Programação I

Prof. Andre Noel

Introdução

- Qual desses dois equipamentos é mais útil para o nosso dia a dia?

Fonte: Freeimages (1415242).

Fonte: Pixabay (767781).

Introdução

- Qual desses dois equipamentos é mais útil para o nosso dia a dia?

Fonte: Freeimages (1415242)

Fonte: Pixabay (767781)

Qual é a principal diferença entre eles?

Introdução

- Um computador é um equipamento programável.
- *“Se você não sabe programar, você usa um computador da mesma forma que um forno de microondas, apenas com as funções que alguém já programou para você” (Sábio desconhecido).*

O que é programação?

- “É um processo que leva uma formulação original de um problema computacional a programas de computador executáveis” (Wikipedia - EN).

O caminho da programação

- Lógica de programação
- ↓
- Algoritmos
- ↓
- Linguagem de programação
- ↓
- Programação

O caminho da programação

- Lógica de programação
- ↓

Fonte: Vida de Programador
<https://vidadeprogramador.com.br/2012/03/22/ilogica-de-programacao/>

Programação

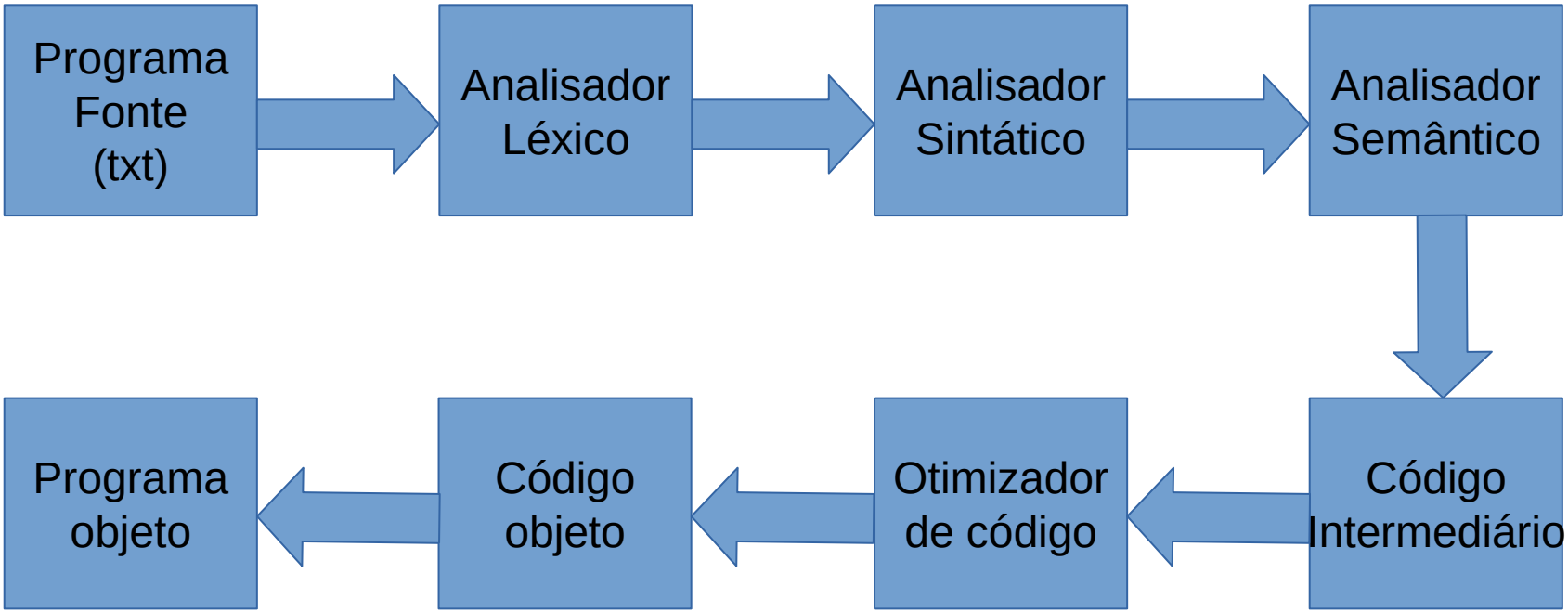
- Programação sequencial (estruturada)

Programação

- Programação sequencial (estruturada)
- Linguagens compiladas x interpretadas

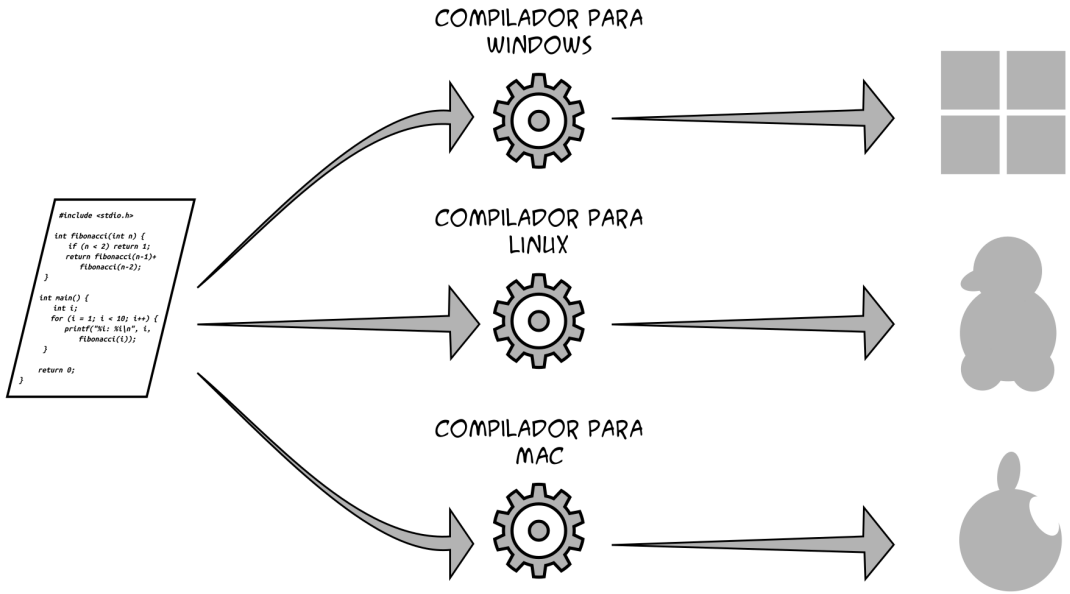
Programação

- Programação sequencial (estruturada)
- Linguagens compiladas x interpretadas
- Compilação

