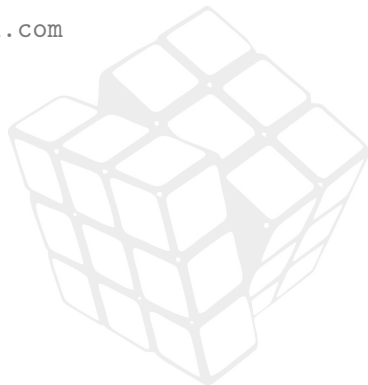


Algoritmos e Programação

Introdução à Linguagem de Programação

Gustavo Sávio
gsoprofessor@gmail.com

2017.2



Até agora nós vimos que existe uma linguagem que os computadores são capazes de compreender.

- ▶ Tal linguagem é utilizada na construção de algoritmos
- ▶ Com o intuito de ensiná-los a executarem tarefas
- ▶ Essa linguagem é denominada Linguagem de Programação

Assim como o português, as linguagens de programação possuem o objetivo de estabelecer **um meio de comunicação** eficaz.

- ▶ São constituídas por um conjunto de palavras especiais (palavras-chave)
- ▶ As palavras são associadas a um **conjunto de regras** de utilização
- ▶ As regras determinam como os algoritmos devem ser especificados para que possam ser decodificados pelo computador

As linguagens de Programação são bastante diferentes das linguagens naturais.

- ▶ A Linguagem de Programação pode ser utilizada como meio de comunicação entre pessoas?
- ▶ O foco é estabelecer a comunicação entre pessoa / computador
- ▶ As Linguagens Naturais são mais tolerantes a erros
- ▶ Erros gramaticais são facilmente superados, não impossibilitam uma conversa

- ▶ Em Linguagens de Programação a **omissão** de alguma instrução impede que a comunicação seja iniciada
- ▶ Com Linguagens Naturais nos comunicamos por meio de textos e sons
- ▶ Nas Linguagens de Programação nos expressamos através de códigos
- ▶ Que são Algoritmos escritos com uma Linguagem de programação

- ▶ Os computadores representam as informações através de dois estados
- ▶ Conhecido como sistema binário
- ▶ A representação binária utiliza os algarismos 0 e 1 chamados de dígitos binários
- ▶ 0 e 1 são os valores que um **bit** pode assumir e estão associados aos valores de tensão presentes nos circuitos elétricos do computador

Trabalhar com representação binária não é tão simples

Exemplo de representação com pseudocódigo:

```
ALGORITMO
DECLARE nota1,
        nota2,
        M : NUMÉRICO
LEIA nota1
LEIA nota2
M <= (nota1 + nota2) / 2
FIM_ALGORITMO.
```

Exemplo de representação binária:

```
10100001
10100011 10010001
           10010010
           10010011 11000001
10100100 10010001
10100100 10010010
10010011 11110011 10010001 11110001 10010010 11110010 00000010
10100010
```


- ▶ Com o intuito de tornar a tarefa de programar menos complicadas, foram criadas as Linguagens de Programação
- ▶ Mais próximas das Linguagens Naturais
- ▶ Compostas por palavras-chave
- ▶ Normalmente em inglês
- ▶ Possui **símbolos que definem os comandos e instruções** que podem se utilizadas pelo programador na elaboração do seu sistema

As linguagens com as características citadas são denominadas de **Linguagens de Alto Nível**

- ▶ Exemplos de Linguagens de Alto Nível: Java, PHP, C, Python, Ruby, etc

As Linguagens que são mais próximas da linguagem de máquina (representação binária), são chamadas de **Linguagem de Baixo Nível**

- Exemplos de Linguagem de Baixo Nível: **Assembly**

As Linguagens de Programação facilitam o trabalho dos programadores

- ▶ Entretanto os computadores só entendem Linguagem de Baixo Nível
- ▶ Como podemos executar um programa em Linguagem de Programação, de Baixo ou Alto Nível, em um computador que trabalha apenas com números binários?

Utilizando um **Tradutor**, para traduzir um programa escrito em linguagem de programação correspondente em linguagem de máquina.

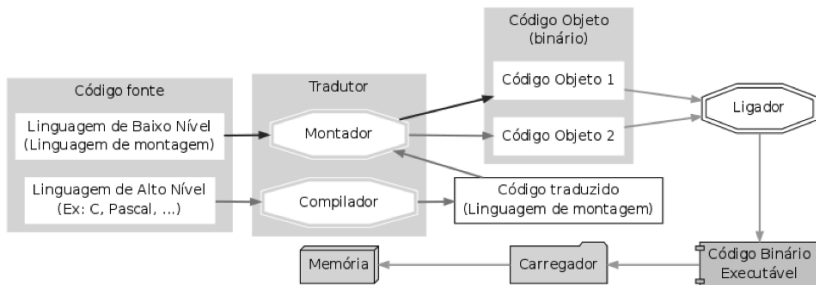
- ▶ Existem dois softwares responsáveis pelo processo de tradução:
O tradutor e o Interpretador

Existem duas categorias de Tradutores: **Montadores** e **Compiladores**

- ▶ **Montador:** Quando o processo de tradução converte um programa que em Linguagem de Montagem (Representação simbólica da linguagem de máquina, Ex: Assembly)
- ▶ Ou seja, para linguagem de máquina utilizamos o montador

- ▶ **Compilador:** Quando o processo de tradução converte um programa com Linguagem de Alto Nível para a Linguagem de Montagem
- ▶ Não existe tradução direta da linguagem de alto nível para linguagem de máquina. São necessárias várias etapas para que o *software* seja traduzido

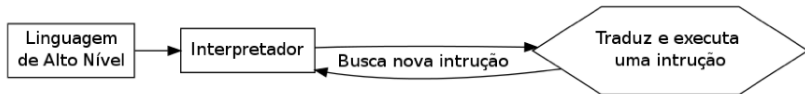
Figura: Processo de Compilação



Interpretadores além de realizar a tradução de um programa para linguagem de máquina, ainda executam as suas instruções

- ▶ Assim que traduz uma instrução ela é executada
- ▶ Gerando um ciclo de tradução e execução até o final do programa

Figura: Processo de Interpretação




Compilação vs Interpretação

- ▶ Um programa uma vez compilado pode ser executado várias vezes sem a necessidade de existir uma recompilação (Se for rodar na mesma arquitetura)
- ▶ Na Interpretação, cada vez que um programa tiver que ser executado, todo o processo de interpretação é refeito. Independente se o programa foi ou não modificado

Um Paradigma de Programação define o modo como o programador irá construir o software

Alguns exemplos de paradigmas:

- ▶ **Paradigma Imperativo** : Diz ao computador o que deve ser feito a cada momento. Ex: Máquina de Turing
- ▶ **Paradigma estruturado**: Soluciona os problemas a partir de sua quebra em problemas menores. Todo processamento utiliza três tipos de estruturas: sequencial, condicional e repetição
- ▶ **Paradigma Orientado a Objetos**: Mapeio o problema em um conjunto de objetos que se comunicam por meio de mensagens. Os objetos são estruturas que possuem um estado e comportamento

-  Formiga, A.; Júnior, J.; Sousa, B. **Introdução a Programação**.
Editora UFPB, 2014