



Processamento da Informação

Modularização

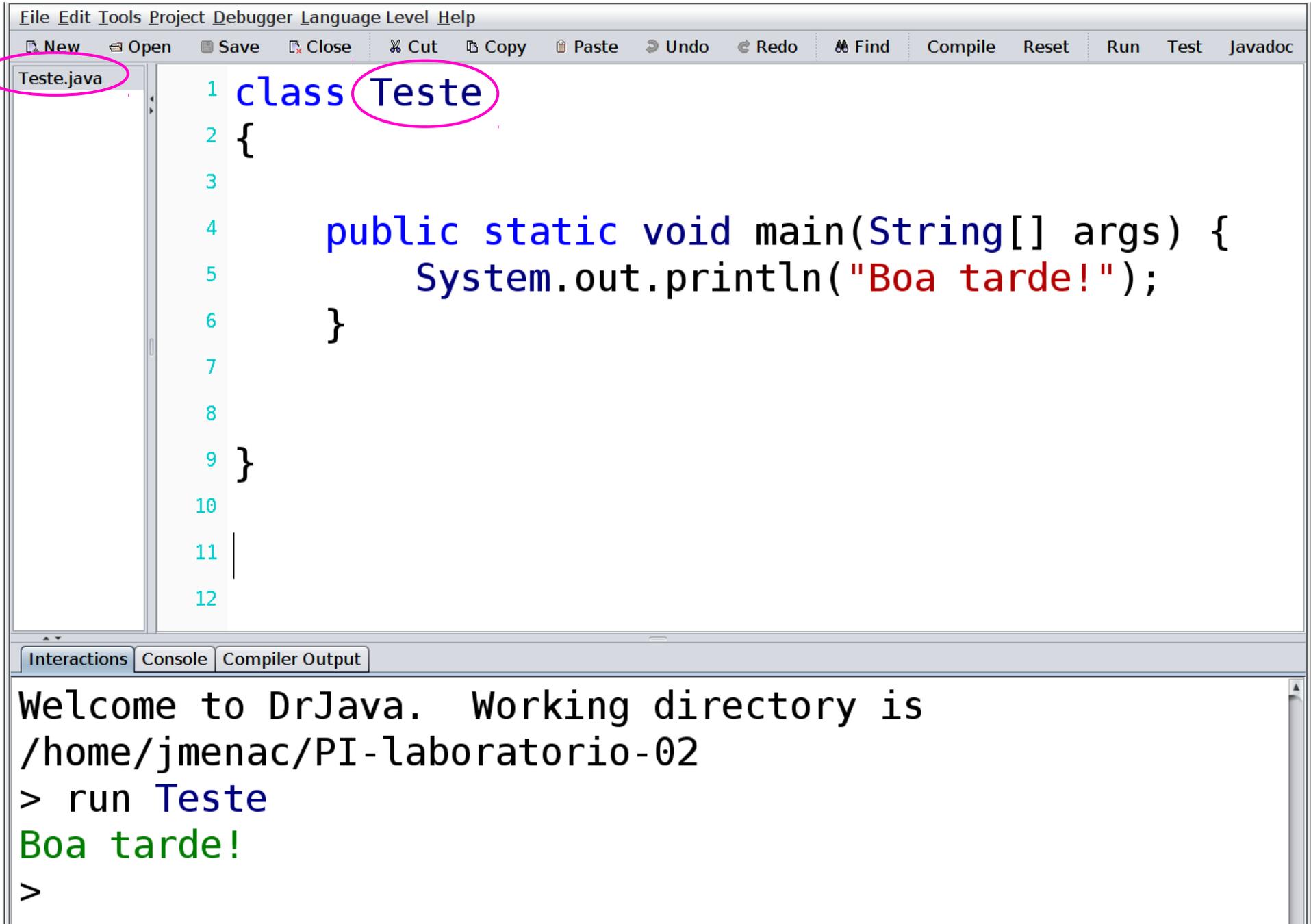
Prof. Jesús P. Mena-Chalco
CMCC/UFABC

Q2/2018



Imprimindo uma mensagem

(1) Exemplo básico



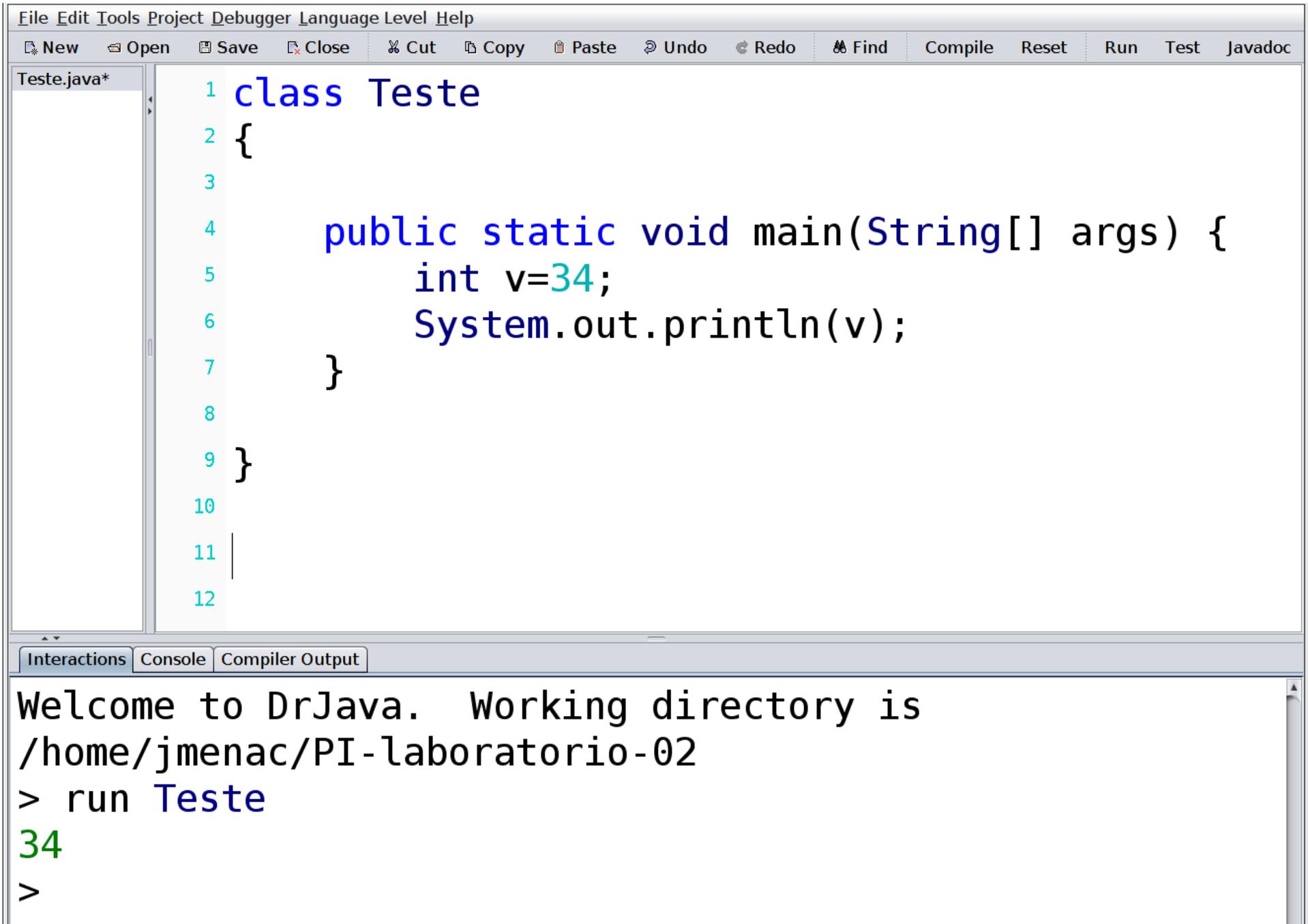
The image shows a screenshot of the DrJava IDE. The main editor window displays the following Java code:

```
1 class Teste
2 {
3
4     public static void main(String[] args) {
5         System.out.println("Boa tarde!");
6     }
7
8
9 }
10
11
12
```

The file name 'Teste.java' is visible in the left sidebar. The 'Teste' class name in the code is circled in pink. The IDE's menu bar includes File, Edit, Tools, Project, Debugger, Language, Level, and Help. The toolbar contains buttons for New, Open, Save, Close, Cut, Copy, Paste, Undo, Redo, Find, Compile, Reset, Run, Test, and Javadoc. The bottom panel shows the 'Console' tab with the following output:

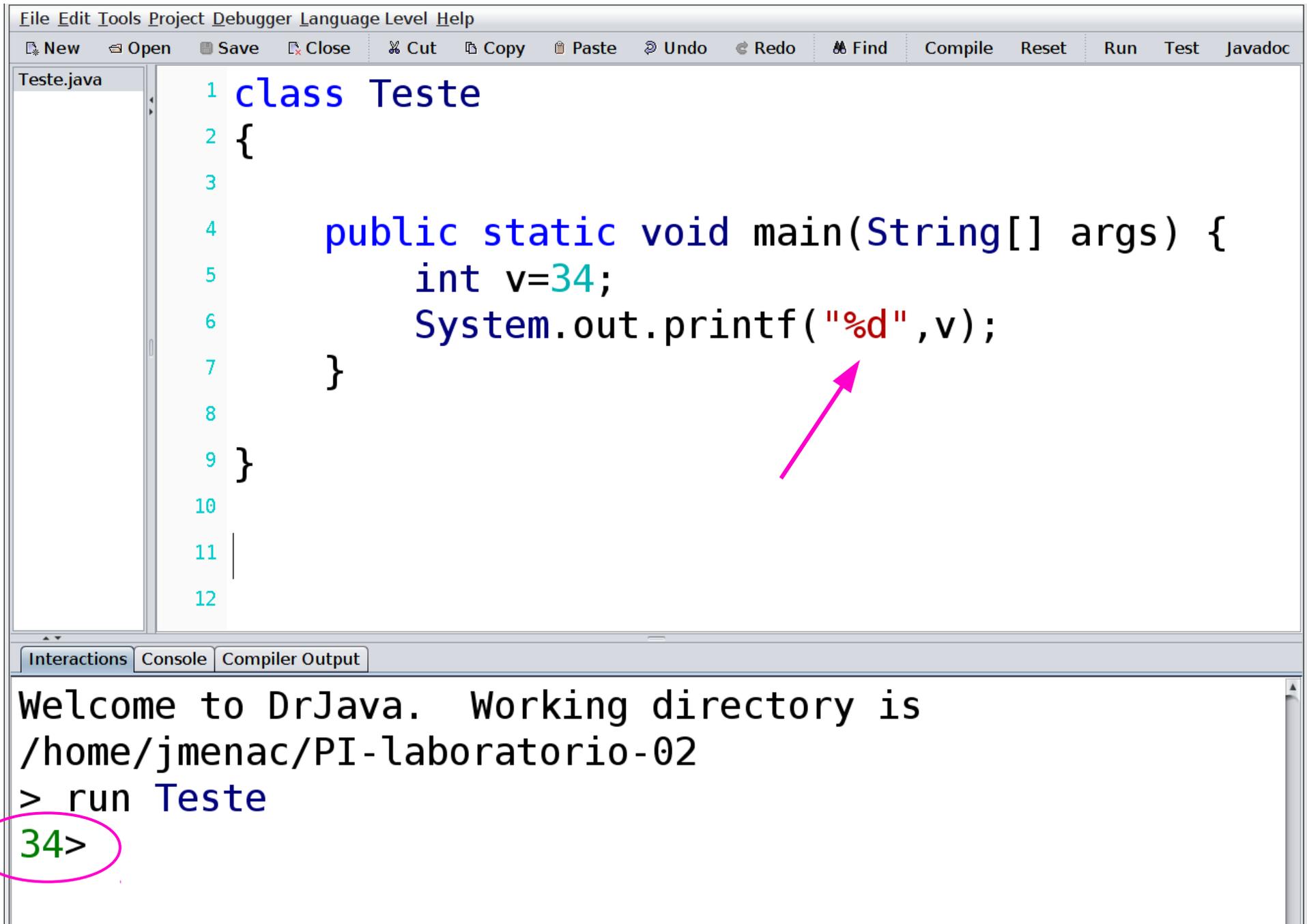
```
Welcome to DrJava. Working directory is
/home/jmenac/PI-laboratorio-02
> run Teste
Boa tarde!
>
```

(2) Imprimindo: println



```
File Edit Tools Project Debugger Language Level Help
New Open Save Close Cut Copy Paste Undo Redo Find Compile Reset Run Test Javadoc
Teste.java*
1 class Teste
2 {
3
4     public static void main(String[] args) {
5         int v=34;
6         System.out.println(v);
7     }
8
9 }
10
11
12
Interactions Console Compiler Output
Welcome to DrJava. Working directory is
/home/jmenac/PI-laboratorio-02
> run Teste
34
>
```

(3) Imprimindo com formato: printf



```
File Edit Tools Project Debugger Language Level Help
New Open Save Close Cut Copy Paste Undo Redo Find Compile Reset Run Test Javadoc
Teste.java
1 class Teste
2 {
3
4     public static void main(String[] args) {
5         int v=34;
6         System.out.printf("%d",v);
7     }
8
9 }
10
11
12
Interactions Console Compiler Output
Welcome to DrJava. Working directory is
/home/jmenac/PI-laboratorio-02
> run Teste
34>
```