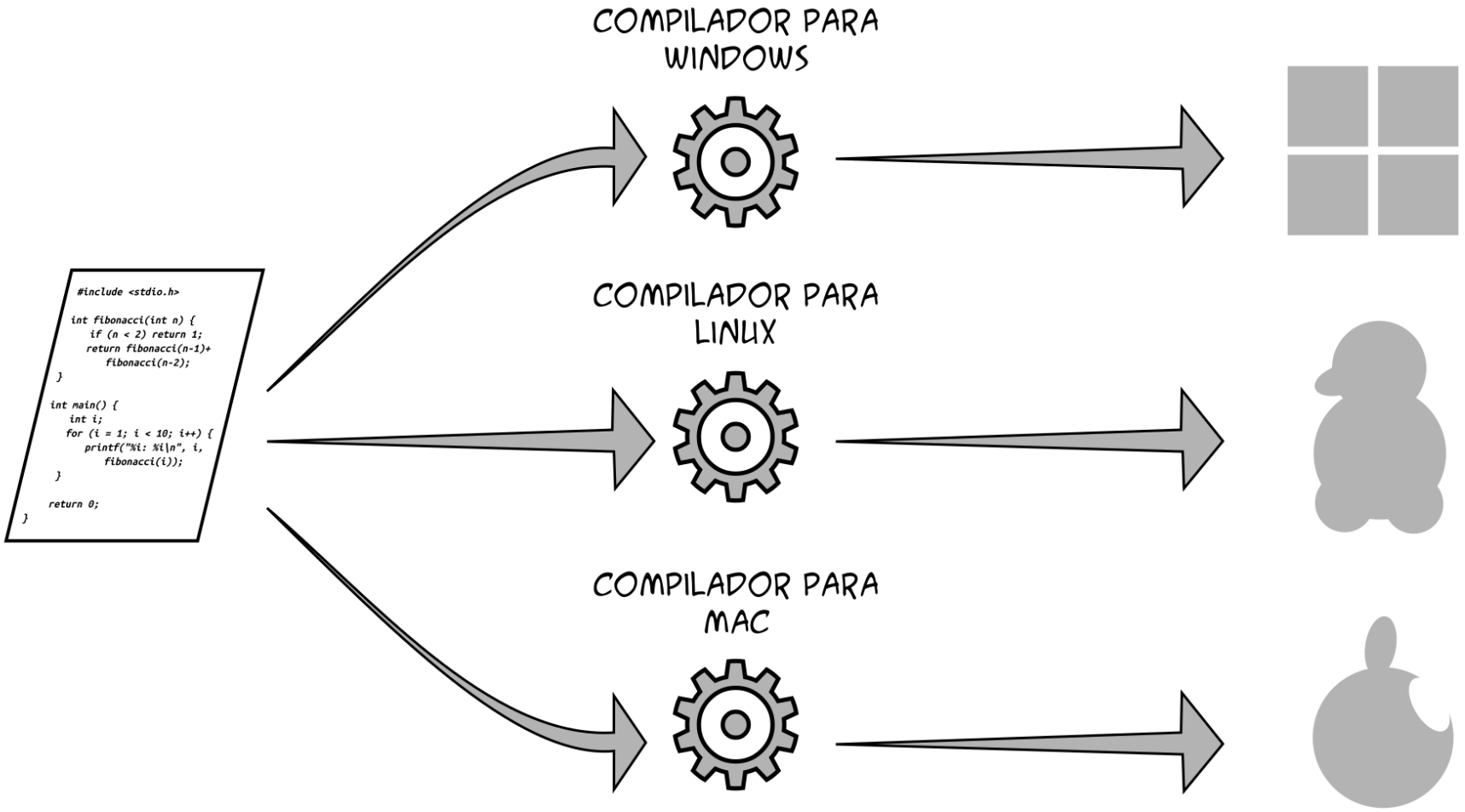


Programação

- Programação sequencial (estruturada)
- Linguagens compiladas x interpretadas
- Compilação
- Dependente do sistema operacional



Fonte: O autor.



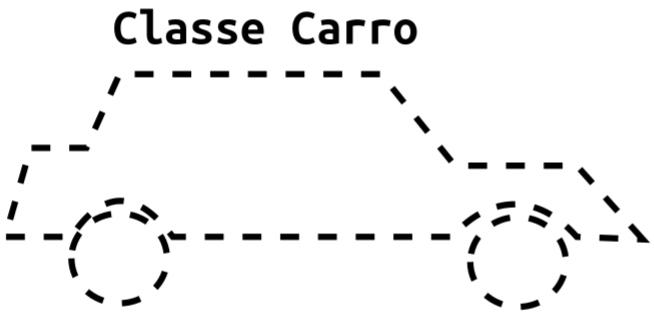
Fonte: O autor.

Programação Orientada a Objetos

Programação Orientada a Objetos

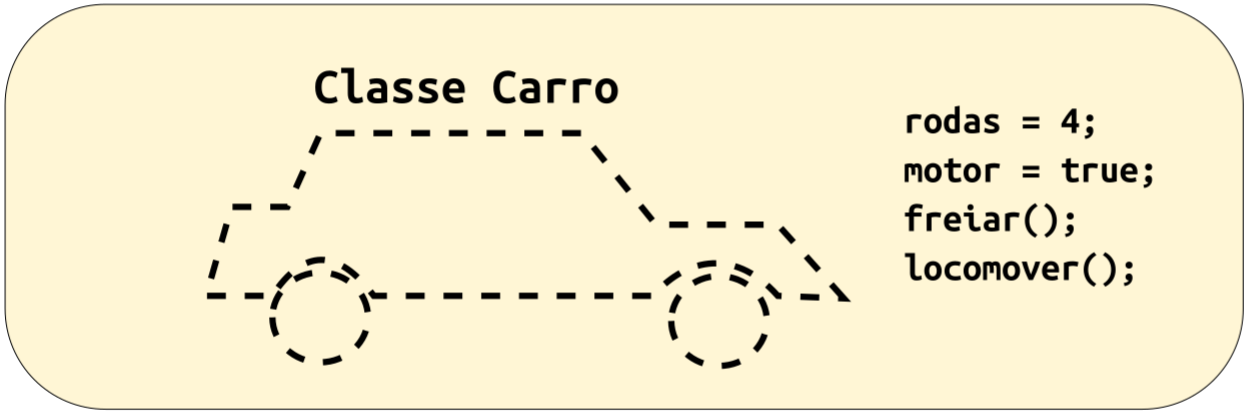
- Objetos.
- Classes de objetos.

