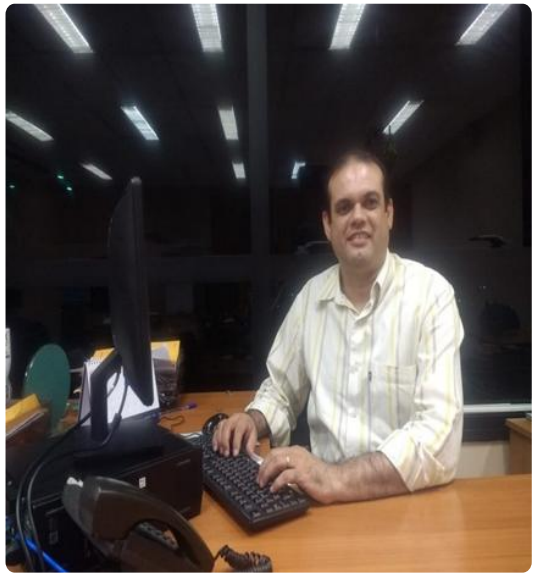


Compiladores

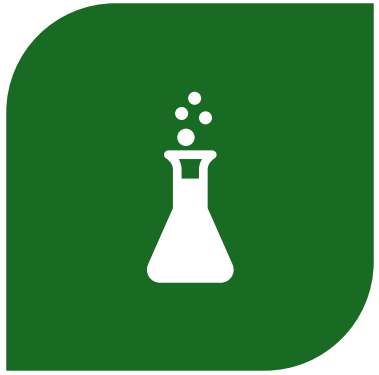
Prof. Me. Wanderlan Carvalho de Albuquerque

Experiência em Nível superior : 19 anos

UNINORTE

Ciência da Computação

TURMA:
SPL0790107NNA



SALA : LABORATÓRIO 7

HORÁRIO : 18:30 ÀS 21:10

Plano de Ensino

- Objetivo: Apresentar Plano de Ensino e conceitos de Compiladores

Plano de Ensino

- **Ementa :**
  Organização e estrutura de compiladores e interpretadores. Análise léxica. sintática. Alocação e gerência de memória. Representação interna de código-fonte. Análise semântica. Geração de código. Otimização de código. Máquinas abstratas e ambientes de tempo de execução. Especificação de linguagem de programação no nível sintático e semântico.
- **Carga horária** : 60h

Referência Bibliográfica

- AHO, A. Compiladores: princípios, técnicas e ferramentas. Editora Pearson. 1995.
- LOUDEN, Kenneth. Compiladores: princípios e práticas. Editora Pioneira Thomson. 2004.
- GERSTING, Judith L. Fundamentos Matemáticos para a Ciência da Computação: Um tratamento moderno de Matemática Discreta. Trad. Valéria de Magalhães Lorio. 5. ed. Rj: LTC, 2004.

- Pode ser atualizada pelo professor

Competências Específicas

- Estudar das linguagens, suas representações, e classificações no âmbito da Teoria da Computação;
- Permitir a classificação dos diferentes tipos de linguagens, e conhecer os mecanismos geradores e reconhecedores para cada tipo.
- Fornecer subsídios para implementar o comp ilador de uma linguagem de programação, desde a definição da linguagem até a construção dos analisadores léxico e sintático

Metodologia de Ensino e Aprendizagem

- A disciplina, dependendo de sua natureza, pod e ser ministrada através de conteúdos teóricos, conteúdos práticos, e ainda pode utilizar recurs os de exposições dialogadas, grupos de discussão, se minários, debates competitivos, apresentação e discussão de filmes e casos práticos, onde os conteúdos podem ser trabalhados mais dinamicamente, estimulando o senso crítico e científico dos alunos.