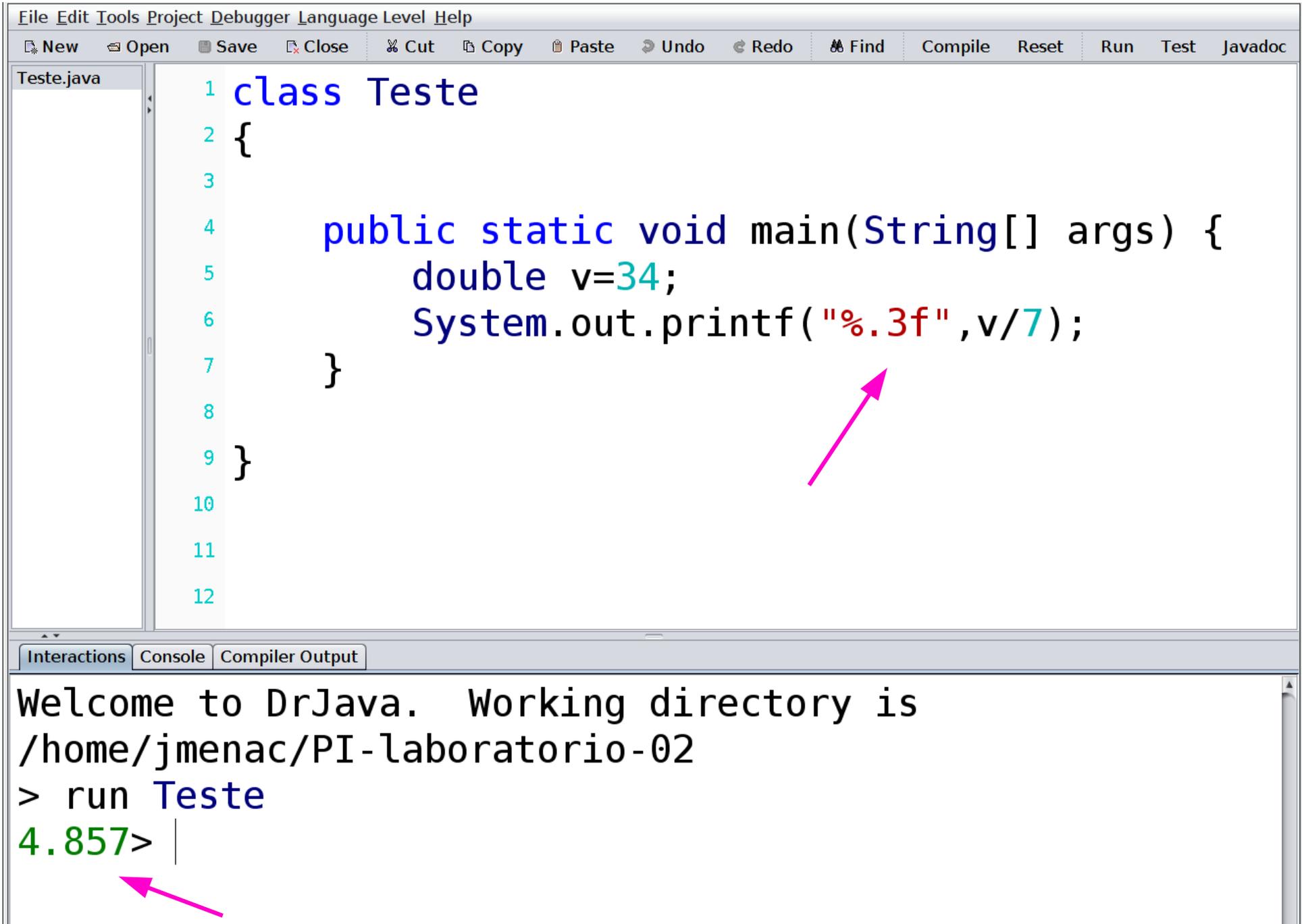
(4) Imprimindo com formato: printf

```
File Edit Tools Project Debugger Language Level Help
New Open Save Close Cut Copy Paste Undo Redo Find Compile Reset Run Test Javadoc
Teste.java
1 class Teste
2 {
3
4     public static void main(String[] args) {
5         double v=34;
6         System.out.printf("%f",v);
7     }
8
9 }
10
11
12
```

Interactions Console Compiler Output

```
Welcome to DrJava. Working directory is
/home/jmenac/PI-laboratorio-02
> run Teste
34.000000>
```

(5) Imprimindo com formato: printf



```
File Edit Tools Project Debugger Language Level Help
New Open Save Close Cut Copy Paste Undo Redo Find Compile Reset Run Test Javadoc
Teste.java
1 class Teste
2 {
3
4     public static void main(String[] args) {
5         double v=34;
6         System.out.printf("%.3f",v/7);
7     }
8
9 }
10
11
12
Interactions Console Compiler Output
Welcome to DrJava. Working directory is
/home/jmenac/PI-laboratorio-02
> run Teste
4.857>
```

The image shows a screenshot of the DrJava IDE. The main editor window displays a Java class named 'Teste'. The code is as follows:

```
1 class Teste
2 {
3
4     public static void main(String[] args) {
5         double v=34;
6         System.out.printf("%.3f",v/7);
7     }
8
9 }
```

A pink arrow points from the console output '4.857' to the format string '%.3f' in the code. The console output is shown in the 'Console' tab at the bottom of the IDE. The output is '4.857' followed by a prompt character '>'. A pink arrow points from the console output to the format string in the code.