Programação Orientada a Objetos



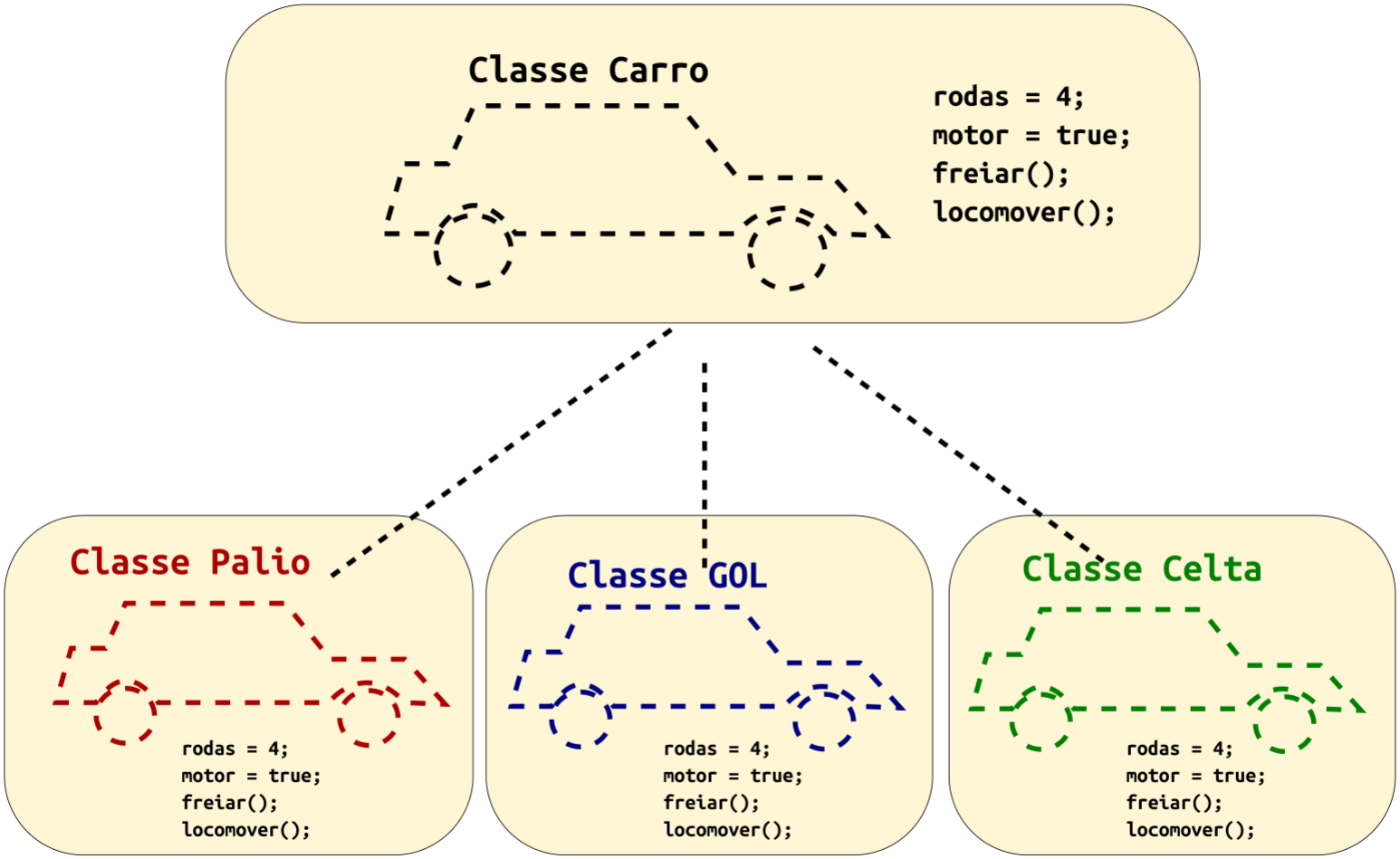
Fonte: O autor.

