



GLADIOS

MORDEKAISER



©PHOTO360

Construção de Algoritmos

Iniciamos em alguns minutos...

Aula 01 – 03

Linguagens de Programação

Prof. Luciano Freire

Linguagens de Programação



- ❖ Servem para escrever programas que permitem a comunicação entre o usuário e o computador
- ❖ Para isto existe a necessidade de programas especiais chamados de tradutores


Um programa tradutor converte um programa escrito em uma linguagem de programação para a linguagem de máquina, a qual o computador consegue entender

Tipos de Tradutores



 Compiladores

 Interpretadores

 Convertem um programa escrito em uma Linguagem de Programação em
sequência de zeros e uns (linguagem de máquina)

Instruções ao Computador



- ✚ Diferentes passos de um algoritmo são expressos em programas como instruções
- ✚ Um programa é então uma sequência de instruções
 - ✚ cada instrução especifica certas operações que um programa deve executar
 - ✚ Importante: é o programa que diz ao computador (CPU) o que deve ser realizado

- ✚ A elaboração de um programa exige a utilização de um repertório de instruções de uma determinada linguagem
- ✚ Apesar das linguagens serem diferentes, alguns conjuntos de instruções são comuns
- ✚ Todas as linguagens possuem o seguinte conjunto de instruções padrão
 - ✚ Instruções de Entrada e Saída
 - ✚ Instruções Aritmética e Lógicas
 - ✚ Instruções de seleção
 - ✚ Instruções de repetição

Instruções de Entrada e Saída



🔗 Usada para transferência de informações entre os dispositivos periféricos

como

🔗 teclado

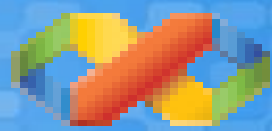
🔗 unidade de disco

🔗 monitor

🔗 impressoras

🔗 para a memória principal onde são armazenados temporariamente e manipulados pelos programas

Instruções Aritmética e Lógicas



🌀 Executam operações aritméticas como

🌀 soma

🌀 subtração

🌀 divisão

🌀 Multiplicação

🌀 Operações Lógicas

🌀 e (and)

🌀 ou (or)

🌀 não (not)

🌀 Operações Relacionais

🌀 $>$

🌀 $<$

🌀 $>=$

🌀 $<=$

🌀 $==$ (igual)

🌀 $!=$ (diferente)

Instruções de Seleção



✚ Permitem a seleção de tarefas alternativas em função do resultado de operações condicionais

✚ Exemplos de Java



if



if ... else



case

Instruções de Repetição



🌀 permitem a repetição de um determinado conjunto de instruções um determinado número de vezes

🌀 Exemplos Java

🌀 for

🌀 while

🌀 do ... while

Linguagens de Alto Nível



- ✚ Cada Linguagem de programação define um determinado conjunto de instruções
- ✚ Estas instruções formam uma linguagem utilizada para definir ações que o computador deve executar
- ✚ As linguagens mais próximas da língua falada são chamadas de **linguagens de alto nível**
- ✚ enquanto que as linguagens mais próximas da linguagem de máquina são chamadas de **linguagens de baixo nível**

Exemplos



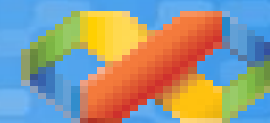
Linguagens de alto nível	Linguagem de Baixo Nível
Pascal	Assembly
C	
C++	
Ruby	
Java	
C#	
Fortran	



Independente se a linguagem é de alto ou baixo nível ela precisa ser convertida para **a linguagem de máquina** para ser executada.
Isto é tarefa dos compiladores e Interpretadores

Exemplos

.net



Assembly

```
;exemplo2
.model small
.stack
.code
mov ah,2h ;move o valor 2h para o registrador ah
mov dl,2ah ;move o valor 2ah para o registrador dl
                ;(é o valor ASCII do caractere *)
int 21h      ;interrupção 21h
mov ah,4ch ;função 4ch, sai para o sistema operacional
int 21h      ;interrupção 21h
end          ;finaliza o programa
```