(6) Imprimindo com formato: printf

```
File Edit Tools Project Debugger Language Level Help
New Open Save Close Cut Copy Paste Undo Redo Find Compile Reset Run Test Javadoc
Teste.java
1 class Teste
2 {
3
4     public static void main(String[] args) {
5         double v=34;
6         System.out.printf("v = %.3f\n",v/7);
7     }
8
9 }
10
11
12
```

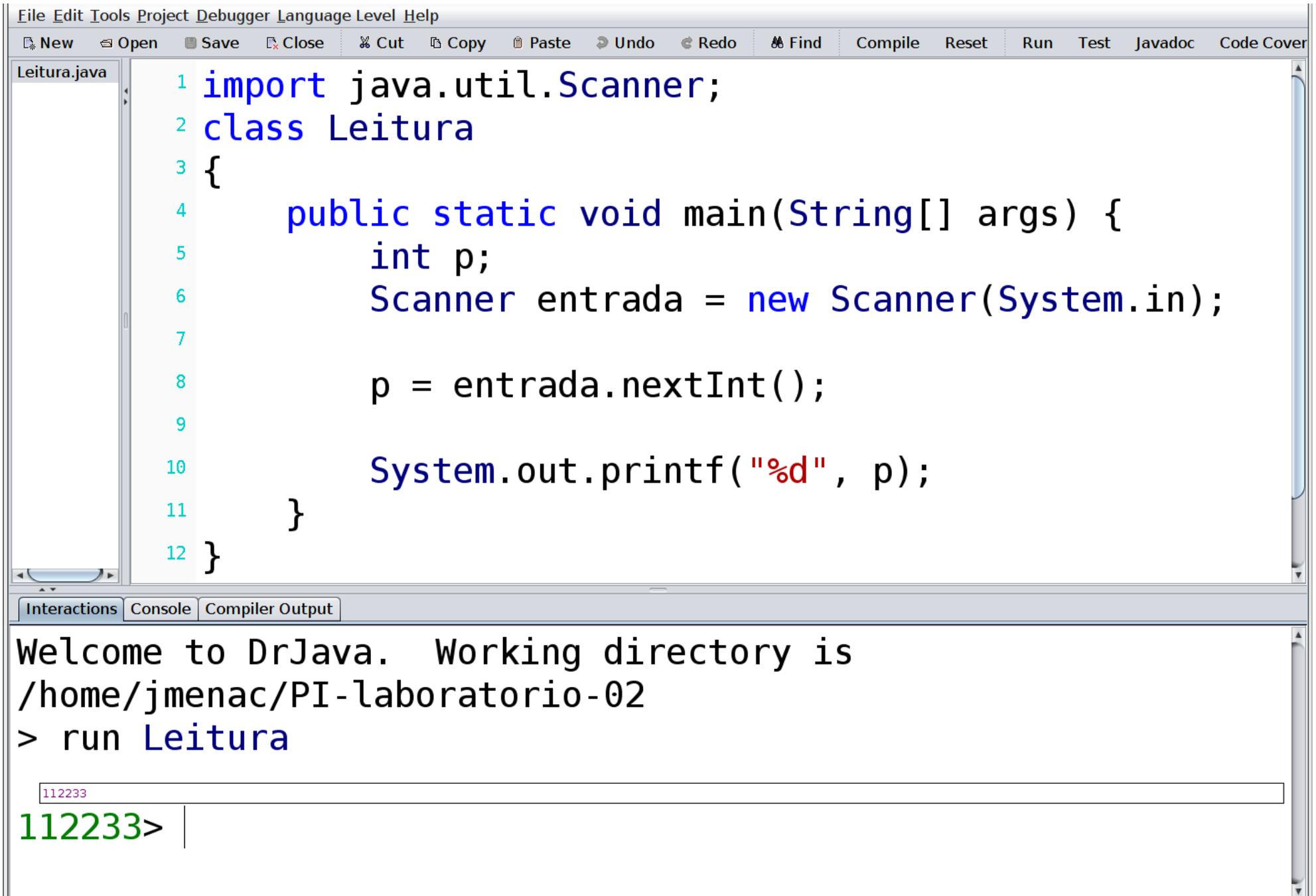
Interactions Console Compiler Output

```
Welcome to DrJava. Working directory is
/home/jmenac/PI-laboratorio-02
> run Teste
v = 4.857
> |
```



Leitura de números (por teclado)

