



#digitalReskilling
Front-End Developer

Algoritmia e Programação

Tipos de Dados



Conteúdo

- Variáveis e Constantes
- Tipos de Dados Primitivos
- Operadores
- Casting

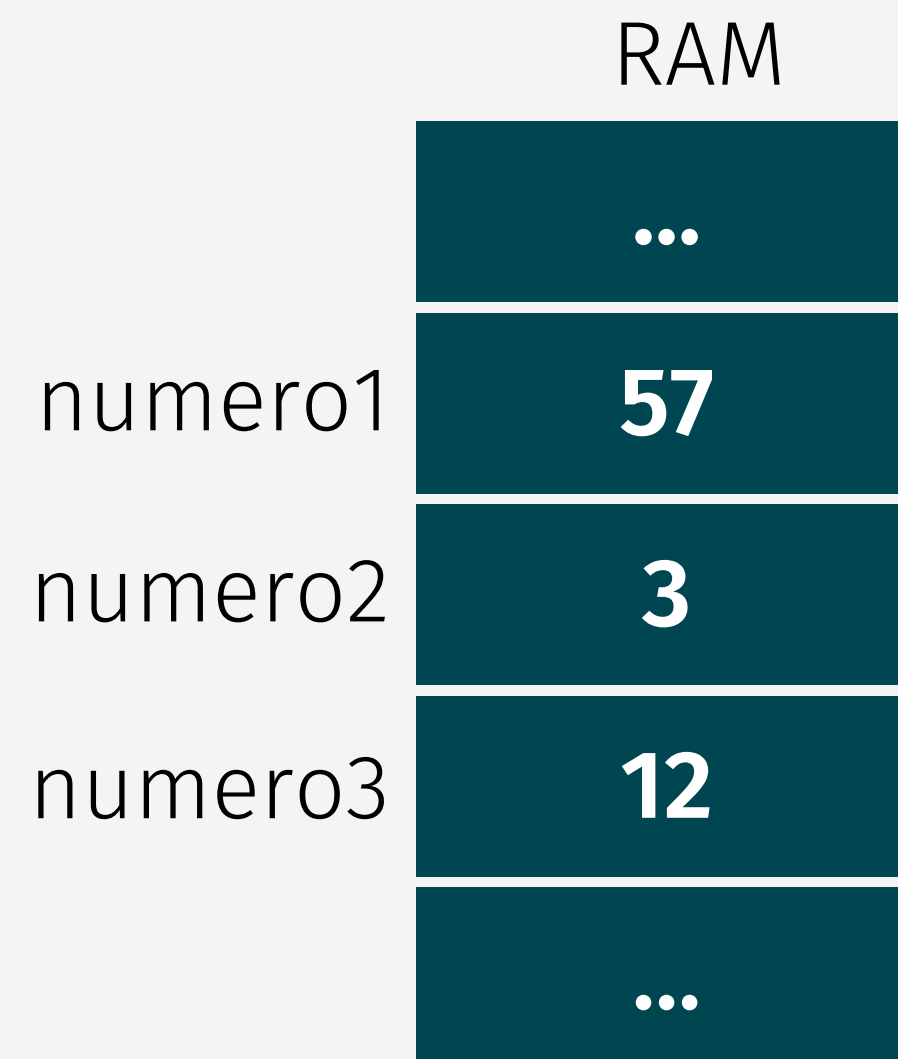
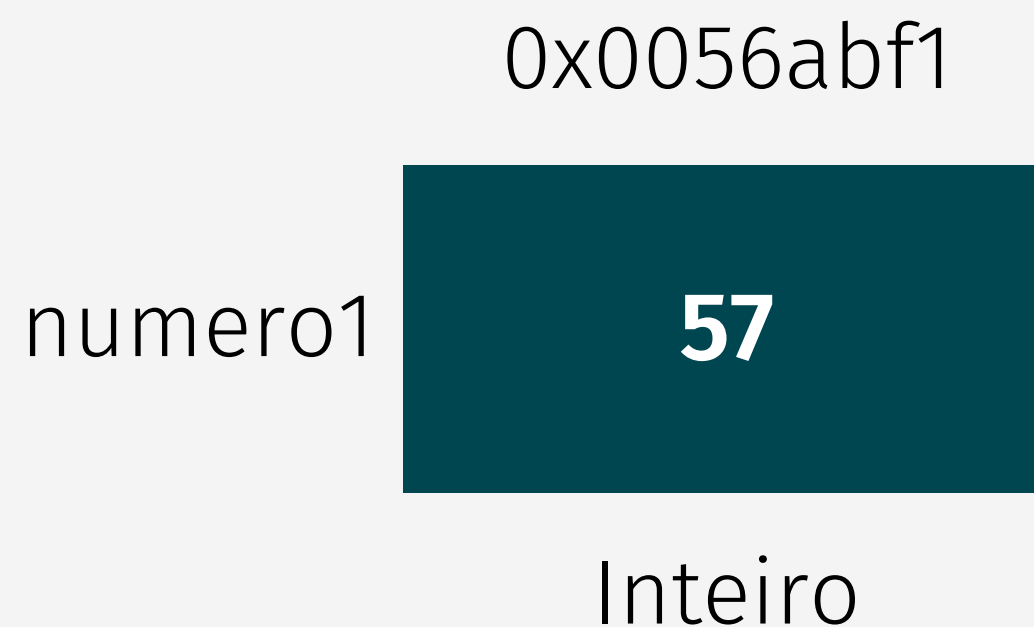
Variáveis