Java

```
public class Estudos{
    public static void main(String args[]){
        int valor = 3;

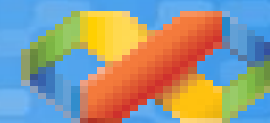
        if(valor > 3)
            System.out.println("Valor maior que 3");
        else if(valor < 3)
            System.out.println("Valor menor que 3");
        else
            System.out.println("Valor é igual a 3");
    }
}
```

C

```
#include <stdio.h>
int main ()
{
    int num;
    printf ("Digite um numero: ");
    scanf ("%d",&num);
    if (num>10)
        printf ("\n\nO numero e maior que 10");
    if (num==10)
    {
        printf ("\n\nVoce acertou!\n");
        printf ("O numero e igual a 10.");
    }
    if (num<10)
        printf ("\n\nO numero e menor que 10");
    return(0);
}
```

Exemplos

.net



Python

```
numero = 10
valor = int(raw_input("Informe um inteiro: "))

if valor == numero:
    print "Parabéns, você acertou"
    print "Já tentou a Megasena?"
elif valor < numero:
    print "Tente um número maior"
else:
    print "Tente um número menor"
```

Ruby

```
x=1
if x > 2
  puts "x is greater than 2"
elsif x <= 2 and x!=0
  puts "x is 1"
else
  puts "I can't guess the number"
end
```

Cobol

```
IF PESO-BRUTO = PESO-INFORMADO
  IF PESO-BRUTO LESS THAN 70
    PERFORM 900-PESO-ABAIXO THRU 900-FIM
  ELSE
    PERFORM 910-PESO-NORMAL THRU 910-FIM
  END-IF
ELSE
  PERFORM 920-PESO-ACIMA THRU 920-FIM
END-IF.
```

Clipper

```
LOCAL MEDIA:= 0
CLEAR
@ 10,10 SAY "DIGITE A MEDIA DO ALUNO..."GET MEDIA
READ
  IF MEDIA <5
    ? "REPROVADO"
  ELSEIF MEDIA = 5
    ? "RECUPERAÇÃO"
  ELSE
    ? "APROVAADO"
  ENDIF
```


Tradução para linguagem máquina



BASEFIP

Tradução para linguagem de Máquina



Existem duas abordagens para fazer esta tradução

Compilação

Interpretação

OBS:

A linguagem Java utiliza uma combinação desta duas técnicas

✚ É executado por um programa especial chamado **compilador**

✚ Existem compiladores para as linguagens de programação:

✚ Java

✚ C

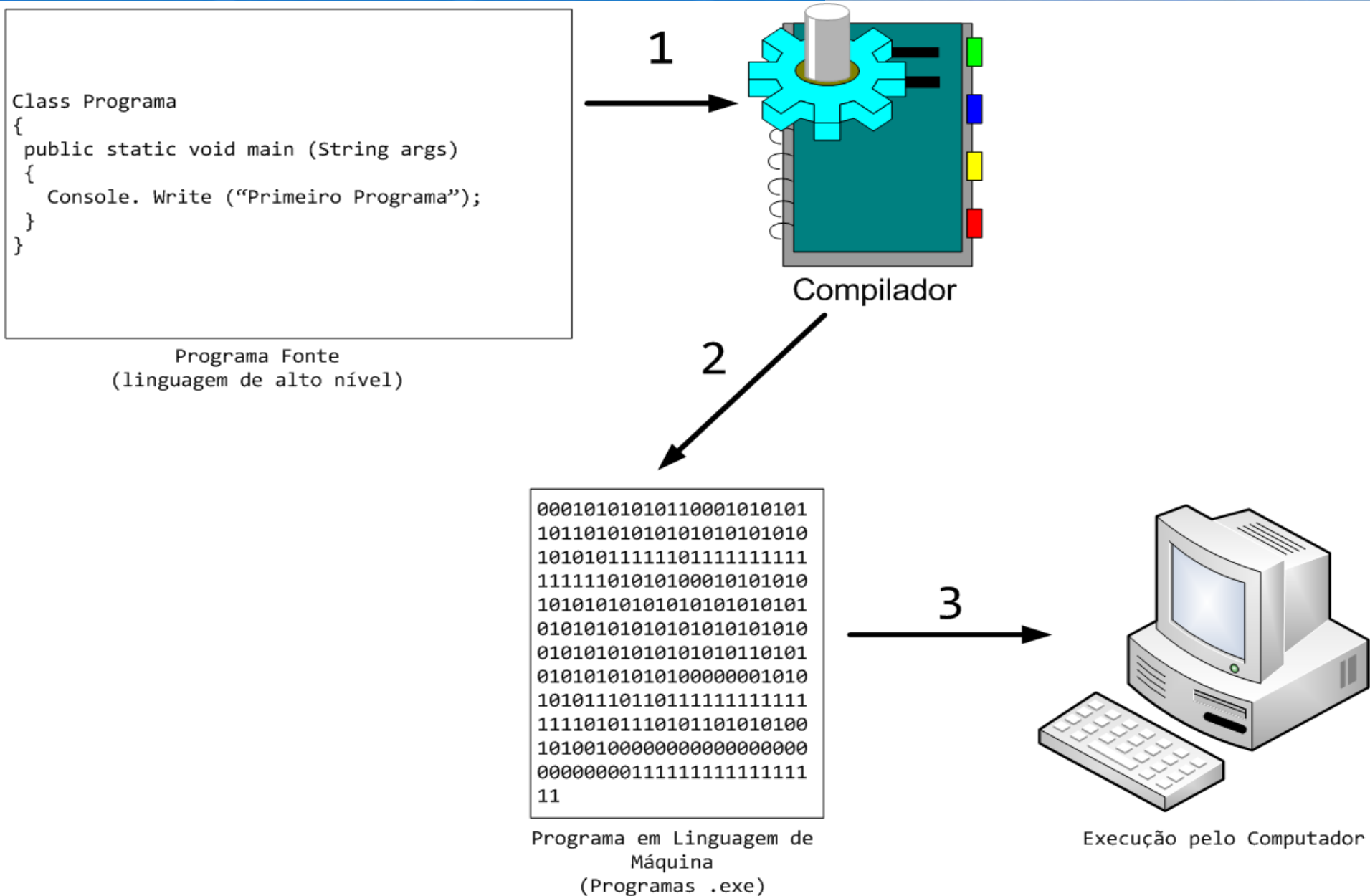
✚ Pascal

✚ C#

✚ C++

- ✚ A partir de um programa, escrito
 - 🌈 utilizando as instruções da linguagem
 - 🌈 armazenadas em um arquivo texto
- ✚ O compilador produz um arquivo no formato binário que pode ser diretamente executado pelo computador