Leitura de um número inteiro



```
File Edit Tools Project Debugger Language Level Help
New Open Save Close Cut Copy Paste Undo Redo Find Compile Reset Run Test Javadoc Code Cover
Leitura.java
1 import java.util.Scanner;
2 class Leitura
3 {
4     public static void main(String[] args) {
5         int p;
6         Scanner entrada = new Scanner(System.in);
7
8         p = entrada.nextInt();
9
10        System.out.printf("%d", p);
11    }
12 }
```

Interactions Console Compiler Output

Welcome to DrJava. Working directory is
/home/jmenac/PI-laboratorio-02
> run Leitura

112233

112233>

Problema 1001 – Lista 1 – URI

URI Online Judge | 1001

Extremamente Básico

Adaptado por Neilor Tonin, URI  Brasil

Timelimit: 1

Leia 2 valores inteiros e armazene-os nas variáveis **A** e **B**. Efetue a soma de **A** e **B** atribuindo o seu resultado na variável **X**. Imprima **X** conforme exemplo apresentado abaixo. Não apresente mensagem alguma além daquilo que está sendo especificado e não esqueça de imprimir o fim de linha após o resultado, caso contrário, você receberá "*Presentation Error*".

Entrada

A entrada contém 2 valores inteiros.

Saída

Imprima a mensagem "X = " (letra X maiúscula) seguido pelo valor da variável **X** e pelo final de linha. Cuide para que tenha um espaço antes e depois do sinal de igualdade, conforme o exemplo abaixo.

| Exemplos de Entrada | Exemplos de Saída |
|---------------------|-------------------|
| 10 9 | X = 19 |
| -10 4 | X = -6 |
| 15 -7 | X = 8 |

Problema 1001 – Lista 1 – URI

CÓDIGO FONTE

```
1 import java.io.IOException;
2 import java.util.Scanner;
3 /**
4  * IMPORTANT:
5  *     O nome da classe deve ser "Main" para que a sua solução execute
6  *     Class name must be "Main" for your solution to execute
7  *     El nombre de la clase debe ser "Main" para que su solución ejecutar
8  */
9 public class Main {
10
11     public static void main(String[] args) throws IOException {
12         int a, b;
13         Scanner entrada = new Scanner(System.in);
14
15         a = entrada.nextInt();
16         b = entrada.nextInt();
17
18         System.out.printf("X = %d\n", a+b);
19
20     }
21
22 }
```

