**JAVA**

**Variáveis:** Um espaço na memória do computador onde se pode guardar valores.

Existem 4 tipos:  
- Instância: objeto  
- Classe: classe  
- Local: dentro dos métodos  
- Parâmetro: Na assinatura do método

**Padrão de definição:**<?visibilidade?><?modificador?> ***tipo nome***<?=valorInicial?>;  
**Visibildiade:** public, protected e private.  
**Modificador:** static e final.  
 Final: significa que a variável é uma constante. Ou seja, o valor não vai mudar.  
**Tipo:** tipo de dado.  
**Nome:** nome que é dado a variável.  
**Valor Inicial:** Valor inicial dado a variável caso deseje.

O que está entre interrogação é opcional, pode ser criado uma variável sem definir.

**Regras para a criação de variáveis:**

* Não devem começar com números;
* $ e \_ devem ser evitados;
* São case sensitive, ou seja, se estiver maiúsculo ou minúsculo faz diferença;
* Sem espaços;
* Não pode ser as palavras reservadas

**Boas práticas:**

* Sempre começar com letra minúscula;
* Nomes expressivos;
* Notação camelCase;
* Quando constante (final), maiúscula e separada po “\_”.Ex: ***INT FINAL*** ESSE\_EH\_UM\_EXEMPLO = 5;

**Tipos de Dados**

São os valores e consequentemente operações que as variáveis podem assumir e sofrer, respectivamente.

**Tipificação:**

* Estática (**static,** *forte)* vs Dinamica (**dinamic,** *fraca*)
* Primitivo vs Composto

**Opção de tipos:**

* Textual: Strings
* Numeral (inteiros ou fracionados, int e double)
* Lógico (boleanos, true or false)
* Objeto

**Operadores Aritméticos**

São símbolos especiais quais são capazes de realizar ações especificas em um, dois ou mais operandos e, em seguida, retornar um resultado.

**Tipos**:

* Pós fixado: exp++ e exp— (primeiro avalia a expressão para depois somar ou subtrair);
  + Int j = k-- ***=>*** j = k; k = k- 1;
* Prefixado: ++exp e –exp (primeiro soma ou subtrai, para depois avaliar a expressão);
  + Int i = ++k ***=>***  i = k+1
* Aritmético: +, -, \*, / e %
* Atribuição: =, +=, -=, /=, %=

**Conversões (Casting)**

É a transformação de uma determinada variável de tipo menos específico para um tipo mais específico ou vice-versa.

**Tipos:**

* Upcast (promoção de um menor capacidade de armazenamento para maior) – Implícito
* Downcast (rebaixamento de um tipo de dado maior para uma variável de menor capacidade) - Explicito

**Métodos**

É uma porção de código (sub-rotina) que é disponibilizada por uma classe. Este é executado quanto é feito uma requisição a ele. São responsáveis por definir e realizar um determinado comportamento. Função. Deve ser criado dentro de uma classe.

<?visibilidade?><?tipo?><?modificador?> ***retorno nome*** *(*<?=parametros?>)<?exceções?> *corpo*

O que está entre interrogação é opcional, pode ser criado uma variável sem definir.

**Visibildiade:** public, protected e private.  
**Tipo:** concreto ou abstrato;  
**Modificador:** static e final.  
 Final: significa que a variável é uma constante. Ou seja, o valor não vai mudar.  
**Retorno:** Tipo de dado ou void (vazio); **Nome:** nome que é dado ao método (toda regra para definição de varivável se aplica para métodos também.  
**Parâmetro:** parâmetros que pode receber;  
**Exceções:** exceções que podem lançar;  
**Corpo:** código que possui ou vazio.

***public String getName() {...}****visibilidade tipo nome corpo(código)*

***public static R N(P){...}***

**mensagem = chamar a função**

**Sobrecarga**

É a capacidade de definir métodos para diferentes contextos, mas preservando seu nome. Ou seja, vai ter o mesmo nome mas vai conseguir se comportar de forma diferente dependendo do contexto. Como? Mudando a lista de parâmetros.

**Operadores Relacionais**

**Igualdade:** *==***Diferença:** *!=***Maior:** *>***Maior igual:** *>=***Menor:** *<***Menor igual:** *<=*

**Xor = só é true quando for oposto**

**Controle de Fluxo**

**Decisão:** if, if-else, switch e operador ternário  
**Repetição:** for, while, do while  
**Interrupção:** Break, continue, return

**Switch:** Utilizar quando for valores exatos, quando estiver analisando uma variável só por exemplo  
**If:** Utilizar com expressões booleanas

**Estruturas de Repetição e Arrays em Java**