Compilação



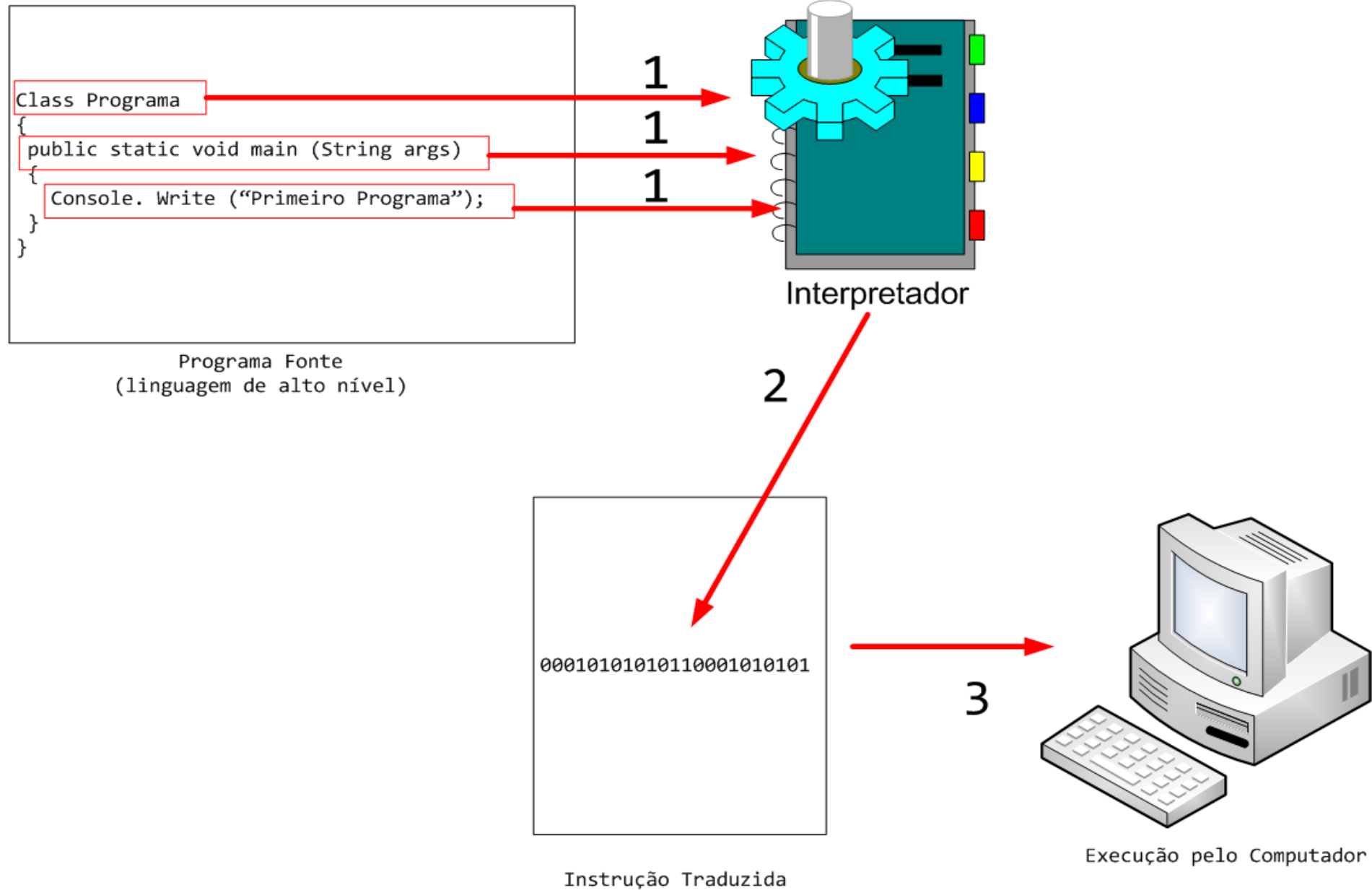
Características da Compilação



- ✚ Maior demora na conversão
- ✚ Execução mais rápida do programa
 - 🌈 programa binário dependente
 - 🌈 sistema operacional
 - 🌈 arquitetura do computador
- ✚ Para cada sistema operacional diferente e arquitetura diferente existe a necessidade de um compilador diferente
- ✚ Caso o programa necessite ser executado em outro SO ou computador existe a necessidade de ser recompilado

- 🧩 O interpretador lê cada instrução do programa fonte traduz uma instrução de cada vez
- 🧩 As instruções são executadas pelo computador a medida que são traduzidas

Interpretação



Características da Interpretação



- ❖ Performance do Programa é menor
- ❖ Independência de plataforma
 - 🔄 desde que exista um interpretador para os diferentes sistemas operacionais e arquitetura
- ❖ Facilidade de encontrar erros em tempo de execução antes que comprometam o funcionamento do programa