- Local de armazenamento de um valor que não é fixo ao longo da execução do programa.
- As variáveis são uma referência (**nome**) definida pelo programador a uma posição de memória (**endereço**) de modo a conter um **valor** com um **tipo**.
- O seu tipo é definido antes de ser usada pois indica o número de bytes (espaço) que irão ser utilizados.

Variáveis

- Quando uma variável é declarada e associada a um valor, 4 atributos fundamentais são considerados:
 - Nome
 - Tipo
 - Valor (Conteúdo)
 - Endereço



Tipos de Dados

- As linguagens de programação permitem lidar com diferentes **tipos de dados**.
- O tipo condicional:
 - O **tamanho** que irá ocupar em memória.
 - O **intervalo de valores** que podem ser armazenados.
 - O **conjunto de operações** que podem ser realizadas sobre a variável



Tipos de Dados

Tipos primitivos

- Números inteiros
- Números reais
- Carácter
- Cadeias de caracteres
- Booleano

Tipos não primitivos (complexos)

- Tipos indexados mono e multidimensionais



Tipos de Dados - Numéricos

- Esta família é constituída pelos diferentes tipos numéricos.
- As principais subdivisões deste tipo são:
 - Inteiros - números inteiros (negativos, zeros e positivos). Exemplo: 10, -15, 2023
 - Reais - números reais (em que figuram partes decimais). Exemplo: 1.5, -14.2, 0.25
- Nota: Em programação o ponto é utilizado como sinal decimal.



Tipos de Dados - Caracteres

- Os dados do tipo **char** armazenam (num byte) um único caractere.
 - Não permite armazenar cadeias de caracteres (strings).
- Delimita-se com `'...'` (plicas) e não `"..."` (aspas).
- Corresponde um valor (inteiro) da tabela de ASCII.
 - É o tipo inteiro mais pequeno na linguagem, mas é tratado como um tipo de dados diferente.
 - Exemplo: `'A' = 65`

