Capítulo 3 - Variáveis, constantes e tipos de dados.

O que são variáveis constantes?

Programas de computador utilizam os recursos de hardware mais básicos para executar algoritmos. Enquanto o processador executa os cálculos, a memória é responsável por armazenar dados e servi-los ao processador. O recurso que nós utilizamos em nossos programas para escrever e ler dados da memória do computador é conhecido como **variável**, que é simplesmente **um espaço na memória o qual reservamos e damos um nome**. Por exemplo, podemos criar uma variável chamada "idade" para armazenar a idade de uma pessoa. Você pode imaginar uma variável como uma gaveta "etiquetada" em um armário. Chamamos este espaço alocado na memória de **variável**, porque o valor armazenado neste espaço de memória pode ser alterado ao longo do tempo, ou seja, o valor ali alocado é "variável" ao longo do tempo. Diferente das **constantes**, que é um espaço reservado na memória para armazenar um valor que não muda com o tempo. Por exemplo, o valor PI (3.14159265359…) que nunca vai mudar.

Como funciona uma variável em um algoritmo

Para não restar dúvidas quanto ao funcionamento de uma variável, vamos ver como elas funcionam em um algoritmo:

```
Algoritmo "Teste de Variável"
Declaração das variáveis
   nome : Texto
Inicio
   nome <- "João"
   imprimir(nome)
   nome <- "Maria"
   imprimir(nome)
```

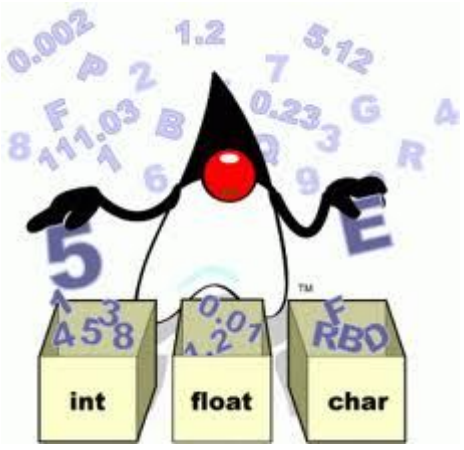
```
Fim
```

Neste algoritmo, declaramos uma variável chamada *nome* do tipo "Texto". Inicialmente armazenamos o texto "João" na variável nome, e mandamos imprimir na tela o valor desta variável. Neste momento aparece na tela o texto "João", em seguida alteramos o valor da variável para "Maria" neste momento o texto "João" é apagado da variável (memória) e em seu lugar é armazenado o texto "Maria". Em seguida, mandamos imprimir na tela novamente o valor da variável, então aparece na tela o texto "Maria".

Deu pra perceber como o valor da variável *nome* pode ser alterado com o tempo?

Tipos de dados

Para otimizar a utilização da memória, nós definimos um tipo de dados para cada variável. Por exemplo, a variável *nome*, deve armazenar textos, já a variável *idade* deve armazenar apenas números inteiros (sem casa decimal), na variável *sexo* podemos armazenar apenas um caractere ("M" ou "F"). Seria correto armazenarmos o valor "M" na variável *idade*? Não né? Por isso devemos especificar em nossos algoritmos o tipo de cada variável.



Podemos classificar os tipos de dados em basicamente duas categorias, os tipos de dados primitivos e os tipos de dados customizados.

Tipos de dados primitivos

Existem quatro tipos de dados primitivos, algumas linguagens subdividem estes tipos de dados em outros de acordo com a capacidade de memória necessária para cada variável, mas no geral, os tipos de dados primitivos são:

- **INTEIRO:** Este é o tipo de dados para valores numéricos negativos ou positivos, sem casas decimais. Por exemplo uma variável *idade*.
- **REAL:** Este é o tipo de dados para valores numéricos negativos ou positivos, com casas decimais. Por exemplo uma variável *peso*.

- **LÓGICO:** Este tipo de dados pode assumir apenas dois valores VERDADEIRO ou FALSO. Também é conhecido como **booleano**. Reserva apenas um bit na memória, onde o valor 1 representa VERDADEIRO e o valor 0 representa FALSO. Por exemplo uma variável *status_da_lampada* (acesa ou apagada).
- **TEXTO:** Tipo de dados para variáveis que armazenam textos. Por exemplo uma variável *nome*.

Algumas linguagens de programação dividem esses tipos primitivos de acordo com o espaço necessário para os valores daquela variável. Na linguagem Java por exemplo, o tipo de dados inteiro é dividido em 4 tipos primitivos: byte, short, int e long. A capacidade de armazenamento de cada um deles é diferente.

- **byte**: é capaz de armazenar valores entre -128 até 127.
- **short**: é capaz de armazenar valores entre – 32768 até 32767.
- **int**: é capaz de armazenar valores entre –2147483648 até 2147483647.
- **long**: é capaz de armazenar valores entre –9223372036854775808 até 9223372036854775807.

Mas essa divisão é uma particularidade da linguagem de programação. O objetivo é otimizar a utilização da memória. Em algumas linguagens de programação não é necessário especificar o tipo de dados da variável, eles são identificados dinamicamente. Porém, é necessário informar o tipo de dados de cada variável em algoritmos.

Tipos de dados customizados

A partir dos tipos de dados primitivos podemos criar outros tipos de dados utilizando uma combinação de variáveis. São estruturas de dados, classes, vetores, matrizes, etc.

```
class Carro {

    String tipo;
    String cor;
    String placa;
    int numPortas;

    void setCor(String c){
        cor = c;
    }

    String getCor(){
        return cor;
    }
}
```

Por exemplo, uma classe chamada *Carro* é um tipo de dados que agrupa outras variáveis básicas como **marca**, **cor**, **ano**, **modelo**, etc.

Um *vetor* é um agrupamento de variáveis do mesmo tipo, uma *matriz* é um agrupamento de vetores. Enfim, a base de todos os tipos de dados são os tipos de dados primitivos, independente da linguagem de programação.

Claro, em **Programação Orientada a Objetos** há todo um conceito para a criação de *classes* que, além de *atributos* também tem *operações*, o estudo de *estruturas de dados* também vai muito além de apenas formar tipos de dados a partir de outros. Mas não se preocupe com isso por agora. Apenas entenda que são tipos de dados formatos a partir de outros tipos de dados.

Diferente dos tipos de dados primitivos que já são implementados internamente pelas linguagens de programação, esses tipos de dados são criados pelo programador. Enfim, neste e-book não vamos utilizar classes. Os tipos de dados customizados que vamos aprender serão vetores e matrizes, assuntos do capítulo 9.

Espero que você tenha entendido esses dois assuntos, pois saber como funcionam as variáveis/constantes e os tipos de dados é de suma importância para você se tornar um bom programador. Releia quantas vezes forem necessárias para entender bem esse assunto.

No próximo capítulo você vai aprender sobre os três tipos de operadores que são utilizados em programação. Ok?

Mas antes vou deixar um exercício mental para você pensar. Olhe para qualquer objeto que esteja perto de você e identifique as suas características, para cada uma delas pense no tipo de dados que você utilizaria se fosse utilizar essa informação no seus software.

Por exemplo, estou olhando agora para o meu notebook e identificando algumas características nele, ele tem cor (texto), teclas (caracteres), botões de mouse para click (booleano, ou seja, pode ter dois estados "clicado" ou "não clicado"), tela (acesa ou apagada), wifi (ligado ou desligado), etc. Esse é um exercício mental que vai facilitar a sua visão sobre manipulação de dados nos seus algoritmos.