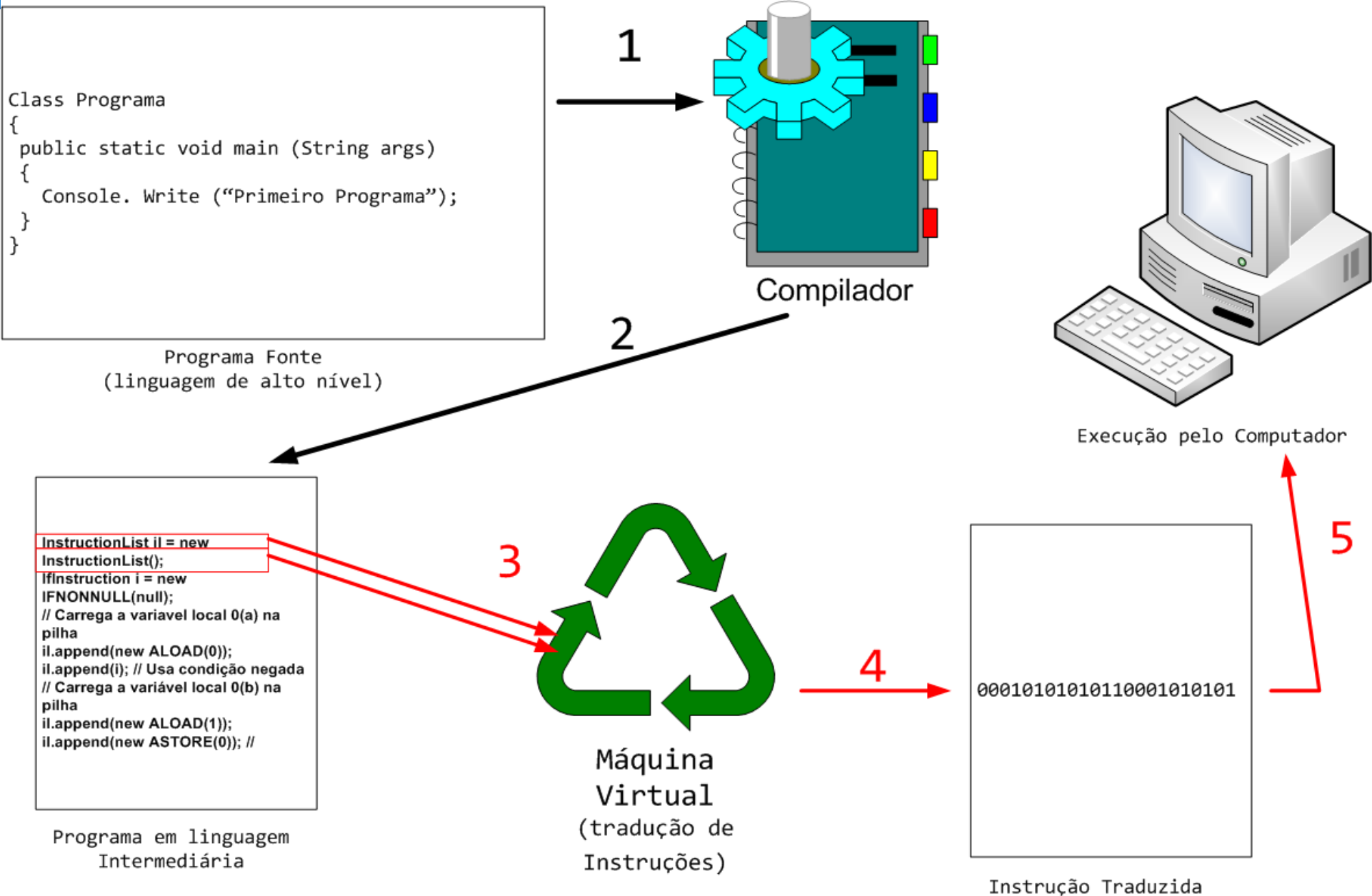
Abordagem Mista



- 🔗 Linguagens de programação que utilizam máquinas virtuais utilizam uma abordagem que envolve compilação e interpretação
- 🔗 Isto garante a
 - 🌐 portabilidade entre múltiplas plataformas (Java)
 - 🌐 integração entre diferentes linguagens (.Net)

- 🌀 O processo funciona da seguinte maneira
 - 🌀 Código fonte é convertido em uma linguagem intermediária (compilação)
 - 🌀 Este código intermediário é traduzido pela máquina virtual em código de máquina instrução por instrução
 - 🌀 e enviado a CPU para execução

Abordagem Mista



X

$$X^2 + pX$$

y

Dúvidas?

Muito Obrigado!

Prof. Luciano Freire
luciano.freire@facens.br