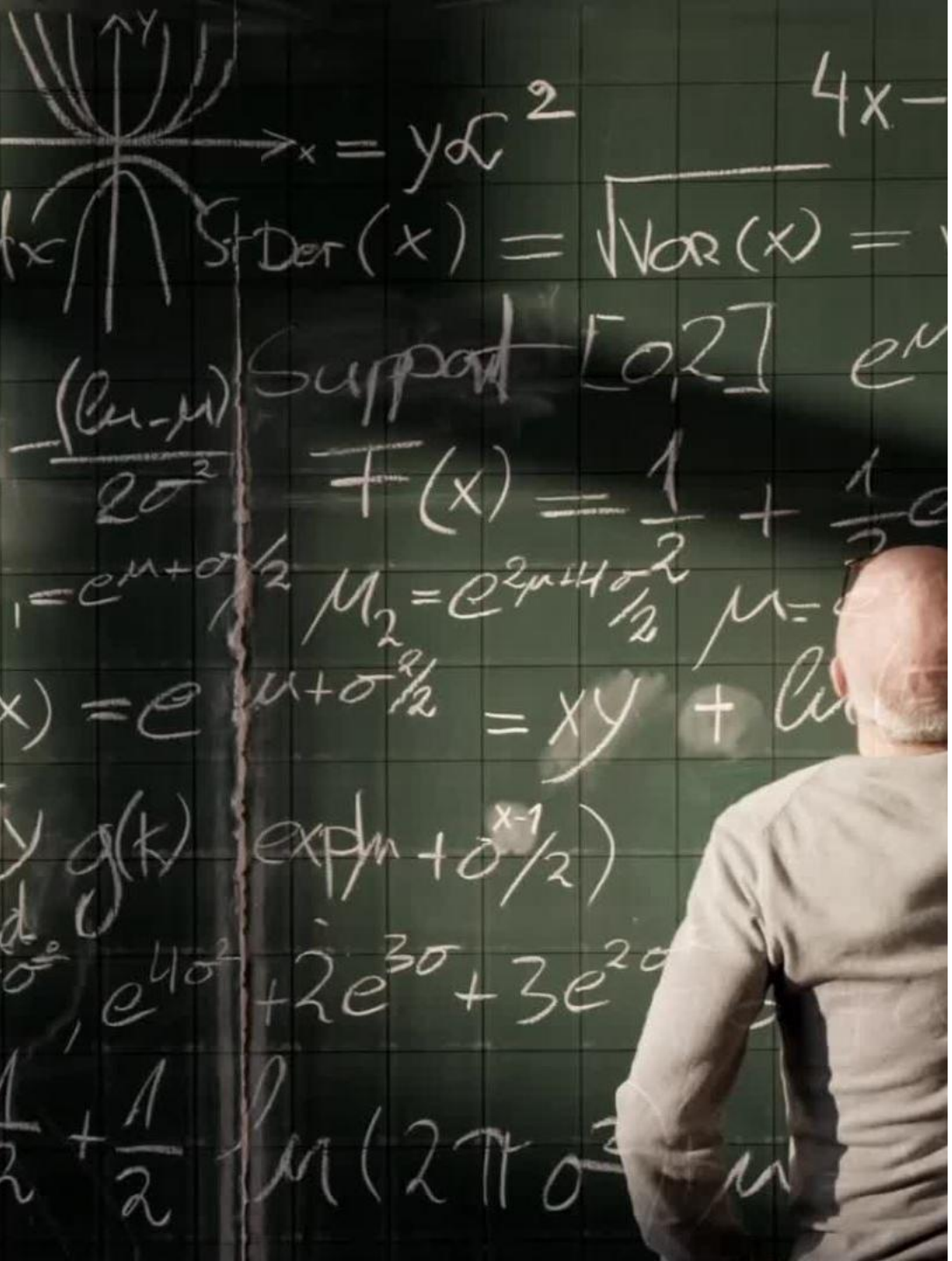
Datas das avaliações

AVALIAÇÃO 1 : 03/04 a 09/04

AVALIAÇÃO 2 : 03/06 a 09/06

AVALIAÇÃO DA SEGUNDA CHAMADA:12/06 a 18/06

AVALIAÇÃO FINAL: 23/06 a 26/6



Faltas e Presenças

- Acompanhar no portal e procurar primeiramente o professor
- Observar se estar matriculado

Disponibilização de Conteúdo

CONTEÚDO
DISPONIBILIZADO VIA TEAMS

LIVROS E ATUALIZAÇÕES
PASSADO PELO PROFESSOR

Agenda

Introdução a Compiladores

Objetivo

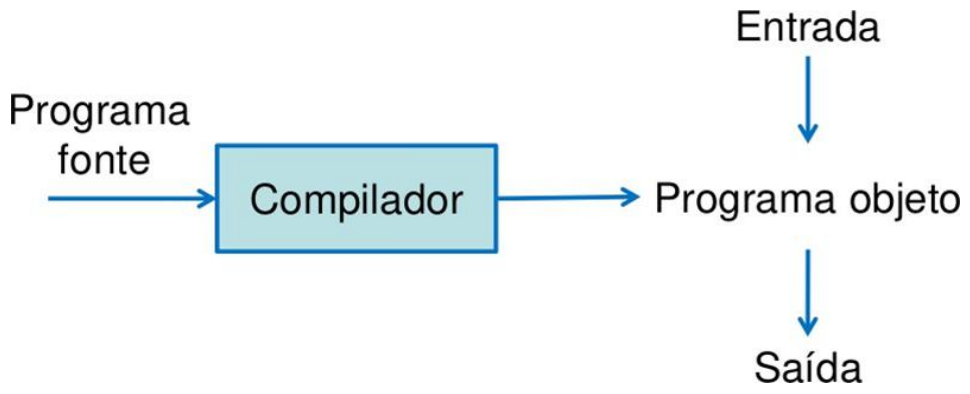
- *Conhecer os conceitos básicos utilizados no emprego de Compiladores*