Programação Orientada a Objetos



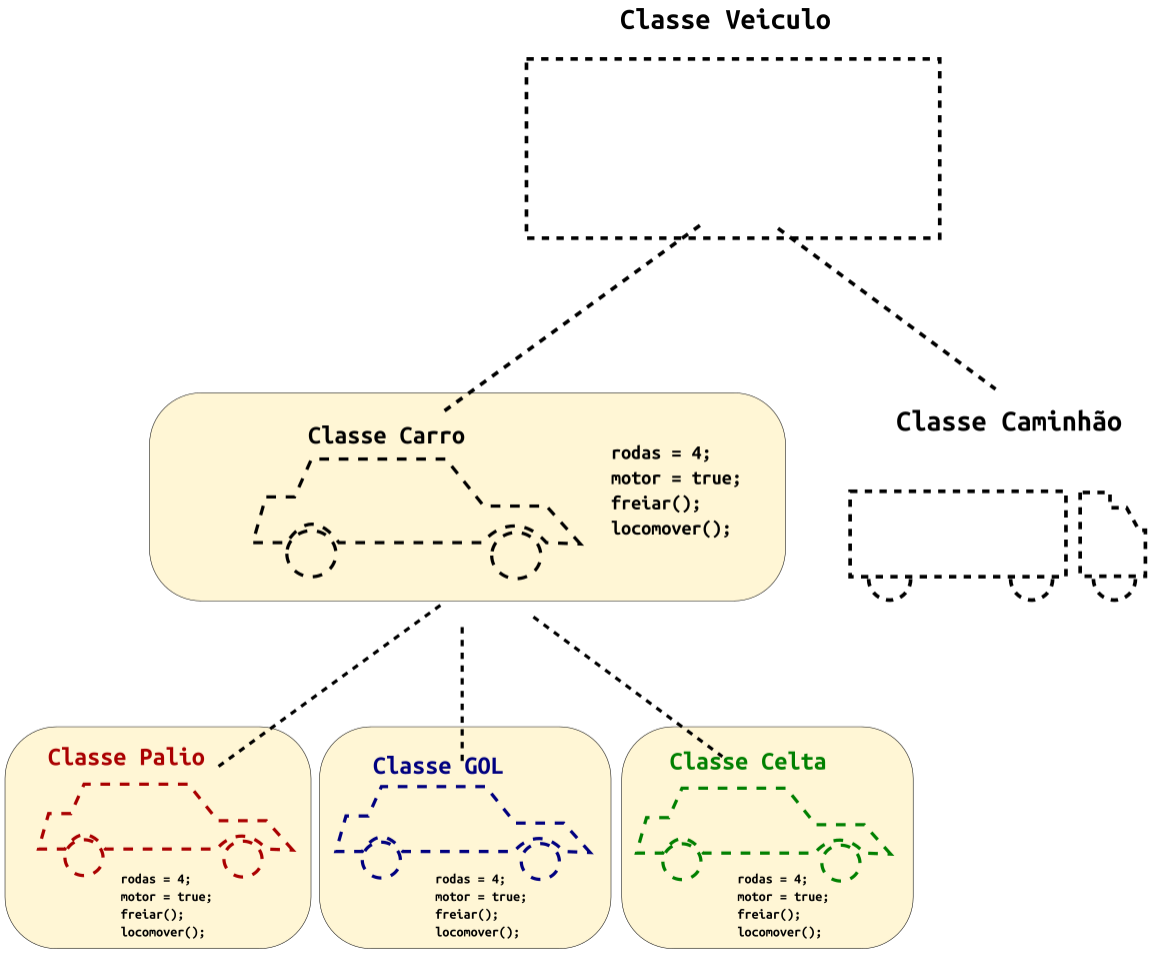
Fonte: O autor.

Programação Orientada a Objetos



Fonte: O autor.

Programação Orientada a Objetos



Fonte: O autor.

Fonte: Pixabay (1580595).

A tecnologia JAVA

- A linguagem mais todas as bibliotecas disponíveis para download e uso.

A tecnologia JAVA

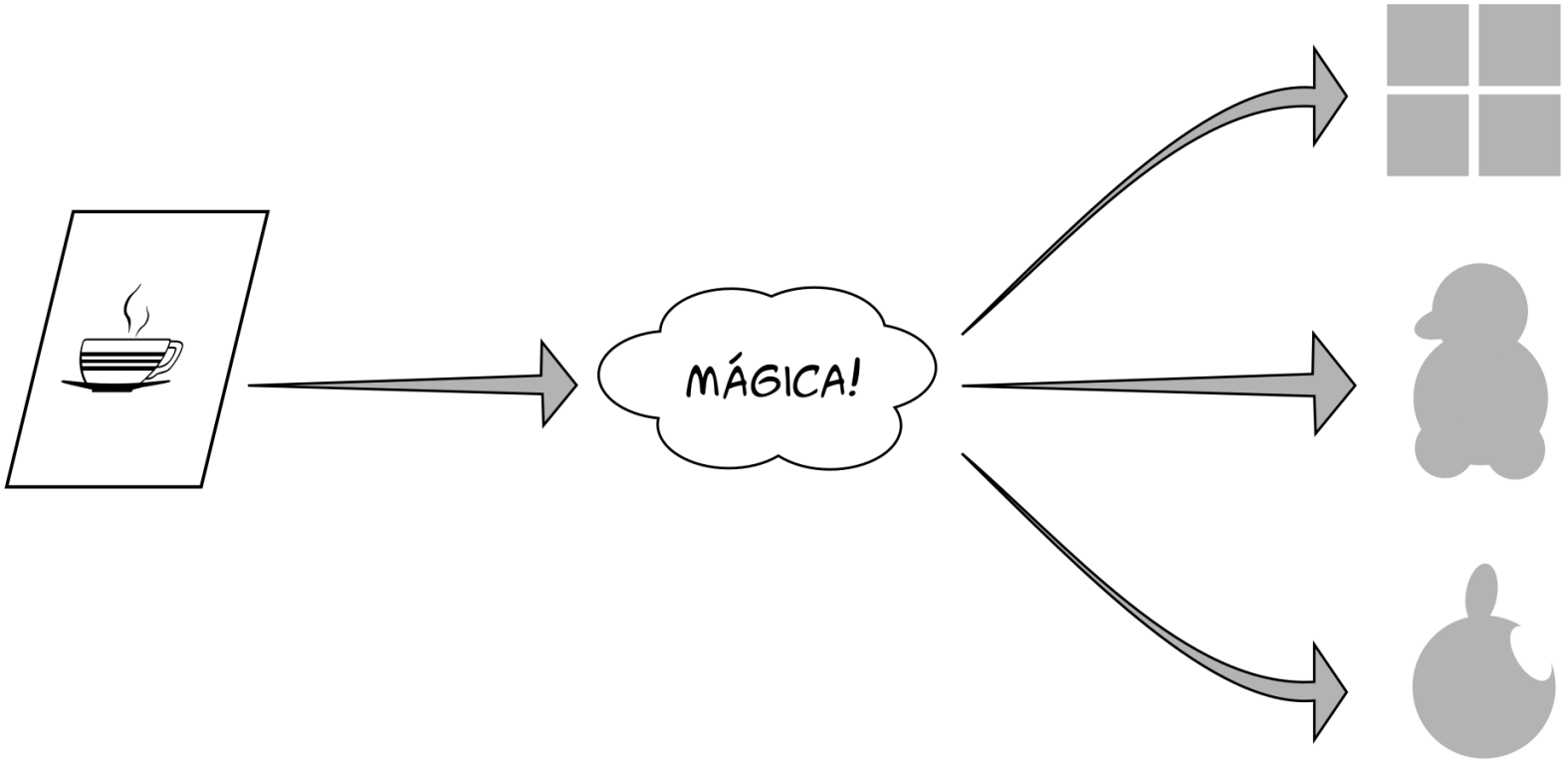
- A linguagem mais todas as bibliotecas disponíveis para download e uso
- Plataforma Java:
  - Java Card;
  - Java ME;
  - Java SE;
  - Java EE.