Leitura de 2 números inteiros

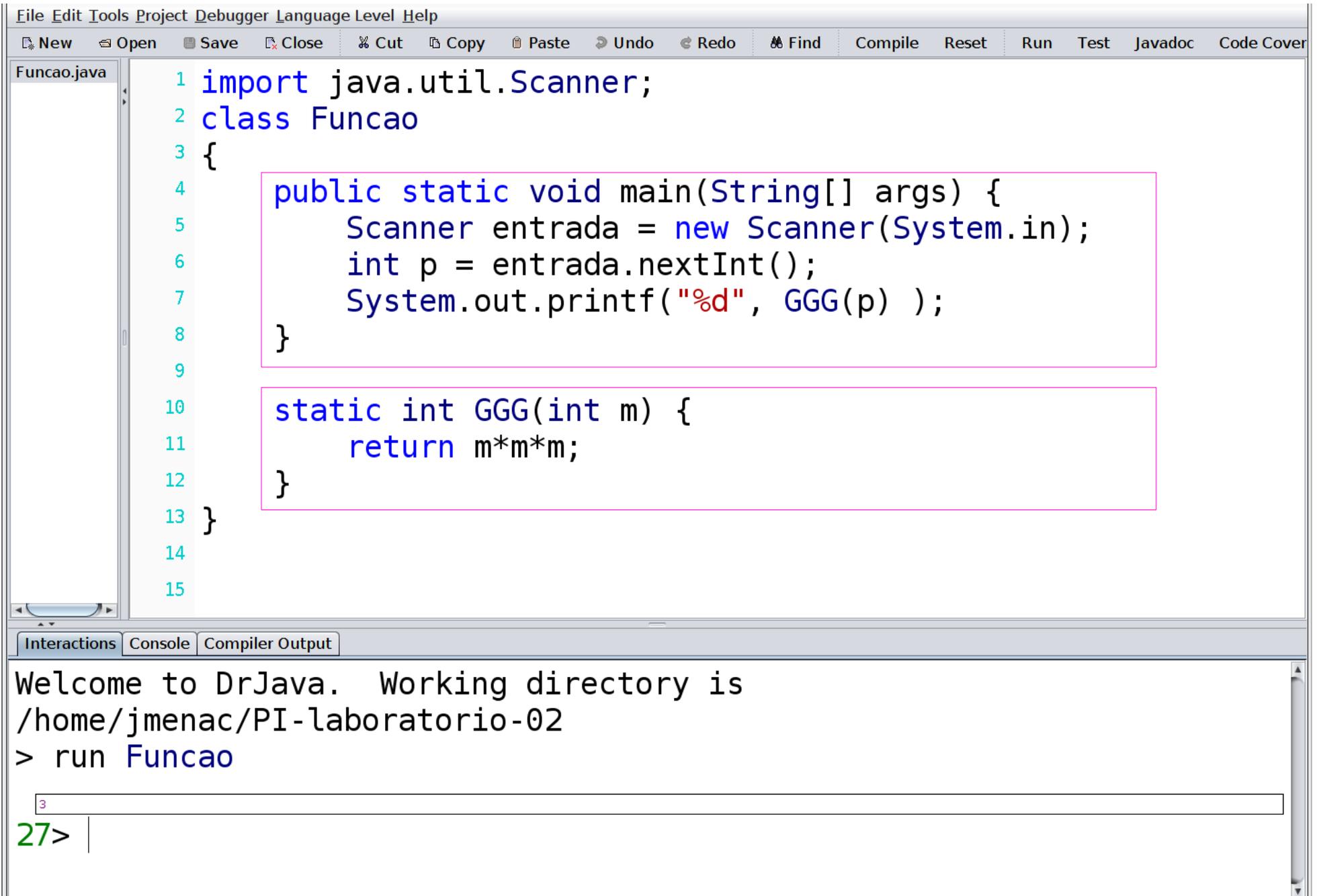


Funções (métodos)

(1) Função eco

```
File Edit Tools Project Debugger Language Level Help
New Open Save Close Cut Copy Paste Undo Redo Find Compile Reset Run Test Javadoc Code Cover
Funcao.java
1 import java.util.Scanner;
2 class Funcao
3 {
4     public static void main(String[] args) {
5         Scanner entrada = new Scanner(System.in);
6         int p = entrada.nextInt();
7         FFF(p);
8     }
9
10    static void FFF(int m) {
11        System.out.printf("%d", m);
12    }
13 }
14
15
Interactions Console Compiler Output
Welcome to DrJava. Working directory is
/home/jmenac/PI-laboratorio-02
> run Funcao
234
234> |
```

(2) Função m^3



```
File Edit Tools Project Debugger Language Level Help
New Open Save Close Cut Copy Paste Undo Redo Find Compile Reset Run Test Javadoc Code Cover
Funcao.java
1 import java.util.Scanner;
2 class Funcao
3 {
4     public static void main(String[] args) {
5         Scanner entrada = new Scanner(System.in);
6         int p = entrada.nextInt();
7         System.out.printf("%d", GGG(p) );
8     }
9
10    static int GGG(int m) {
11        return m*m*m;
12    }
13 }
14
15

Interactions Console Compiler Output
Welcome to DrJava. Working directory is
/home/jmenac/PI-laboratorio-02
> run Funcao
3
27> |
```

(3) Função m^3 , m^6 ou m^9 ?

The image shows a screenshot of a Java IDE (DrJava) with a code editor and a console window. The code editor displays the following Java code:

```
1 import java.util.Scanner;
2 class Funcao
3 {
4     public static void main(String[] args) {
5         Scanner entrada = new Scanner(System.in);
6         int p = entrada.nextInt();
7         System.out.printf("%d", GGG(GGG(p)) );
8     }
9
10    static int GGG(int m) {
11        return m*m*m;
12    }
13 }
14
15
```

A yellow callout bubble points to the `GGG` method, containing the text: "Número de multiplicações? 4 para calcular m^9 ".

The console window shows the following output:

```
Welcome to DrJava. Working directory is
/home/jmenac/PI-laboratorio-02
> run Funcao
2
512>
```



Fibonacci

Números de Fibonacci

Os números de Fibonacci estão relacionados com a razão aurea .

O i-ésimo número pode ser aproximado pela seguinte equação (formula explícita):