Linguagem de Programação

- Linguagens de programação são notações que se usa para descrever as computações e comunicar com as pessoas e máquinas;
- Então, todos os softwares executados em todos os computadores foram escritos em alguma linguagem de programação;
- Mas, antes que possa rodar, um programa primeiro precisa ser traduzido p/ um formato que lhe permita ser executado em um computador.
- Os compiladores fazem essa tradução;

Compilador

- “**Compilador** é um programa de computador que lê um programa escrito em uma linguagem (linguagem fonte) e a traduz em um programa equivalente em outra linguagem (linguagem objeto)”;
- A **linguagem fonte** é a linguagem de programação que você usa para escrever o código original do seu programa. Por exemplo, se você escrever um programa em C++, C++ é a linguagem fonte desse programa.
- **Função** : relatar quaisquer erros no programa fonte detectados durante este processo de tradução.



O que são Compiladores?

1 Tradução de Código

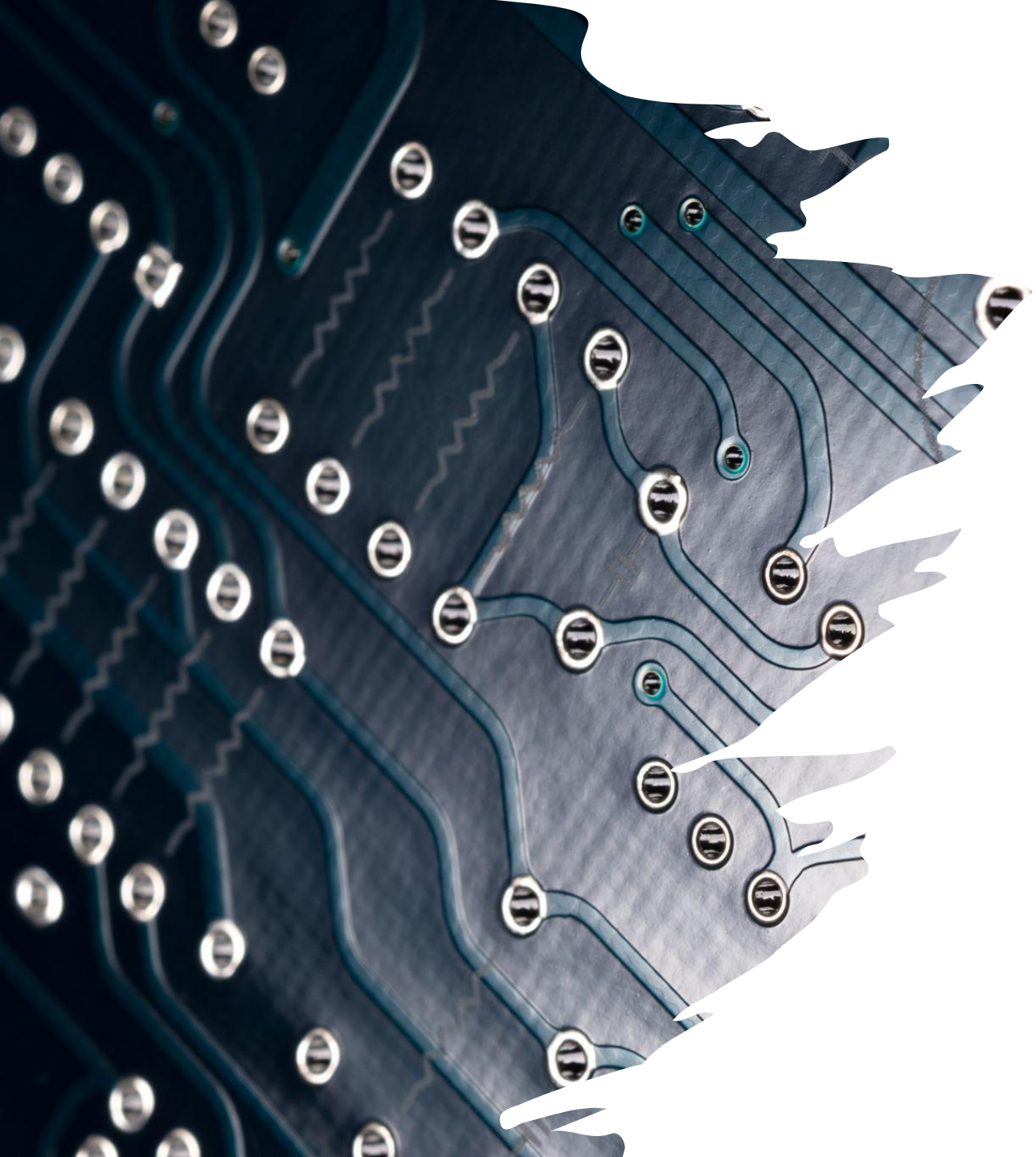
Os compiladores são programas que traduzem o código escrito em linguagens de programação de alto nível para linguagem de máquina, que pode ser executada pelo computador.

