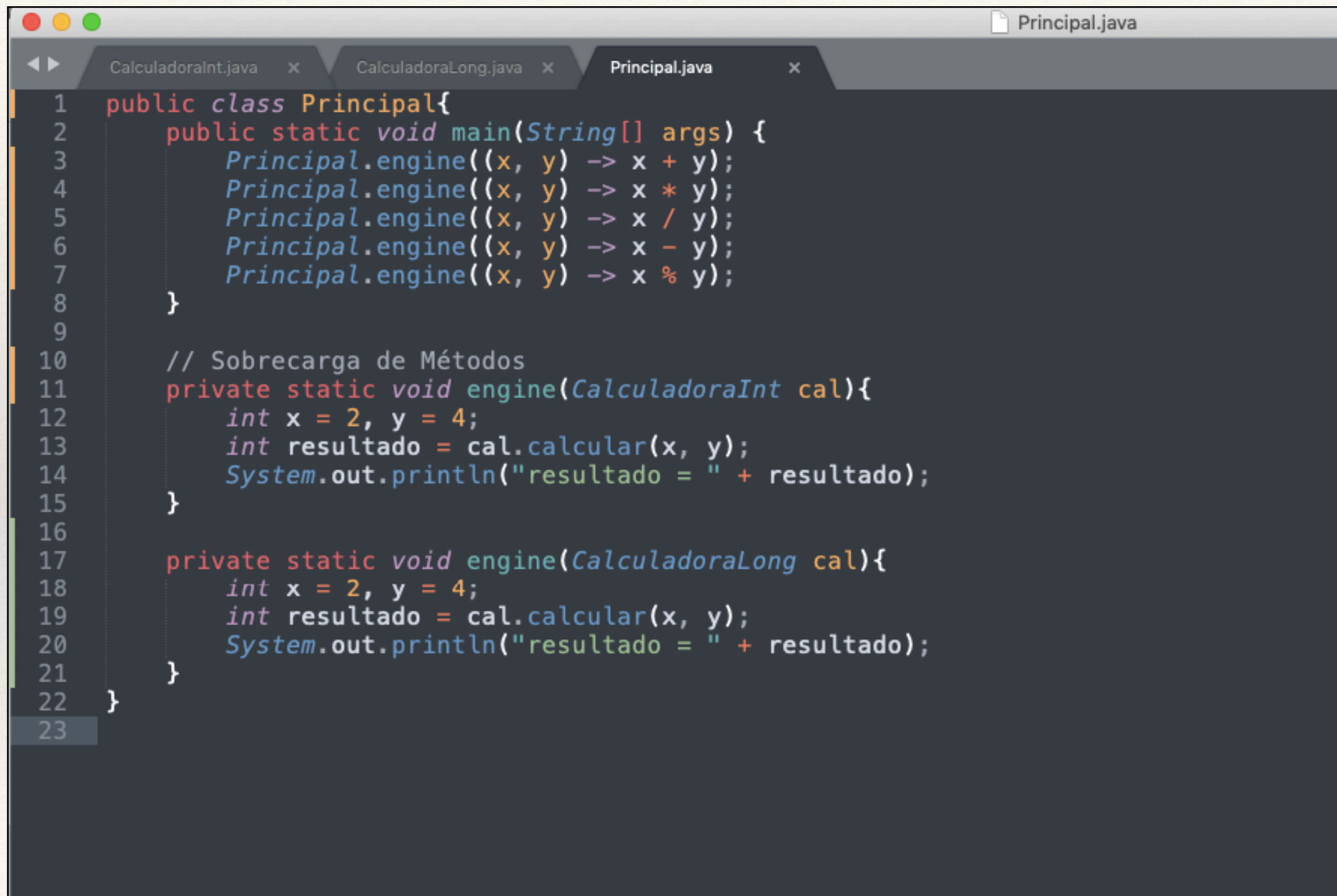
```
1  /*
2     Interfaz Funcional
3  */
4
5  public interface CalculadoraInt{
6      public int calcular(int x, int y);
7  }
8
```