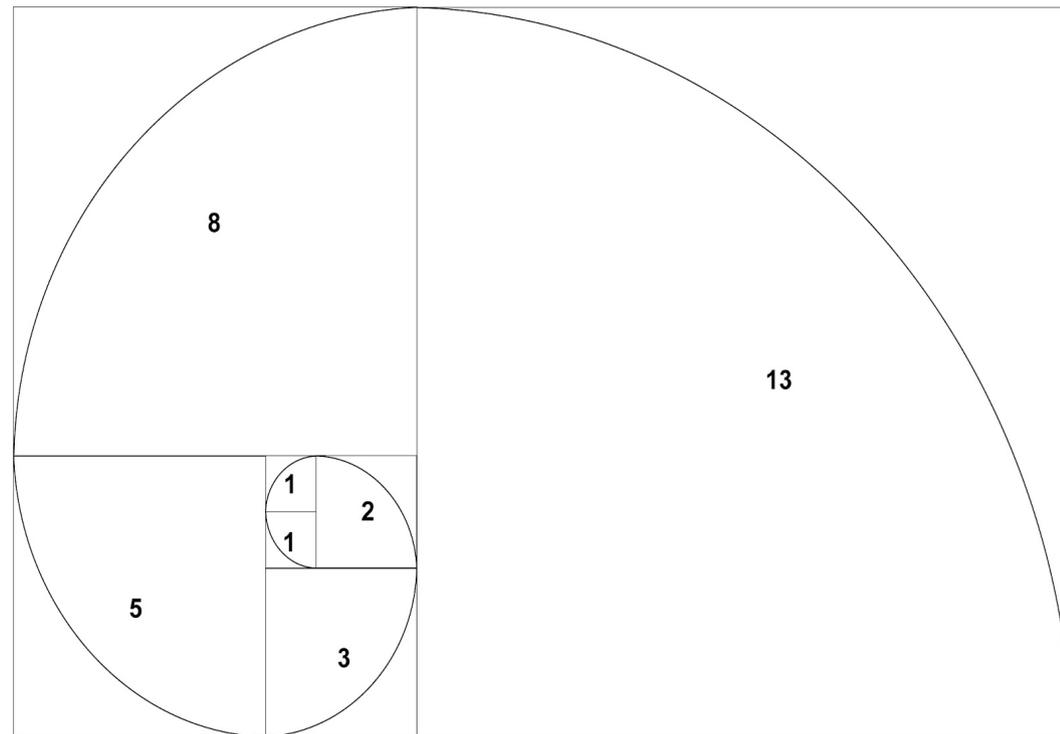
$$F_i = \left\lfloor \frac{\left(\frac{1+\sqrt{5}}{2}\right)^i - \left(\frac{1-\sqrt{5}}{2}\right)^i}{\sqrt{5}} \right\rfloor$$

Crie uma função/método em Java que receba um número inteiro i, e devolva Fi.

Assinatura: `static double iessimoTermo(int i)`

Números de Fibonacci

| F_0 | F_1 | F_2 | F_3 | F_4 | F_5 | F_6 | F_7 | F_8 | F_9 | F_{10} | F_{11} | F_{12} | F_{13} | F_{14} | F_{15} | F_{16} | F_{17} | F_{18} | F_{19} | F_{20} |
|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|----------|----------|----------|----------|----------|----------|----------|----------|----------|----------|----------|
| 0 | 1 | 1 | 2 | 3 | 5 | 8 | 13 | 21 | 34 | 55 | 89 | 144 | 233 | 377 | 610 | 987 | 1597 | 2584 | 4181 | 6765 |





Exemplo de função chamada dentro de uma outra função

O que imprime esse programa?

```
1 class Funcao2
2 {
3     public static void main(String[] args) {
4         System.out.printf("%d", PP(5) );
5     }
6
7     static int PP(int m) {
8         return QQ(m) + QQ(m+1) + QQ(m+2);
9     }
10
11    static int QQ(int s) {
12        return 2*s;
13    }
14 }
15
```

O que imprime esse programa?

```
1 class Funcao3
2 {
3     public static void main(String[] args) {
4         System.out.printf("%d", TT(5) );
5     }
6
7     static int TT(int w) {
8         return 2*w + TT(w);
9     }
10 }
```



Duas aproximações de Pi

(1) Aproximação de M. Schneider

3

Crie uma função/método em Java que devolva a aproximação de Pi desenvolvido por M. Schneider.

$$\pi \approx \sqrt{7 + \sqrt{6 + \sqrt{5}}}$$

Assinatura: `static double pi1()`

A aproximação é boa para quantos dígitos?

(2) Aproximação de S. Irvine

8

Crie uma função/método em Java que devolva a aproximação de Pi desenvolvido por S. Irvine.

$$\pi \approx \sqrt{\sqrt{3^4 + \frac{19^2}{78 - 56}}}$$

Assinatura: `static double pi2()`

A aproximação é boa para quantos dígitos?

Palavras “reservadas” que não podem ser usadas como nomes de variáveis

| Java Keywords | | | | |
|--|-----------|------------|--------------|-----------|
| abstract | boolean | break | byte | case |
| catch | char | class | continue | default |
| do | double | else | extends | false |
| final | finally | float | for | if |
| implements | import | instanceof | int | interface |
| long | native | new | null | package |
| private | protected | public | return | short |
| static | super | switch | synchronized | this |
| throw | throws | transient | true | try |
| void | volatile | while | | |
| <i>Keywords that are reserved but not used by Java</i> | | | | |
| const | goto | | | |