2 Essencial para Programação

Os compiladores são essenciais para a programação, pois permitem que os desenvolvedores escrevam código em linguagens entendíveis e mais próximas da linguagem humana.

3 Detecção de Erros

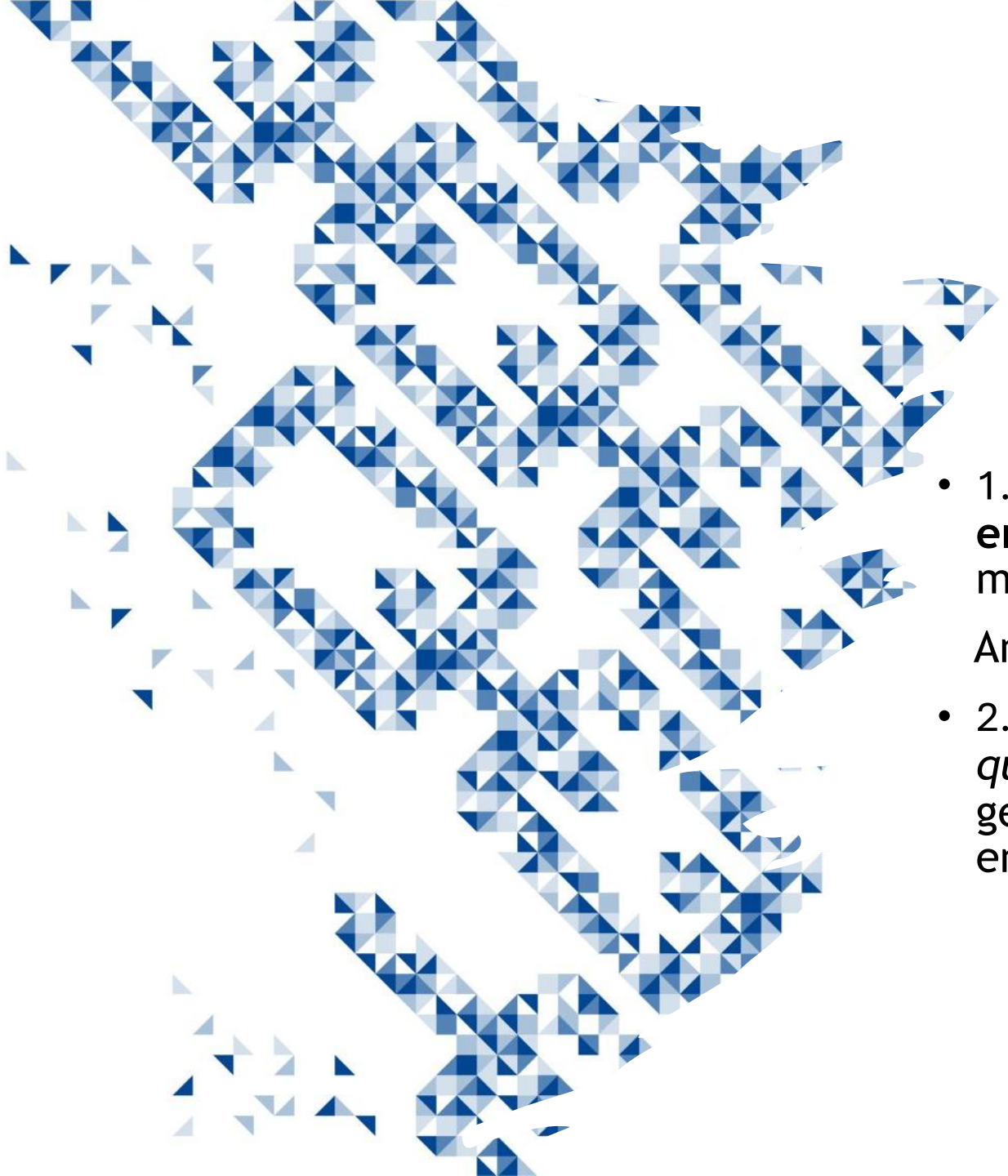
Além de traduzir o código, os compiladores fazem uma verificação da exatidão e integridade do código, identificando possíveis erros de sintaxe e semântica.