Tabela ASCII

| Decimal | Hex | Char | Decimal | Hex | Char | Decimal | Hex | Char | Decimal | Hex | Char |
|---------|-----|------------------------|---------|-----|---------|---------|-----|------|---------|-----|-------|
| 0 | 0 | [NULL] | 32 | 20 | [SPACE] | 64 | 40 | @ | 96 | 60 | ` |
| 1 | 1 | [START OF HEADING] | 33 | 21 | ! | 65 | 41 | A | 97 | 61 | a |
| 2 | 2 | [START OF TEXT] | 34 | 22 | " | 66 | 42 | B | 98 | 62 | b |
| 3 | 3 | [END OF TEXT] | 35 | 23 | # | 67 | 43 | C | 99 | 63 | c |
| 4 | 4 | [END OF TRANSMISSION] | 36 | 24 | \$ | 68 | 44 | D | 100 | 64 | d |
| 5 | 5 | [ENQUIRY] | 37 | 25 | % | 69 | 45 | E | 101 | 65 | e |
| 6 | 6 | [ACKNOWLEDGE] | 38 | 26 | & | 70 | 46 | F | 102 | 66 | f |
| 7 | 7 | [BELL] | 39 | 27 | ' | 71 | 47 | G | 103 | 67 | g |
| 8 | 8 | [BACKSPACE] | 40 | 28 | (| 72 | 48 | H | 104 | 68 | h |
| 9 | 9 | [HORIZONTAL TAB] | 41 | 29 |) | 73 | 49 | I | 105 | 69 | i |
| 10 | A | [LINE FEED] | 42 | 2A | * | 74 | 4A | J | 106 | 6A | j |
| 11 | B | [VERTICAL TAB] | 43 | 2B | + | 75 | 4B | K | 107 | 6B | k |
| 12 | C | [FORM FEED] | 44 | 2C | , | 76 | 4C | L | 108 | 6C | l |
| 13 | D | [CARRIAGE RETURN] | 45 | 2D | - | 77 | 4D | M | 109 | 6D | m |
| 14 | E | [SHIFT OUT] | 46 | 2E | . | 78 | 4E | N | 110 | 6E | n |
| 15 | F | [SHIFT IN] | 47 | 2F | / | 79 | 4F | O | 111 | 6F | o |
| 16 | 10 | [DATA LINK ESCAPE] | 48 | 30 | 0 | 80 | 50 | P | 112 | 70 | p |
| 17 | 11 | [DEVICE CONTROL 1] | 49 | 31 | 1 | 81 | 51 | Q | 113 | 71 | q |
| 18 | 12 | [DEVICE CONTROL 2] | 50 | 32 | 2 | 82 | 52 | R | 114 | 72 | r |
| 19 | 13 | [DEVICE CONTROL 3] | 51 | 33 | 3 | 83 | 53 | S | 115 | 73 | s |
| 20 | 14 | [DEVICE CONTROL 4] | 52 | 34 | 4 | 84 | 54 | T | 116 | 74 | t |
| 21 | 15 | [NEGATIVE ACKNOWLEDGE] | 53 | 35 | 5 | 85 | 55 | U | 117 | 75 | u |
| 22 | 16 | [SYNCHRONOUS IDLE] | 54 | 36 | 6 | 86 | 56 | V | 118 | 76 | v |
| 23 | 17 | [END OF TRANS. BLOCK] | 55 | 37 | 7 | 87 | 57 | W | 119 | 77 | w |
| 24 | 18 | [CANCEL] | 56 | 38 | 8 | 88 | 58 | X | 120 | 78 | x |
| 25 | 19 | [END OF MEDIUM] | 57 | 39 | 9 | 89 | 59 | Y | 121 | 79 | y |
| 26 | 1A | [SUBSTITUTE] | 58 | 3A | : | 90 | 5A | Z | 122 | 7A | z |
| 27 | 1B | [ESCAPE] | 59 | 3B | ; | 91 | 5B | [| 123 | 7B | { |
| 28 | 1C | [FILE SEPARATOR] | 60 | 3C | < | 92 | 5C | \ | 124 | 7C | |
| 29 | 1D | [GROUP SEPARATOR] | 61 | 3D | = | 93 | 5D |] | 125 | 7D | } |
| 30 | 1E | [RECORD SEPARATOR] | 62 | 3E | > | 94 | 5E | ^ | 126 | 7E | ~ |
| 31 | 1F | [UNIT SEPARATOR] | 63 | 3F | ? | 95 | 5F | _ | 127 | 7F | [DEL] |

Tipos de Dados - Cadeia de Caracteres

- Os caracteres podem também ser agrupados formando cadeias de caracteres (strings).
- Exemplo: "Hoje é dia 4".



Dados Lógicos ou Booleanos

- Este é um tipo de dados utilizado com muita frequência em algoritmia e programação.
- Caracteriza-se por admitir apenas um de dois valores possíveis: **verdadeiro** (true) ou **falso** (false).



Tipos de Dados - Qualificadores

- **signed** e **unsigned** - Quando o qualificador **unsigned** é utilizado, o número é sempre positivo e, quando é usado o **signed**, o número pode ser positivo ou negativo.
 - Se o qualificador não for mencionado, por defeito é atribuído o **signed**.
 - Só podem ser usados com os tipos **int** e **char**.



Tipos de Dados - Qualificadores

- **short** e **long** - Quando **short** é usado o intervalo do tipo de dados é reduzido. Quando **long** é usado o intervalo de dados é aumentado.
- O tipo **int** pode usar ambos os qualificadores, **double** só pode usar **long**.
- Eles não podem ser usados com **char** e **float**.