O que tem de tão especial no Java?

- Multiplataforma!

O que tem de tão especial no Java?

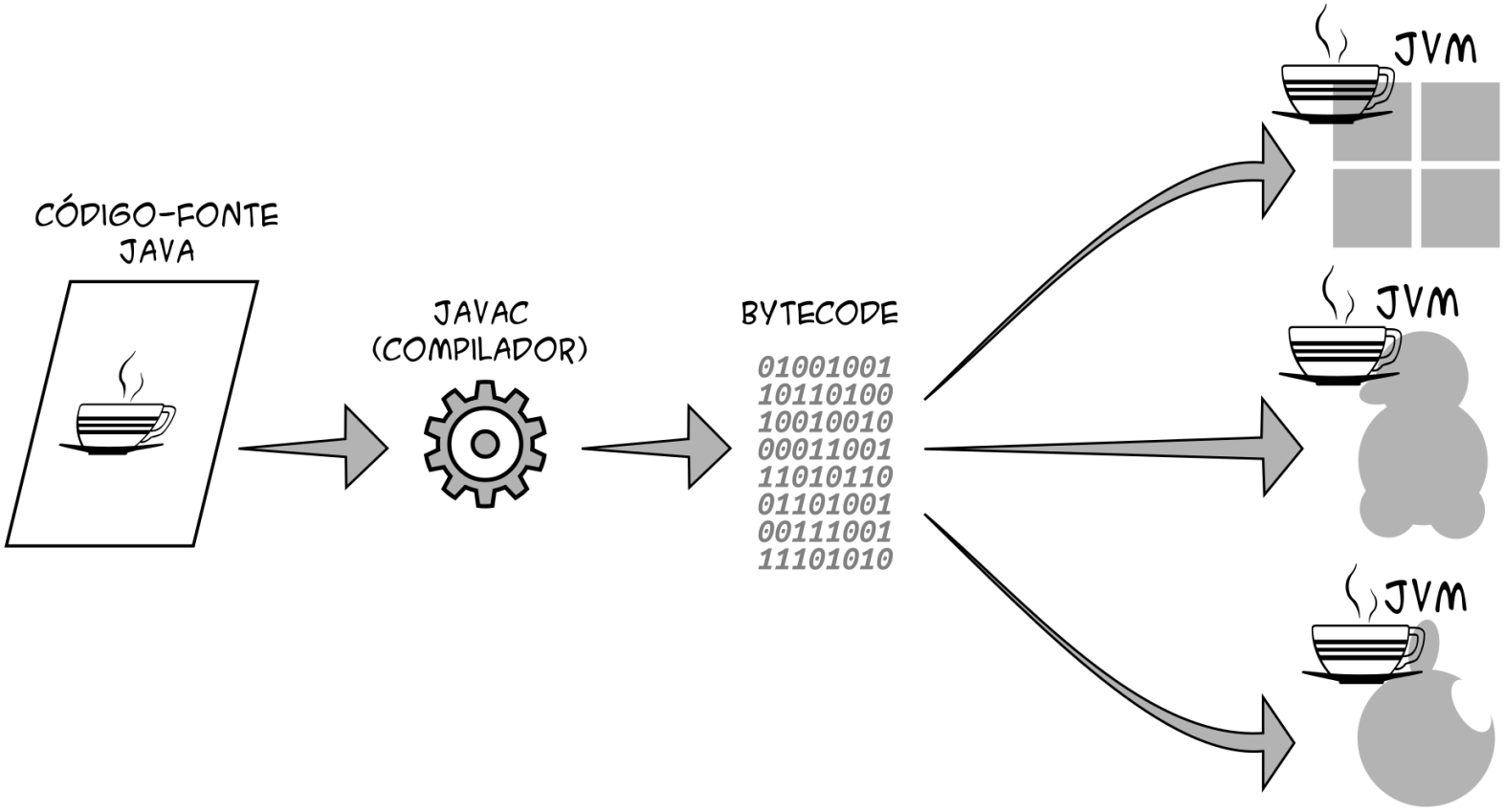
- Multiplataforma!



Fonte: O autor.

O que tem de tão especial no Java?

- Multiplataforma!



Fonte: O autor.

"Write Once, Run Everywhere"

Programação em JAVA

- **Java (Código fonte)**
  - Linguagem de programação de alto-nível.
- **JavaC (Java Compiler)**
  - Compilador da linguagem JAVA.
- **Bytecode**
  - Código intermediário para a Máquina Virtual.
- **JVM (Java Virtual Machine)**
  - Máquina virtual que interpreta o bytecode e gera código executável para o dispositivo.

UniCesumar

Sopa de letrinhas

- **JDK (Java Development Kit)**
  - Utilitários para compilação, depuração e execução.
- **JRE (Java Runtime Environment)**
  - Ambiente de tempo de execução onde o *bytecode* é executado.
- **Compilador JIT (Just-in-time)**
  - O compilador que traduz o *bytecode* em instruções do processador nativo.

Processo de compilação e interpretação

Processo de compilação e interpretação

- Uma aplicação Java deve ter pelo menos uma classe que contenha um método main().

```
public class OlaMundo {
    public static void main(String[] args) {
        System.out.println("Olá, Mundo!");
    }
}
```

Arquivo OlaMundo.java

Compilação

- O compilador (javac) gera o *bytecode:*

```
javac [opções] [arquivo fonte]
```

```
javac OlaMundo.java
```

Resulta em um *bytecode* OlaMundo.class

Compilação

- Como é um arquivo *bytecode* por dentro?

Arquivo OlaMundo.class

UniCesumar

Como rodar

- O bytecode pode ser executado pelo comando java:

```
java [-opções] nome_da_classe [argumentos]
```

```
java OlaMundo
```

Estruturas Básicas da Linguagem

Estruturas Básicas da Linguagem

- Tipos primitivos.
- Tipos inteiros.