Compilador X Interpretador

- O programa objeto em uma linguagem de máquina produzido por um **compilador** normalmente é **muito mais rápido** no mapeamento de entrada para saídas do que um interpretador.

- Porém, um **interpretador** frequentemente oferece um melhor **diagnóstico de erro** do que um compilador, pois executa no programa fonte instrução por instrução.



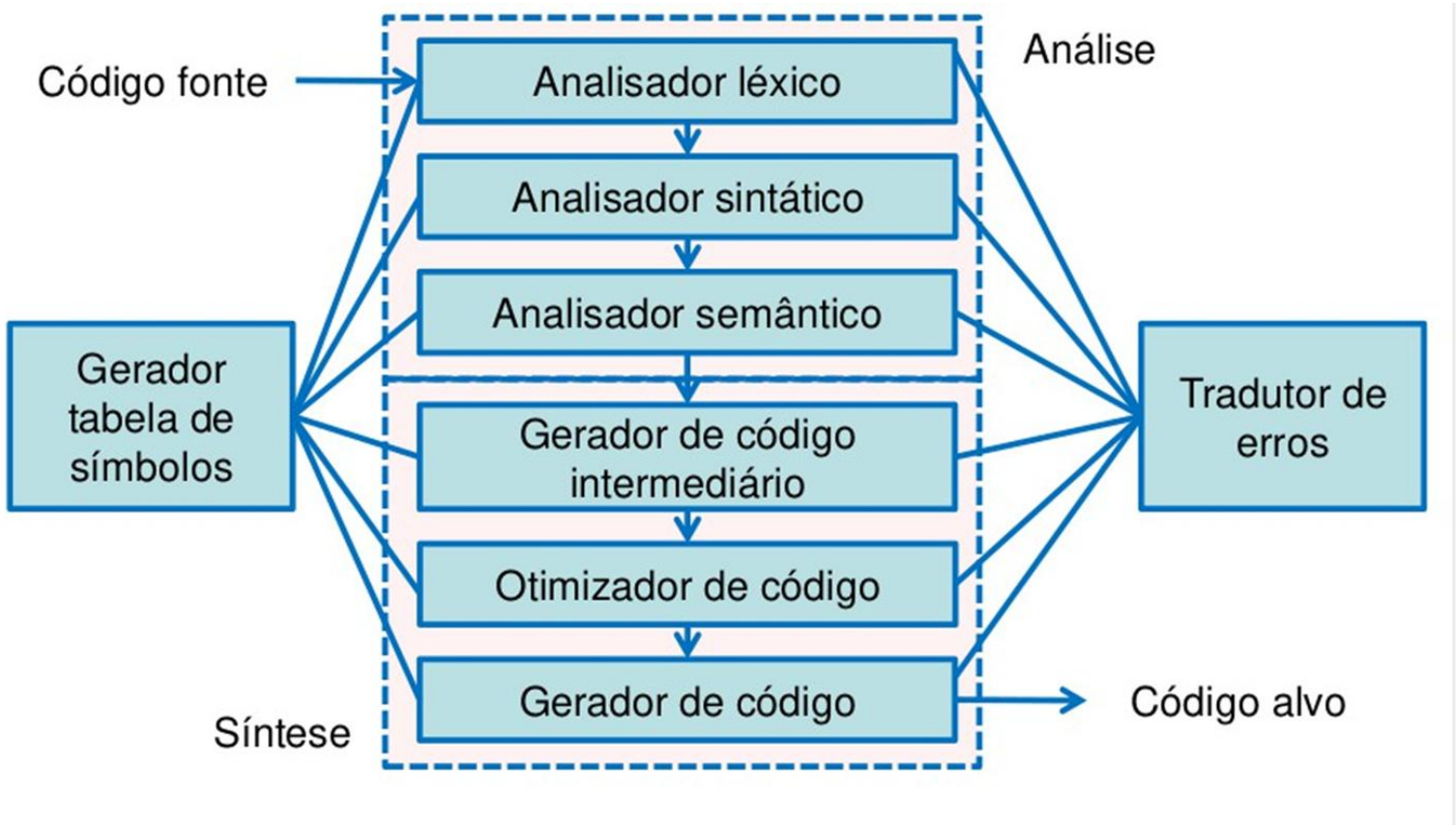
Tarefas básicas executas por um compilador

- 1.**Análise (Front-end)**, *em que o texto de entrada (na* linguage m fonte) é examinado, verificado e compreendido :

