

Tec. de Desenvolvimento de Algoritmos

- ✓ ALGORITMOS?
- ✓ INTERPRETADOR/COMPILADOR
- ✓ TIPOS DE DADOS
- ✓ VARIÁVEIS
- ✓ ENTRADA E SAÍDA
- ✓ FUNÇÕES MATEMÁTICAS

Algoritmo?

- ✓ Em computação pode ser definido como uma sequência de instruções ou operações básicas, cuja execução, em tempo finito resolve um problema computacional.
- ✓ A partir do **algoritmo** será construído um **programa**, que estará escrito em alguma **linguagem de programação** para que possa ser executado em um computador.
- ✓ Como o algoritmo descreve uma lógica geral de solução, poderá ser programado em diferentes linguagens de programação...



Algoritmo – Representação em Pseudocódigo

```
algoritmo Nome_do_algoritmo
  inicio
    declaração de variáveis
    corpo do algoritmo
  fim
```



```
algoritmo Nome_do_algoritmo
  declaração de variáveis
  inicio
    corpo do algoritmo
  fim
```

Como a máquina entende os códigos?

- ✔ Para que o computador "entenda" um programa escrito em uma linguagem (de alto nível) é necessário um meio de **tradução** entre a linguagem de alto nível utilizada no programa e a linguagem de máquina.
- ✔ Para essa tarefa temos basicamente dois métodos:
 - Compilador
 - Interpretador

Interpretador

Traduz e faz a checagem da sintaxe e envia para execução, instrução por instrução.

Precisa estar presente todas as vezes que vamos executar o programa e o processo acima é repetido.



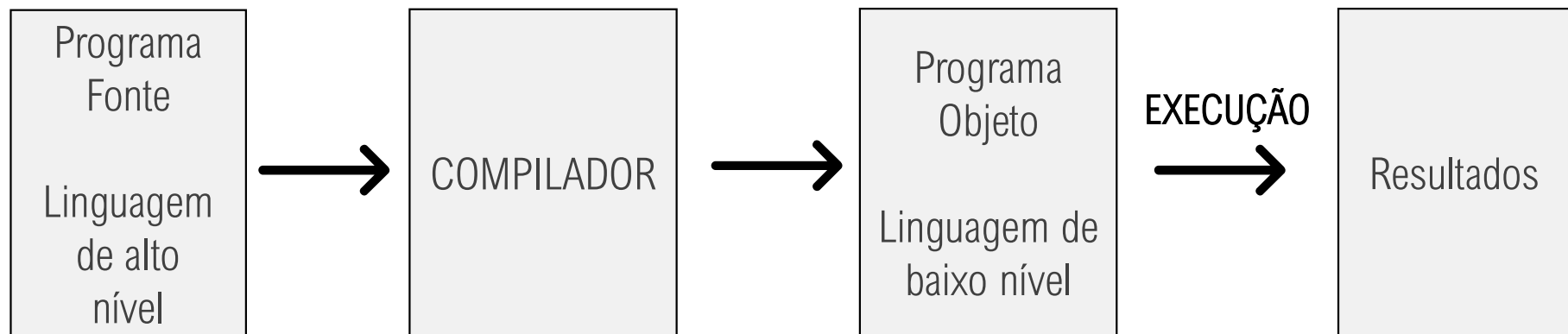
Vantagem: consome menos memória

Desvantagem: execução mais lenta

Exemplo: Uma página html é interpretada pelo Navegador.

Compilador

Traduz o programa escrito em uma linguagem de programação para um programa equivalente escrito em linguagem de máquina (programa-objeto).



Vantagens:

- Velocidade de execução
- Oculta o código fonte


Desvantagem: A cada alteração no programa fonte é necessário gerar novamente o programa-objeto

Tipos de Dados

Quando especificamos um algoritmo, detalhamos:

- ✔ os **DADOS** (números binários, isto é, sequências de 0s e 1s, armazenados na memória, correspondem à porção das informações a serem processadas) que serão processados e
- ✔ as **INSTRUÇÕES** (ou comandos, comandam o funcionamento da máquina e determinam como devem ser manipulados os dados) que vão operar sobre esses.

O objetivo é classificar os dados de acordo com o tipo de informação contida neles. A classificação apresentada não se aplica a nenhuma linguagem de programação específica.