Ambigüedad de tipo en expresiones lambda



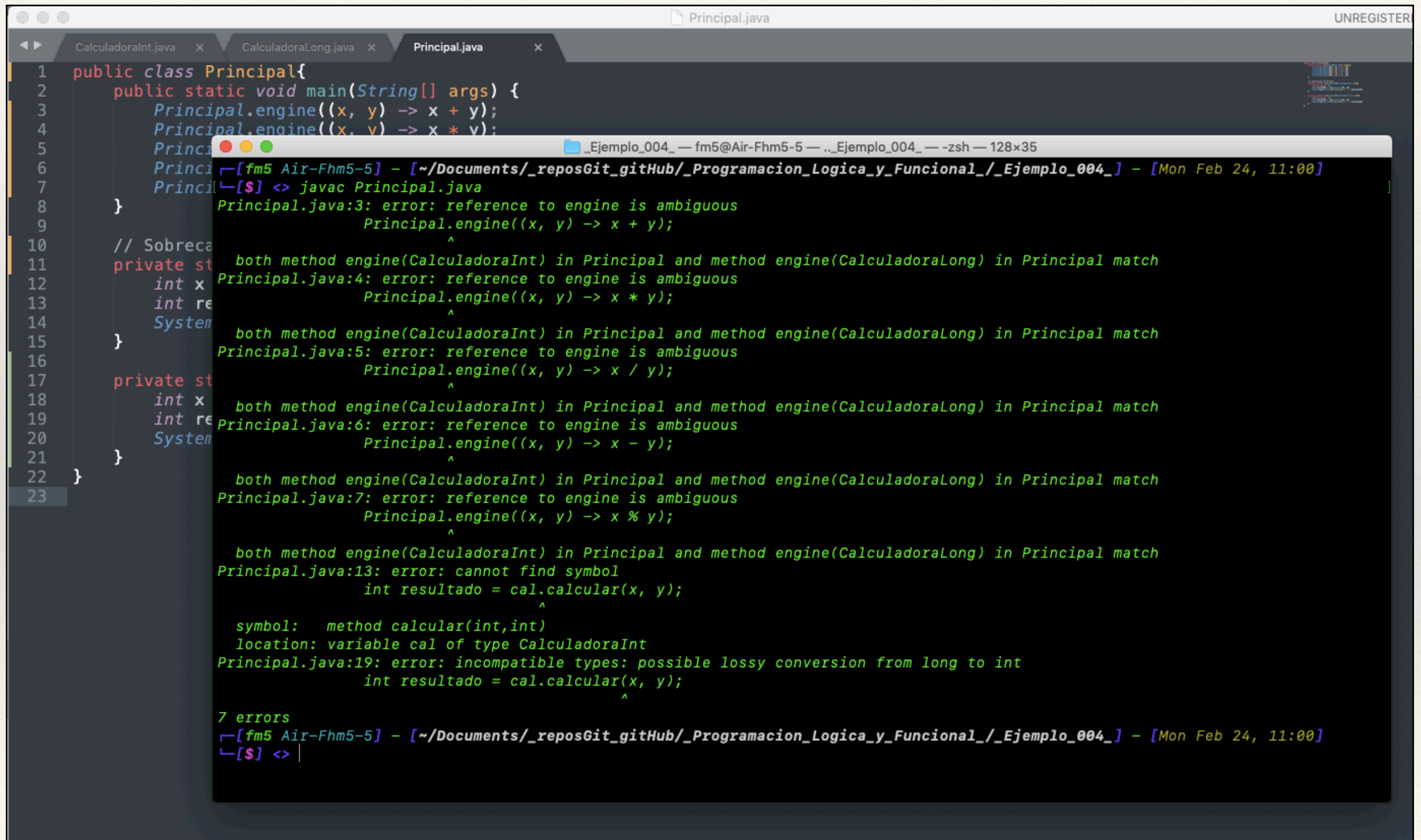
```
1  /*
2  ...   Interfaz Funcional
3  ... */
4
5  public interface CalculadoraLong{
6      public long calcular(long x, long y);
7  }
8
```

Ambigüedad de tipo en expresiones lambda



```
1 public class Principal{
2     public static void main(String[] args) {
3         Principal.engine((x, y) -> x + y);
4         Principal.engine((x, y) -> x * y);
5         Principal.engine((x, y) -> x / y);
6         Principal.engine((x, y) -> x - y);
7         Principal.engine((x, y) -> x % y);
8     }
9
10    // Sobrecarga de Métodos
11    private static void engine(CalculadoraInt cal){
12        int x = 2, y = 4;
13        int resultado = cal.calcular(x, y);
14        System.out.println("resultado = " + resultado);
15    }
16
17    private static void engine(CalculadoraLong cal){
18        int x = 2, y = 4;
19        int resultado = cal.calcular(x, y);
20        System.out.println("resultado = " + resultado);
21    }
22 }
23
```


Ambigüedad de tipo en expresiones lambda



The screenshot shows an 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         Principal.engine((x, y) -> x + y);
4         Principal.engine((x, y) -> x * y);
5         Principal.engine((x, y) -> x / y);
6         Principal.engine((x, y) -> x - y);
7         Principal.engine((x, y) -> x % y);
8     }
9
10    // Sobre carga de metodo
11    private static void engine(CalculadoraInt cal) {
12        int x = cal.getX();
13        int y = cal.getY();
14        System.out.println("CalculadoraInt: " + x + " + " + y + " = " + cal.calcular(x, y));
15    }
16
17    private static void engine(CalculadoraLong cal) {
18        long x = cal.getX();
19        long y = cal.getY();
20        System.out.println("CalculadoraLong: " + x + " + " + y + " = " + cal.calcular(x, y));
21    }
22 }
23
```

Below the code, a terminal window shows the output of `javac Principal.java` with the following errors:

```
Principal.java:3: error: reference to engine is ambiguous
    Principal.engine((x, y) -> x + y);
                  ^
    both method engine(CalculadoraInt) in Principal and method engine(CalculadoraLong) in Principal match
Principal.java:4: error: reference to engine is ambiguous
    Principal.engine((x, y) -> x * y);
                  ^
    both method engine(CalculadoraInt) in Principal and method engine(CalculadoraLong) in Principal match
Principal.java:5: error: reference to engine is ambiguous
    Principal.engine((x, y) -> x / y);
                  ^
    both method engine(CalculadoraInt) in Principal and method engine(CalculadoraLong) in Principal match
Principal.java:6: error: reference to engine is ambiguous
    Principal.engine((x, y) -> x - y);
                  ^
    both method engine(CalculadoraInt) in Principal and method engine(CalculadoraLong) in Principal match
Principal.java:7: error: reference to engine is ambiguous
    Principal.engine((x, y) -> x % y);
                  ^
    both method engine(CalculadoraInt) in Principal and method engine(CalculadoraLong) in Principal match
Principal.java:13: error: cannot find symbol
    int resultado = cal.calcular(x, y);
                      ^
    symbol:   method calcular(int,int)
    location: variable cal of type CalculadoraInt
Principal.java:19: error: incompatible types: possible lossy conversion from long to int
    int resultado = cal.calcular(x, y);
                      ^
7 errors
```

Ambigüedad de tipo en expresiones lambda

¿Cómo resolvemos el
problema (error)
anterior?