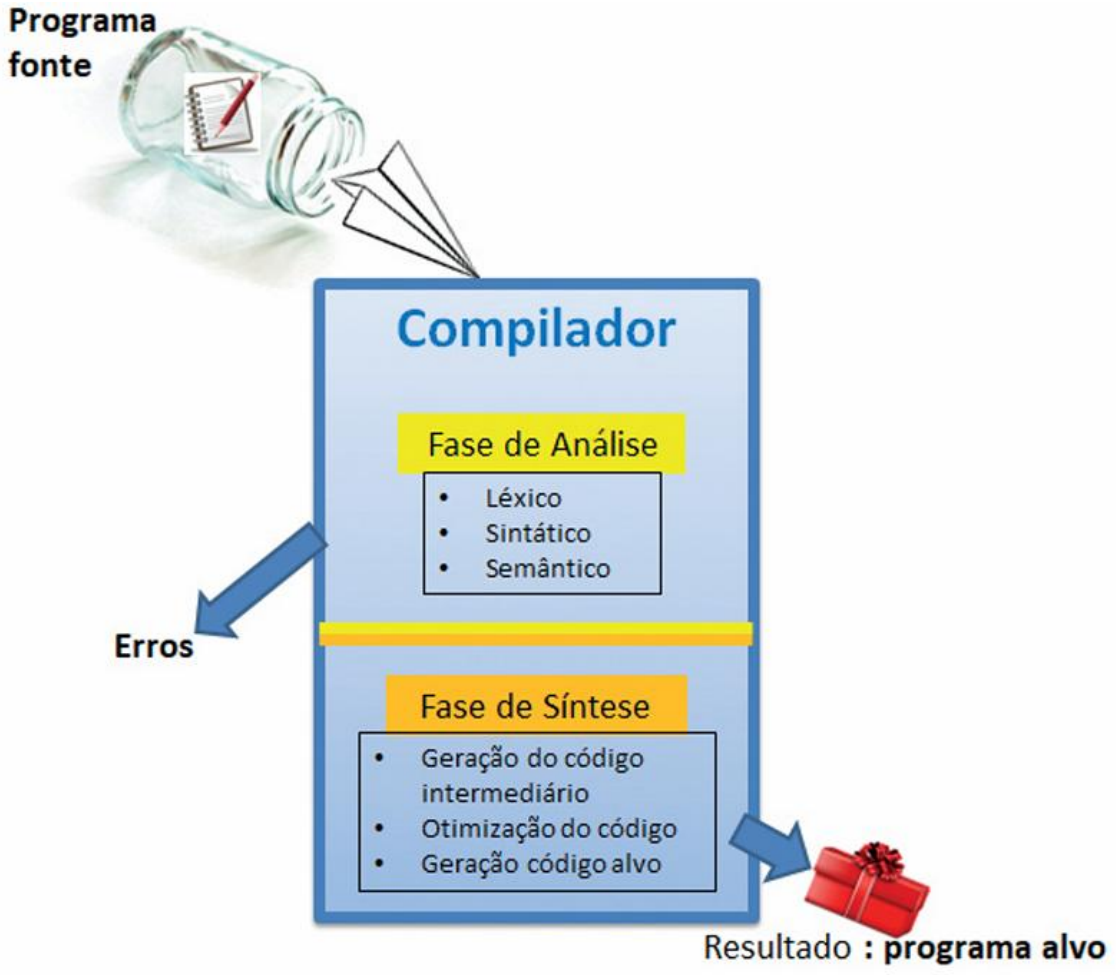
  Análise léxica, sintática e semântica;

- 2.***Síntese (Back-end)***, *ou geração de código, em que o* texto de saída (na linguagem objeto) é gerado, de forma a corresponder ao texto de entrada