Tipos de Dados - Qualificadores

| Tipo | Bytes | Intervalo |
|-------------------------------|-------|--|
| char ou signed char | 1 | -128 - 127 |
| unsigned char | 1 | 0 - 255 |
| int ou signed int | 4 | -2147483648 - 2147483647 |
| unsigned int | 4 | 0 - 4294967295 |
| short int ou short signed int | 2 | -32768 - 32767 |
| unsigned short int | 2 | 0 - 65535 |
| long int ou signed long int | 8 | -9223372036854775808 - 9223372036854775807 |
| unsigned long int | 8 | 0 - 18446744073709551615 |
| float | 4 | 1.175494e-38 - 3.402823e+38 |
| double | 8 | 2.225074e-308 - 1.797693e+308 |
| long double | 16 | 3.362103e-4932 - 1.189731e+4932 |

Tipos de Dados

Exercício



- Identifique os seguintes tipos de dados:

1. 613

2. 'R'

3. 1

4. 613.0

5. Falso

6. "613"

7. -3.012×10^{15}

8. -613

9. "Verdadeiro"

10. 17×10^{12}

11. '6'

12. "Fim do Exercício"

Variáveis - Declaração

- Uma variável é declarada indicando o seu **tipo** e o seu **nome**.
 - Sintaxe: **tipo nomeVariavel;**
- Podem ser declaradas várias variáveis do mesmo tipo na mesma instrução.
 - Sintaxe: tipo nomeVariavel1, nomeVariavel2, ...;
- As variáveis têm que se declaradas antes da sua utilização.

```
int idadeAluno1, idadeAluno2;
```


Variáveis - Nome - Regras

- Pode conter letras, dígitos e o caractere _ (underscore).
- O **primeiro** caractere não pode ser um dígito.
- Não se podem usar palavras reservadas.
- *Case sensitive.*

| Válido |
|---|
| Nome, _nome, nome, nome1, nome_, nome_1 |
| Inválido |
| 1nome, nome_+, +nome, return |

Variáveis - Nome - Cuidados

- Usar nomes descritivos
 - `n`
 - `nome`
- Separar nomes que utilizam mais do que uma palavra.
 - `nomecliente`
 - `nome_cliente`
 - `nomeCliente`
- Evitar nomear totalmente em maiúsculas.
 - Prática tradicionalmente utilizada para nomear constantes.
- Não iniciar o nome de variáveis com o underscore.

Variáveis - Atribuição

- A atribuição é realizada com recurso ao operador = (igual).
- Sintaxe: variável = expressão;
- Podemos atribuir o valor de uma variável a outra variável.
- As variáveis podem ser inicializadas quando declaradas.

```
int idadeAluno1 = 20, idadeAluno2;  
idadeAluno2 = idadeAluno1;
```

Variáveis - Atribuição

- É possível atribuir o mesmo valor a várias variáveis simultaneamente.

```
int a, b, c;
```

```
a = 0;
```

```
b = 0;
```

```
c = 0;
```

=

```
int a, b, c;
```

```
a = b = c = 0;
```

Constantes

- Armazenam dados que permanecerão inalteráveis ao longo do programa.
- Têm de ser declaradas antes da sua utilização.
- Os nomes devem ser escritos em maiúsculas.
 - Não é obrigatório mas é boa prática.
- Vantagens:
 - Melhoram legibilidade.
 - Facilitam manutenção.
 - Previnem erros.




Constantes

- "Variável" com a denotação *final*, evita erros quando tentamos trocar o valor destas "variáveis".
- Boa prática declarar no início do programa.

```
public static void main(String[] args) {  
  
    final double PI = 3.1415926;  
    final double IVA = 1.23;  
  
    System.out.println("O valor de Pi é: " + PI);  
    System.out.println("O valor do IVA é: " + IVA);  
}
```


Operadores Aritméticos



| Operação | Descrição | Exemplo | Resultado |
|----------|------------------|-----------|-----------|
| + | Soma | $21 + 4$ | 25 |
| - | Subtração | $21 - 4$ | 17 |
| * | Multiplicação | $21 * 4$ | 84 |
| / | Divisão | $21 / 4$ | 5 |
| % | Resto da Divisão | $21 \% 4$ | 1 |

- As operações entre inteiros devolvem um inteiro.
- A divisão entre 21 e 4 não resulta em 5.25 mas sim 5.
- $21/3$ devolve o **quociente** (5).
- $21\%4$ devolve o **resto** (1).