Nome	Tamanho	Valores
byte	1 byte	-128 a 127
short	2 bytes	-32.768 a 32.767
int	4 bytes	-2.147.483.648 a 2.147.483.647
long	8 bytes	-9.223.372.036.854.775.808 até 9.223.372.036.854.775.807

Estruturas Básicas da Linguagem

- Tipos primitivos.
- Tipos de ponto flutuante.

Nome	Valores
float	Representam números reais de 32 bits com precisão simples. Podem assumir valores de ponto flutuante com formato definido pela especificação IEEE 754.
double	Representam números reais de 64 bits com precisão dupla. Assim como o float, podem assumir valores de ponto flutuante com formato especificado pela especificação IEEE 754.

Estruturas Básicas da Linguagem

- Tipos primitivos.
- Tipo char.

Nome	Valores
char	Representam notação de caracteres de 16 bits (2 bytes) para o formato Unicode UTF-16. Podem assumir valores entre ‘\u0000’ e ‘\uffff’ e valores numéricos entre 0 a 65535.

Estruturas Básicas da Linguagem

- Tipos primitivos.
- Tipo boolean.

Nome	Valores
boolean	Representam apenas 1 bit de informação (0 ou 1). Podem assumir apenas os valores *true* e *false*.

Variáveis

- Declaração

```
public class TiposPrimitivos {

    float pi;
    double tamanho;
    char estadoCivil;
    boolean aprovado;
    short s;
    byte b;
    int _variavel;
    long $variavel2;
```

Variáveis

- Inicialização.
  - Uma variável deve receber um valor inicial.

```
int numero

//Código ok
numero = 5;
numero = numero + 5;
```

```
int numero;

//Código incorreto
numero = numero + 5;
```

Escopo de variável

- Inicialização.
  - Uma variável deve receber um valor inicial.

```
int numero

//Código ok
numero = 5;
numero = numero + 5;
```

```
int numero;

//Código incorreto
numero = numero + 5;
```

Escopo de variável

Escopo de variável

- Variáveis de instância (atributos):

```
public class Livro {

    long preco;
    int quantidade;
    char tipo;

}
```

* Declaradas dentro da classe, mas fora dos métodos.

Escopo de variável

- Variáveis locais:

```
public class Livro {

    public void metodoContar() {
        int contador = 20;
        contador = contador + 1;
    }

}
```

* Declaradas dentro dos métodos.

Escopo de variável

- Variáveis de classe (estáticas):

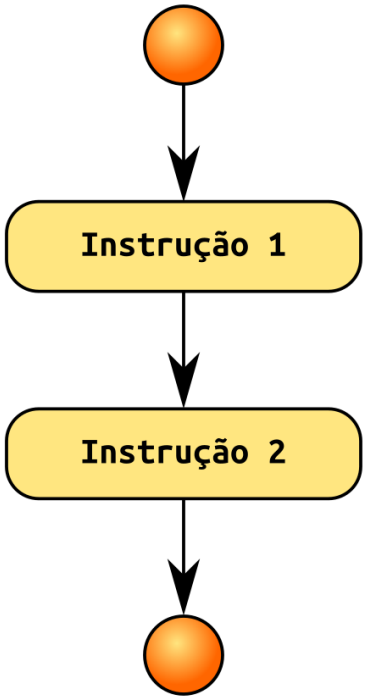
```
public class Livro {

    // Variável estática representando 20% de desconto
    public static int DESCONTO = 20;

    public void vender(double valor) {
        double novoValor = valor - (valor * DESCONTO)/100;
    }

}
```

* A mesma variável disponível a todos os objetos da classe.

Estruturas de controle

Estruturas de controle

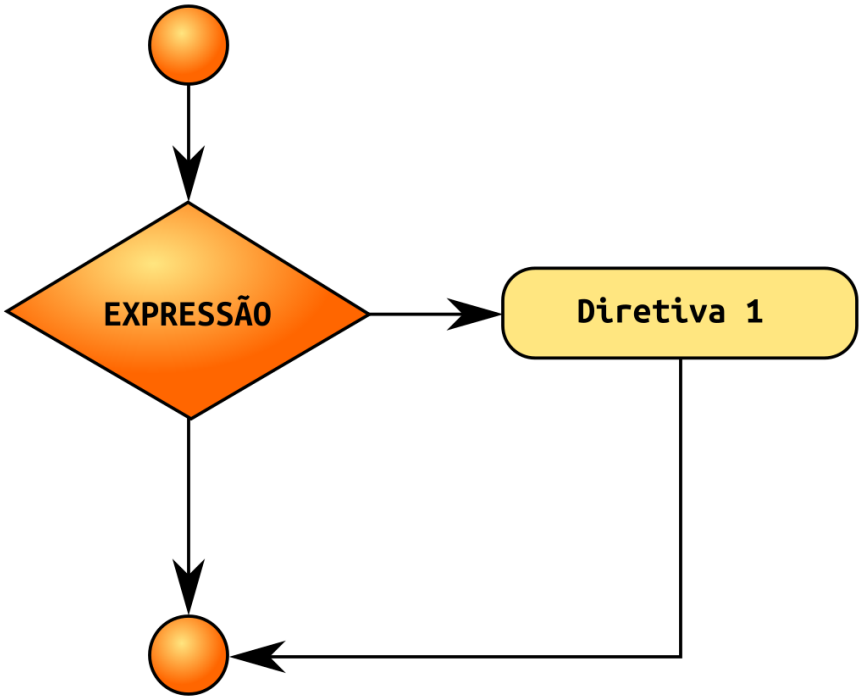
- Instruções de sequência:




```
Leia um valor X;

Calcule Y como função de X: Y = X * 2;

Imprima o resultado;
```