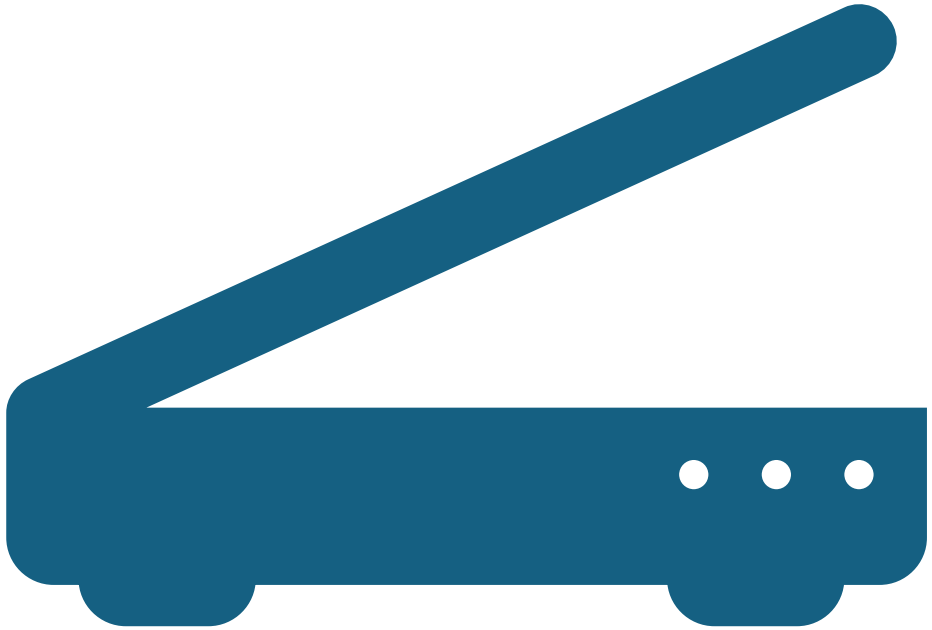
Fase do Compilador



Fase do Compilador



Análise léxica



- Também chamada de **scanner**
- Agrupa caracteres em **símbolos (ou tokens)**
- Entrada: fluxo de caracteres
- Saída: fluxo de símbolos Símbolos são:

Palavras reservadas, identificadores de variáveis e procedimentos, operadores, pontuação,...

- Uso de expressões regulares no reconhecimento
- **Lex/Flex s**ão ferramentas que geram scanners

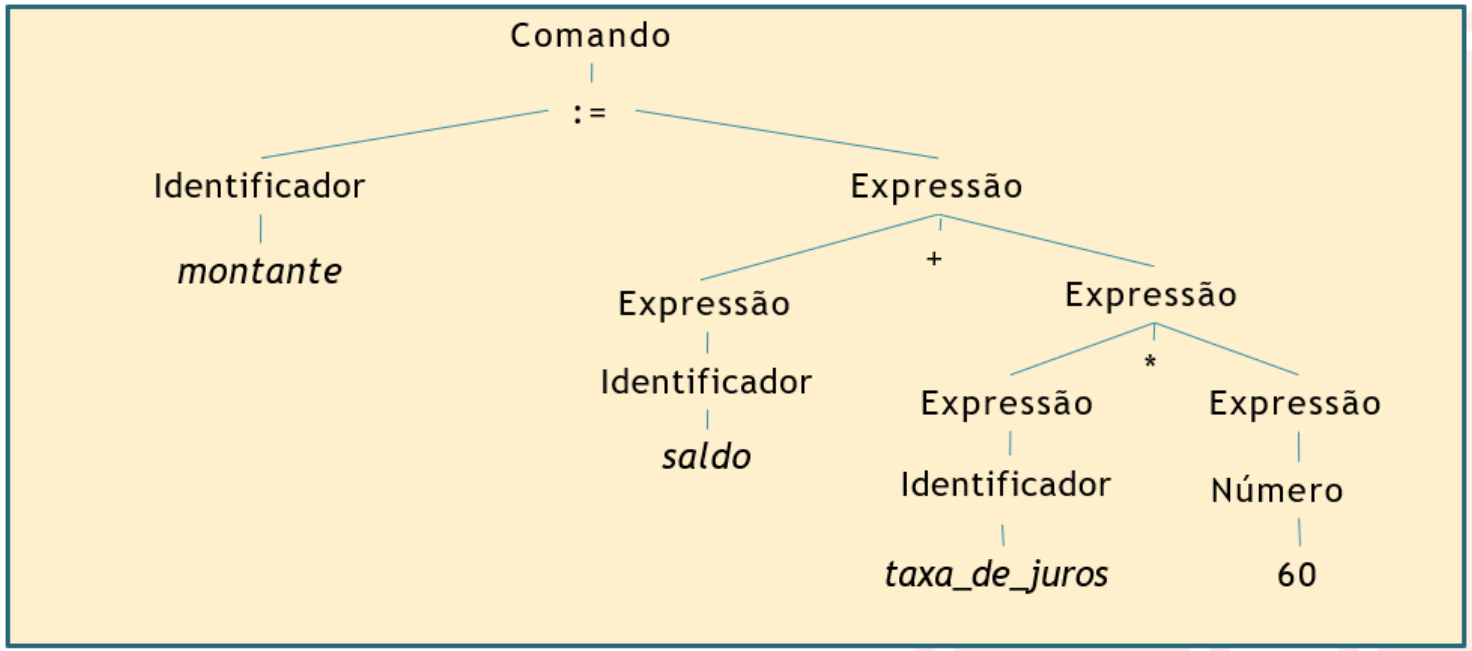
Análise Léxica

- Dado os caracteres da instrução
- **montante := saldo + taxa_de_juros * 30;**
- São identificados os seguintes *tokens*:
  - Identificador *montante*
  - Símbolo de atribuição *:=*
  - Identificador *saldo*
  - Símbolo de adição +
  - Identificador *taxa_de_juros*
  - Símbolo de multiplicação *
  - Número *30*