Operadores Aritméticos



| Operação | Descrição | Exemplo | Resultado |
|----------|------------------|------------|-----------|
| + | Soma | $21.1 + 4$ | 25.1 |
| - | Subtração | $21.1 - 4$ | 17.1 |
| * | Multiplicação | $21.1 * 4$ | 42.2 |
| / | Divisão | $42.2 / 4$ | 21.1 |
| % | Resto da Divisão | | |

- Qualquer operação em que pelo menos um dos operandos seja real, produz um resultado do tipo real.

Operadores Aritméticos

| Operação | Equivale a... | Descrição |
|----------------------------|-------------------------------|-------------------------|
| <code>a += b</code> | <code>a = a + b</code> | Soma |
| <code>a -= b</code> | <code>a = a - b</code> | Subtração |
| <code>a *= b</code> | <code>a = a * b</code> | Multiplicação |
| <code>a /= b</code> | <code>a = a / b</code> | Divisão |
| <code>a %= b</code> | <code>a = a % b</code> | Resto |
| <code>a <<= b</code> | <code>a = a << b</code> | Deslocamento (esquerda) |
| <code>a >>= b</code> | <code>a = a >> b</code> | Deslocamento (direita) |
| <code>a &= b</code> | <code>a = a & b</code> | E |
| <code>a ^= b</code> | <code>a = a ^ b</code> | OU (exclusive) |
| <code>a = b</code> | <code>a = a b</code> | OU (inclusive) |

Operadores de Incremento/Decremento

- ++ incrementa 1:
 - Pré-incremento: ++variável;
 - Pós-incremento: variável++;
- -- decrementa 1:
 - Pré-decremento: --variável;
 - Pós-decremento: variável--;
- Utilizado sobre variáveis, não sobre valores ou expressões.
 - **(ano + i)++; //incorreto**
 - **idade++; //correto**

Operadores de Incremento/Decremento

| | | Resultado ¹ | |
|-------------------------------|---|------------------------|-------|
| Operador | Instruções equivalentes | final | valor |
| <code>final = valor++;</code> | <code>final = valor;</code> <code>valor += 1;</code> | 1 | 2 |
| <code>final = ++valor;</code> | <code>valor += 1;</code> <code>final = valor;</code> | 2 | 2 |
| <code>final = valor--;</code> | <code>final = valor;</code> <code>valor -= 1;</code> | 1 | 0 |
| <code>final = --valor;</code> | <code>valor -= 1;</code> <code>final = valor;</code> | 0 | 0 |

Assumindo valor = 1;

Exercícios



1. Escreva um programa que faça a soma de dois números introduzidos pelo utilizador e apresente o resultado dessa soma.
2. Escreva um programa que subtraia dois números introduzidos pelo utilizador e multiplique o resultado pelo primeiro número. No final apresente o resultado.
3. Escreva um programa que calcule a área e o perímetro de uma circunferência. Nota $\text{perímetro} = 2\pi r$ e $\text{área} = \pi r^2$
4. Escreva um programa que leia o ano de nascimento de uma pessoa e imprima a idade que ela terá em 2026.
5. Escreva um programa que leia um número inteiro e imprima o seu antecessor e o seu sucessor.

Exercícios

1. Escreva um programa que faça a soma de dois números introduzidos pelo utilizador e apresente o resultado dessa soma.



Exercícios



1. Escreva um programa que faça a soma de dois números introduzidos pelo utilizador e apresente o resultado dessa soma.

- declarar real numero1, numero2, soma;
- utilizador introduz numero1;
- utilizador introduz numero2;
- $soma = numero1 + numero2$;
- escrever soma;

Exercícios

2. Escreva um programa que faça a soma de dois números introduzidos pelo utilizador e apresente o resultado dessa soma. No final apresente o resultado.



Exercícios



2. Escreva um programa que subtraia dois números introduzidos pelo utilizador e multiplique o resultado pelo primeiro número. No final apresente o resultado.

- declarar real numero1, numero2, final
- ler numero1
- ler numero2
- $\text{final} = (\text{numero1} - \text{numero2}) * \text{numero1};$
- `//subtração=numero1-numero2`
- `//multi = subtração * numero1`
- escrever final



#digitalReskilling
Front-End Developer

Algoritmia e Programação

Tipos de Dados

