



# Paradigmas de Progamação Semana 2

- Descrevendo Sintaxe e Semântica
- Análise Léxica e Sintática

# Semântica em Linguagens de Programação

- ❖ A semântica em linguagens de programação refere-se ao significado das construções e expressões dentro do código. Ela define como o programa deve ser executado e como o resultado deve ser interpretado.
- ❖ A compreensão da semântica em linguagens de programação é fundamental para escrever código correto e eficiente, e para entender como os programas são executados.

# Semântica em Linguagens de Programação

- ❖ Semântica estática vs. semântica dinâmica:
  - ❖ **Semântica Estática:** Refere-se ao significado das construções em tempo de compilação, como o escopo das variáveis, a verificação de tipos e a análise de código.  
Exemplo: Em C#, se você tentar usar uma variável que não foi declarada dentro de um escopo, o compilador emitirá um erro.
  - ❖ **Semântica Dinâmica:** Refere-se ao significado das construções em tempo de execução, como a avaliação de expressões e o comportamento de tempo de execução.  
Exemplo: Em Python, você pode adicionar um inteiro e uma string, e o resultado será uma nova string que combina os dois valores.

# Semântica em Linguagens de Programação

## ❖ Tipagem estática vs. tipagem dinâmica:

- ❖ **Tipagem Estática:** Os tipos das variáveis são verificados em tempo de compilação. O tipo de uma variável é definido e não pode mudar durante a execução.

Exemplo: Em Java, se você declarar uma variável como int, ela sempre conterá um valor inteiro.

- ❖ **Tipagem Dinâmica:** Os tipos das variáveis são verificados em tempo de execução. O tipo de uma variável pode mudar durante a execução do programa.

Exemplo: Em JavaScript, uma variável pode conter um número em um momento e uma string em outro momento.

# Semântica em Linguagens de Programação

- ❖ Semântica de atribuição e expressões:
  - ❖ Atribuição: Define como os valores são atribuídos às variáveis e como os efeitos colaterais são tratados.  
  
Exemplo: Em C, o operador de atribuição = atribui o valor à variável.
  - ❖ Expressões: Define como as expressões são avaliadas, incluindo a ordem de avaliação e as regras de precedência dos operadores.  
  
Exemplo: Em Python,  $2 + 3 * 4$  é avaliado como 14, pois a multiplicação tem precedência sobre a adição.

# Semântica em Linguagens de Programação

- ❖ Semântica de controle de fluxo:
  - ❖ Define como as instruções condicionais (if-else) e as instruções de repetição (loops) controlam o fluxo de execução do programa.  
Exemplo: Em C++, um loop for é usado para iterar sobre uma sequência de valores.

# Semântica em Linguagens de Programação

- ❖ Semântica de funções e procedimentos:
  - ❖ Define como as funções e procedimentos são definidos, chamados e executados, incluindo o escopo das variáveis dentro de funções.  
Exemplo: Em Java, você define uma função usando a palavra-chave void para procedimentos ou o tipo de retorno desejado para funções.

# Semântica em Linguagens de Programação

- ❖ Semântica de tratamento de exceções
  - ❖ Define como as exceções são lançadas, propagadas e tratadas durante a execução do programa.
  - Exemplo: Em C#, você pode usar blocos try, catch e finally para lidar com exceções.

# Análise Semântica

---

Análise Semântica

---

Verificação de Tipos

---

Resolução de Nomes

---

Escopo de Variáveis

---

Geração de Código Intermediário

---

Verificação de Regras Semânticas

---

Detecção de Erros Semânticos

# Análise Semântica

- ❖ Em linguagens de programação é uma etapa crucial do processo de compilação ou interpretação, onde o significado do código fonte é verificado para garantir que ele esteja em conformidade com as regras semânticas da linguagem.
- ❖ Desempenha um papel fundamental na garantia de que o código fonte de um programa está corretamente estruturado e em conformidade com as regras da linguagem, preparando-o para as etapas subsequentes do processo de compilação ou interpretação.

# Análise Semântica

## ❖ Verificação de Tipos:

❖ Garante que os tipos de dados usados em expressões, atribuições e chamadas de função sejam compatíveis com as regras semânticas da linguagem.

Exemplo: Em C#, a análise semântica verifica se uma operação de adição está sendo realizada entre tipos numéricos compatíveis.

# Análise Semântica

## ❖ Resolução de Nomes:

- ❖ Mapeia os identificadores (como variáveis, funções e classes) para seus respectivos objetos no escopo do programa.

Exemplo: Em Python, a análise semântica verifica se uma variável foi declarada antes de ser usada e resolve seu escopo.

# Análise Semântica

- ❖ Escopo de Variáveis:
  - ❖ Determina a visibilidade e a acessibilidade das variáveis em diferentes partes do código. -  
Exemplo: Em C++, a análise semântica verifica se uma variável está acessível dentro do escopo de um bloco de código.

# Análise Semântica

- ❖ Geração de Código Intermediário:

- ❖ Converte o código fonte em uma representação intermediária que é mais fácil de ser otimizada e traduzida para código de máquina.

- Exemplo: Durante a análise semântica, um compilador C pode gerar uma árvore de sintaxe abstrata (AST) para o código fonte.

# Análise Semântica

## ❖ Verificação de Regras Semânticas Específicas:

- ❖ Aplica as regras semânticas específicas da linguagem, como restrições de acesso, operações válidas em tipos de dados e convenções de nomenclatura.

Exemplo: Em Java, a análise semântica verifica se uma classe abstrata é corretamente estendida por uma subclasse.

# Análise Semântica

- ❖ Geração de Código Intermediário:

- ❖ Converte o código fonte em uma representação intermediária que é mais fácil de ser otimizada e traduzida para código de máquina.

- Exemplo: Durante a análise semântica, um compilador C pode gerar uma árvore de sintaxe abstrata (AST) para o código fonte.

# Análise Semântica

- ❖ Detecção de Erros Semânticos:
  - ❖ Identifica e relata erros semânticos, como uso incorreto de variáveis, chamadas de função inválidas e operações não suportadas.  
Exemplo: Um compilador C pode detectar um erro semântico ao tentar usar uma variável que não foi declarada.