Estruturas de controle

- Instruções de seleção:




```
if (expressão) {
    diretiva
}
```

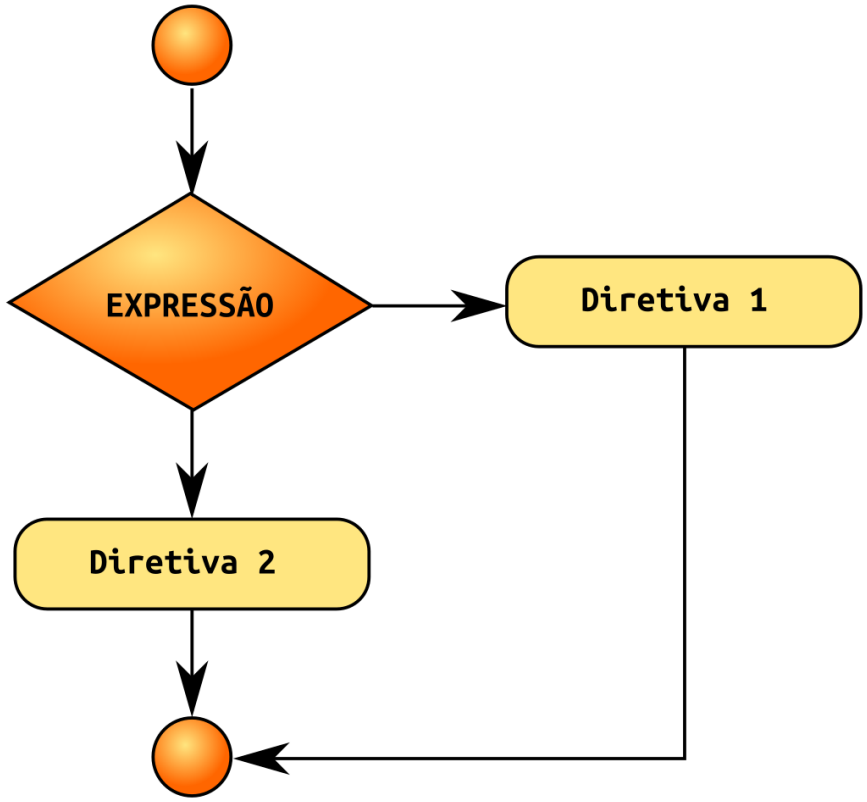
Estruturas de controle

- Instruções de seleção:

```
public class EstruturaDeControle {
    //exemplo de comando if

    public static void main(String args[]) {
        int valor1 = 5;
        int valor2 = 10;

        if (valor1 > valor2) {
            System.out.println("Valor 1 é maior que valor 2");
        }

        System.out.println("Soma dos valores: " + (valor1 + valor2));
    }

}
```

Estruturas de controle

- Instruções de seleção:




```
if (expressão) {
    Diretiva 1
} else {
    Diretiva 2
}
```

Estruturas de controle

- Instruções de seleção:

```
public class EstruturaDeControle {
    //exemplo de comando if e else

    public static void main(String args[]) {
        int valor1 = 5;
        int valor2 = 10;

        if (valor1 > valor2) {
            System.out.println("Valor 1 é maior que valor 2");
        } else {
            System.out.println("Valor 2 é maior que valor 1");
        }

        System.out.println("Soma dos valores: " + (valor1 + valor2));
    }

}
```

UniCesumar

Exercício

- Crie uma classe “Clima” que tenha uma variável interna “temperatura” (que você pode inicializar com qualquer valor. O método main deve verificar a temperatura. Se for maior que 25, imprima “Está calor!”. Senão imprima “Está frio!”.

Exercício

- Crie uma classe “Clima” que tenha uma variável interna “temperatura” (que você pode inicializar com qualquer valor. O método main deve verificar a temperatura. Se for maior que 25, imprima “Está calor!”. Senão imprima “Está frio!”.

```
public class Clima {
    public static void main(String args[]) {
```

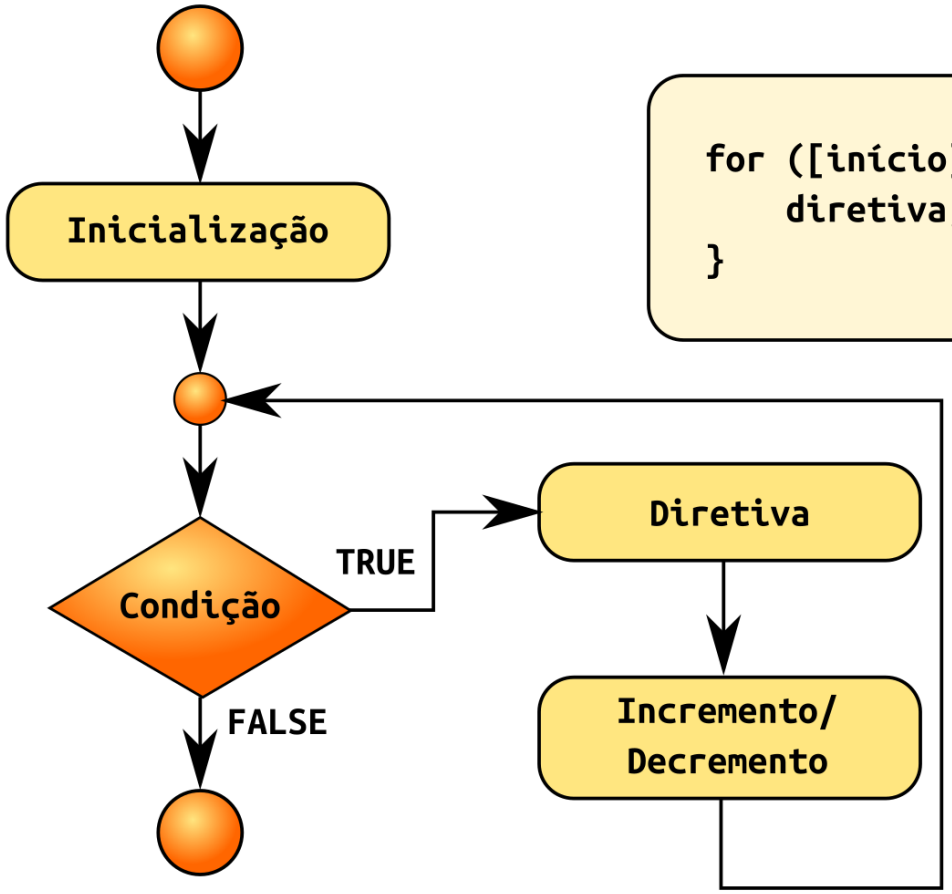
UniCesumar

Exercício

```
public class Clima {
    public static void main(String args[]) {
        int temperatura = 27;

        if (temperatura > 25) {
            System.out.println("Está calor!");
        } else {
            System.out.println("Está frio!");
        }
    }
}
```