Análise Sintática

- Também **chamada de *parser***
- Agrupa símbolos em unidades sintáticas
- Ex.: os 3 símbolos **A+B** podem ser agrupados em uma estrutura chamada de *expressão.*
- Expressões depois podem ser agrupados para formar comandos ou outras unidades**.** Saída: representação árvore de parse do programa
- **Gramática livre de contexto** é usada para definir a estrutura do programa reconhecida por um parser
- **Yacc/Bison** são ferramentas para gerar parsers

Análise Sintática



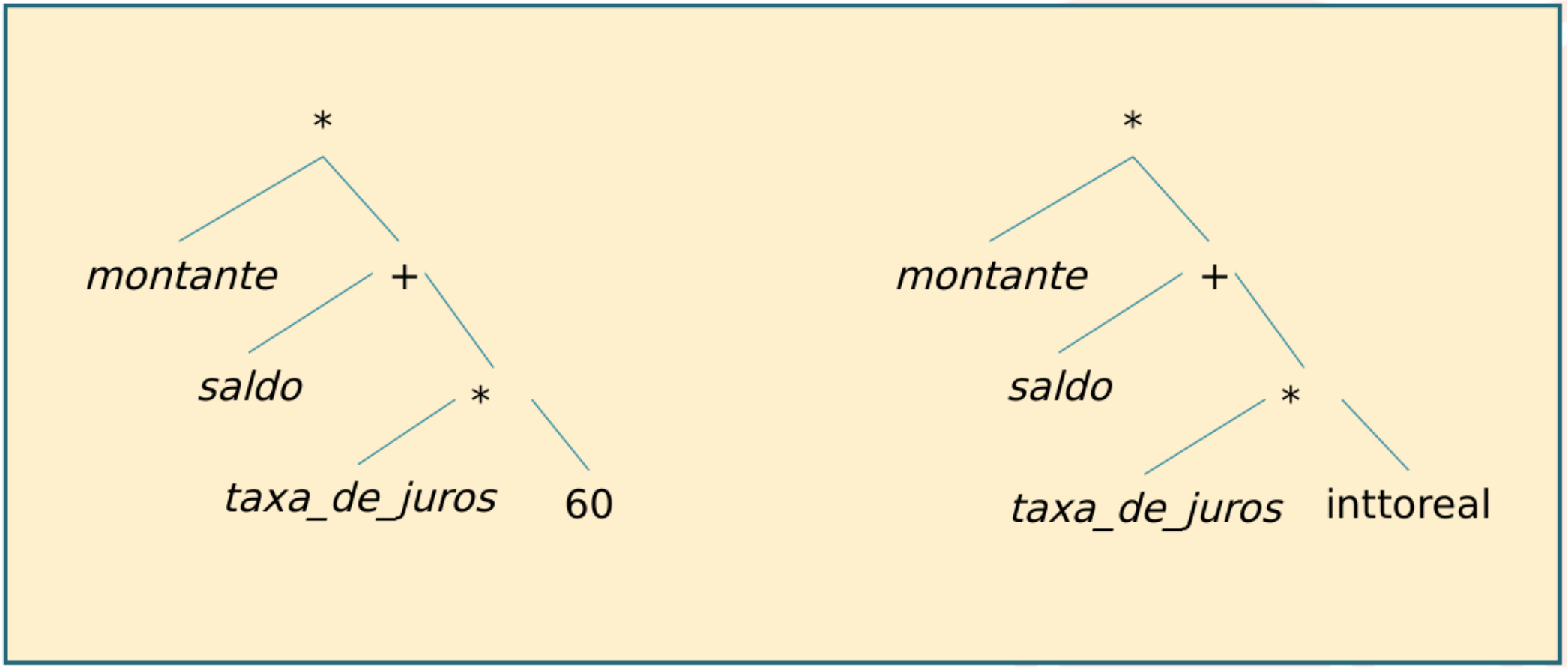
Árvore gerada para: montante := saldo + taxa_de_juros * 60

Análise Semântica

- Verifica se estruturas sintáticas, embora corretas sintaticamente, têm **significado admissível na** linguagem.
- Por exemplo, não é possível representar em uma gramática livre de contexto uma regra como *“**todo identificador deve ser declarado antes de ser usado**“*
- Um importante componente é **checagem de tipos.**
- Utiliza informações coletadas anteriormente e armazenadas na tabela de símbolos

  Considerando “**A + B”, quais os possíveis** problemas semânticos?
- Saída: **árvore de parse anotada**

Análise Semântica



Conversão de inteiro para real inserida pela análise semântica

Atividade

- Elabore um mapa conceitual sobre o conceito de compiladores destacando suas fases e funcionalidades
- Assista Engenharia elétrica: compiladores

Dúvidas