**. Algoritmo , Automação , Progama de computador . (aula 01)**

**Algoritmo** – algoritmos são uma sequência finita de instruções para resolver um problema (não se aplica somente na area da progamação , **mas sim em diversas areas** de conhecimento ).

Exemplo = Lavar roupa suja .

Qual seria os passos para resolver esse problema ?

**Algoritmo =**

1. Colocar a roupa em um recipiente
2. Colocar um pouco de sabão e amaciante
3. Encher de água
4. Mexer tudo até dissolver o sabão
5. Deixar de molho por vinte minutos
6. Esfregar a roupa
7. Enxaguar
8. Torcer

**Automação**  - consiste em utilizar máquina(s) para executar o procedimento desejado de forma automática e semiautomática.

* Se pegarmos o exemplo de lavar roupa suja , podemos ter a ciência de que parte desse algoritmo pode ser feito por uma máquina de lavar ? , Automação é isso , nós pegamos parte do nosso algoritmo e executamos com uma máquina.

------------------------------------------------------------------------------------------------------------------------------------------------------------

**Mas o que algoritmo e automação têm a ver com a progamação de computadores ?**

**Computador:**

**. Hardware** - parte física (a máquina em si )

**. Software** - parte lógica (progamas)

. sistema operacional ( Windows , Linux , Mac )

. Aplicativos (apps de escritório , apps de câmera , navegador web)

. jogos

. utilitários (antivírus , compactador de arquivos)

. outros

**O progama de computador tem uma similaridade com o algoritmo , no contexto da computação , então podemos definir grosso modo que.**

Progamas de computador são algoritmos executados pelo computador (em linhas gerais )

**Conclusão** : o computador e uma máquina que **automatiza** a execução de **algoritmos**.

Qualquer tipo de algoritmo ? não apenas algoritmos computacionais.

- Processamento de dados

- Cálculos

**.O que é preciso para se fazer um progama de computador . (aula 02)**

Para construirmos um progama de computador nós vamos precisar de 4 coisas essenciais :

* Uma **linguagem de progamação** ; regras **léxicas** e **sintáticas** para se escrever um progama.
* Uma **IDE** : software para editar e testar o progama.
* Um **compilador** : Software para transformar o **codigo fonte** em **codigo objeto.**
* **Gerador de codigo** ou **máquina virtual** : software que permite que o progama seja executado.

**.linguagem de progamação , léxica e sintática . (aula 03)**

Linguagem de progamação - e um conjunto de regras **léxicas** (ortográfica) e **sintáticas** (gramática) para se escrever progamas.

**Léxica** diz respeito a correção das palavras “isoladas” (ortografia).

**Ex:**

**Exemplo (português) Linguagem de progamação**

Cachorro (***V***) main (***V***)

Caxorro (**X**) maim (**X**)

**Sintática** diz respeito a correção das **sentenças** (gramática).

**Exemplo (português) Linguagem de progamação**

O cachorro está com fome (***V***) x = 2 + y ;(***V***)

A cachorro está com fome (**X**) x = + 2 y ; (**X**)

**. IDE - Ambiente Integrado de Desenvolvimento . (aula 04)**

É um conjunto de softwares utilizado para a construção de progamas.

**Exemplo :**

C/C++ : Code Blocks

Java : Eclipse , NetBeans

C# : Microsoft Visual Studio

Funcionalidades de uma IDE

***. edição de códigos fonte ( indentação, autocompletar , destaque de palavras , etc. )***

***. depuração e testes***

***. construção do produto final (build)***

***. Sugestão de modelos (templates )***

***. auxiliar em várias tarefas do seu projeto***

***. etc***.

**. Compilação e interpretação , codigo fonte e objeto Máquina virtual. (aula 05)**

**Codigo fonte** = é aquele escrito pelo programador em linguagem de progamação.

Esses códigos, entretanto, não são entendidos pelo computador e nem pelo sistema operacional , ele tem que ser convertido em um codigo que possa ser entendido e executado:

**Compilação**

**Gerador de codigo : construção (build)**

**Compilador : compilação**

**Codigo fonte**

**Codigo executável**

**Codigo objeto**

Geração de código

Análise léxica

+

Análise sintática

Compilação e um processo que transforma o codigo fonte no que chamamos de codigo objeto , esse processo e feito por um software chamado compilador . durante o processo de compilação o compilador faz a análise léxica e sintática , se tiver algum erro léxico ou sintático no meu codigo fonte , a compilação e abortada , e o programador e obrigado a corrigir os erros que porventura existam.

Uma vez o que o código fonte passou pela verificação léxica e sintática e se tornou o codigo objeto , esse codigo objeto terá que passar por um gerador de codigo , para que seja feito um processo de construção ou build , sendo assim o codigo podendo ser executado pelo sistema operacional.

No entanto , nem toda linguagem utiliza esse processo , tambem há outro tipo de processo chamado :

**Interpretação**

**Interpretador : interpretação**

Codigo fonte

Análise léxica + análise sintática + geração de codigo SOB DEMANDA

O processo interpretador ele vai ler o codigo fonte e vai realizar a análise léxica , análise sintática , geração de codigo (SOB DEMANDA ) , ou seja , gradualmente o interpretador vai lendo o codigo fonte traduzindo-o para o codigo executável e executando .

Esse modelo de interpretação e execução e utilizado por várias linguagens , tais como : PHP, JavaScript, Python, Ruby.

**Abordagem Hibrida**

**Máquina virtual : interpretação**

**Compilador : precompilação**

Codigo fonte

Bytecode

**Geração de codigo**

**SOB DEMANDA**

**Análise léxica**

**+**

**Análise sintática**

Abordagem hibrida e quando eu tenho o codigo fonte , esse codigo fonte passa por uma pre-compilação que faz a análise léxica e sintática , gerando um codigo objeto que recebe o nome de Bytecode (o Bytecode e um codigo já pre-compilado) , porém ao invés de passar por um gerador de codigo e virar um codigo executável , esse bytecode vai passar por uma máquina virtual , que faz a interpretação desse bytecode e aí sim teremos a geração de códigos sob demanda e a execução do progama.

Essa abordagem hibrida de compilação , interpretação e execução e adotada por algumas linguagens , tais como : Java (JVM) , C# (Microsoft .NET Framework)

Vantagens

**Compilação :**

. velocidade do progama

. auxílio do compilador antes da execução

**Interpretação :**

. flexibilidade de manutenção do aplicativo em produção

. expressividade da linguagem

. codigo fonte não precisa ser recompilado para rodar em plataformas diferentes

Diagrama

Descrição gerada automaticamente

**Linguagem compilada**

**Linguagem interpretadas**

**Linguagem hibrida**