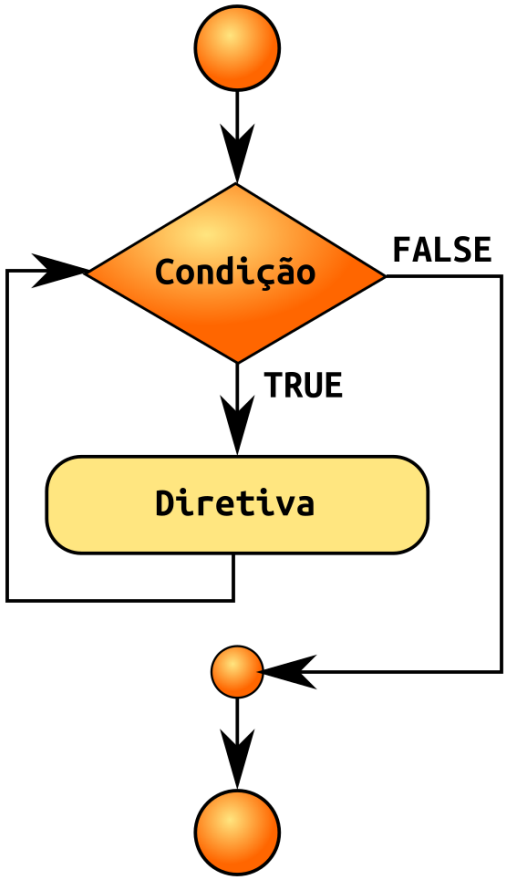
Estruturas de repetição

Estruturas de repetição



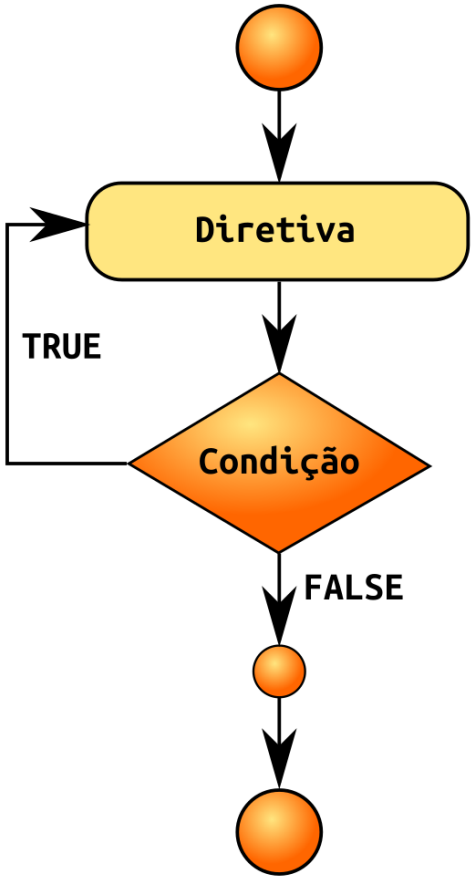

```
for ([início]; [condição]; [inc/dec]) {
    diretiva;
}
```

Estruturas de repetição




```
while (condição) {
    diretiva;
}
```

Estruturas de repetição




```
do {
    diretiva;
} while (condição);
```

Desvios incondicionais

- Break: interrompe o looping e continua a sequência que vem após o looping.
- Continue: interrompe apenas a iteração atual e continua o looping na próxima iteração.

Desvios incondicionais

```
public class Desvios {
    public static void main(String args[]) {
        for (int i = 0; i < 5; i++) {
            if (i == 2) continue;

            System.out.println(i);
        }
    }
}
```

Saída:

```
0
1
3
4
```

Desvios incondicionais

```
public class Desvios {
    public static void main(String args[]) {
        for (int i = 0; i < 5; i++) {
            if (i == 2) break;

            System.out.println(i);
        }
    }
}

Saída:
0
1
```

UniCesumar

Exercício

- 1. Crie uma classe em Java cuja função *main* realize a soma de todos os números pares de 0 a 500.

UniCesumar

Exercício

```
public class Soma {
    public static void main(String args[]) {
        int resultado = 0;

        for (int i = 0; i <= 500; i++) {
            if (i % 2 == 0) {
                resultado = resultado + i;
            }
        }

        System.out.println(resultado);
    }
}
```

UniCesumar

Exercício

```
public class Soma {
    public static void main(String args[]) {
        int resultado = 0;

        for (int i = 0; i <= 500; i+=2) {
            //if (i % 2 == 0) {
                resultado = resultado + i;
            //}
        }

        System.out.println(resultado);
    }
}
```

UniCesumar

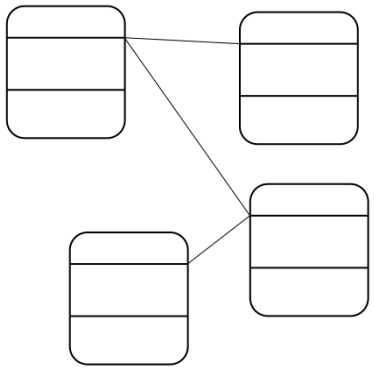
Exercício

```
public class Soma {
    public static void main(String args[]) {
        int resultado = 0;

        for (int i = 0; i <= 500; i+=2) {

                resultado += i;

        }

        System.out.println(resultado);
    }
}
```

Cenas dos próximos capítulos

UNIDADE III

CLASSES x OBJETOS

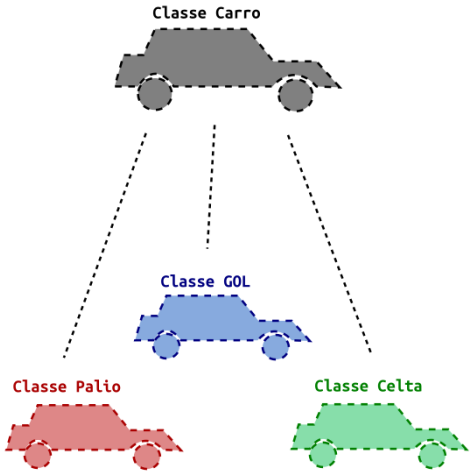


UNIDADE IV

MODIFICADORES E ENCAPSULAMENTO

UNIDADE III

HERANÇA E POLIMORFISMO



Programação I

Prof. Andre Noel