Tipos de Dados

- ✓ **inteiro**: informações que não possuem componente decimal ou fracionário, podendo ser positivo ou negativo.
- ✓ **real**: informações que podem possuir componentes decimais ou fracionários, podem ser positivos ou negativos. A simples existência do ponto decimal diferencia um dado numérico do tipo inteiro de um do tipo real.
- ✓ **literal ou caracteres**: é constituído por uma sequência de caracteres contendo letras, dígitos e/ou símbolos especiais. São representados nos algoritmos pela coleção de caracteres, delimitada pelas aspas (“texto”) ou aspas simples para um caracter (‘p’).
- ✓ **lógico**: informação que podem assumir apenas dois possíveis valores: verdadeiro ou falso, sim/não, 1/0, true/false.

Tipos de Dados

	Tipos	Exemplo de utilização
numérico	inteiro	idade, ano, quantidade de filhos
	real (separador de casas decimais é o ponto)	salário, peso, altura
texto	literal (representa 1 caracter, aspas simples ou sequencia de caracteres, aspas duplas)	opção, primeira letra do nome, operação matemática, nome, cargo, endereço
lógico	logico (verdadeiro ou falso)	formado, solteiro

Tipos de Dados Primitivos – Java

Classificação	Tipo	Descrição
Lógico	boolean	Pode possuir valores true (verdadeiro) ou false (falso)
Inteiro	byte	Abrange de -128 a 127 (8 bits)
	short	Abrange de -32768 a 32767 (16 bits)
	int	Abrange de -2.147.483.648 a 2.147.483.647 (32 bits)
	long	Abrange de -2^{63} a $2^{63} - 1$ (64 bits)
Ponto flutuante	float	Abrange de $-3.4028E+38$ a $3.4028E+38$ (32 bit) com precisão simples
	double	Abrange de $-1.7976E+308$ a $1.7976E+308$ (32 bit) com precisão dupla
Caracter	char	Pode armazenar um caracter Unicode (16 bits) ou um inteiro entre 0 e 65535

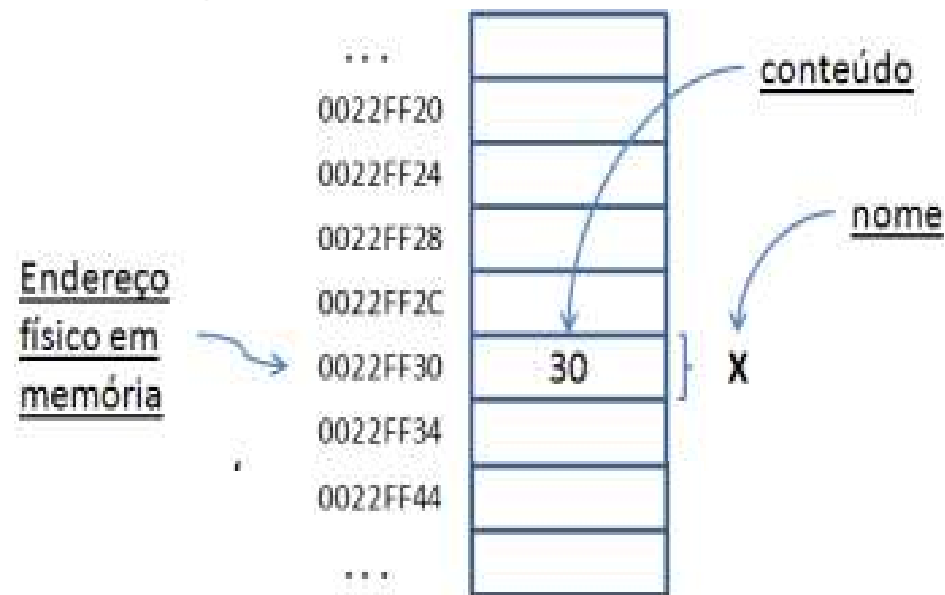
O double é a opção padrão para a representação de números em ponto flutuante, devido a baixa precisão do tipo float.

Tipos de Dados Primitivos (C#)

Tipo	Tamanho em Bits	Valores
bool	8	true ou false
char	16	\u0000' a '\uFFFF
byte	8	0 a 255
sbyte	8	-128 a 127
short	16	-32768 a 32767
ushort	16	0 a 65535
int	32	-2147483648 a 2147483647
uint	32	0 a 4294967296
long	64	-9223372036854775808 a 9223372036854775807
ulong	64	0 a 18446744073709551615
decimal	128	1 X 10e-28 a 7,9 x 10e 28
float	32	1,5 X 10e-45 a 3,4 x10e38
double	64	5,0x10e-324 a 1,7x10e308

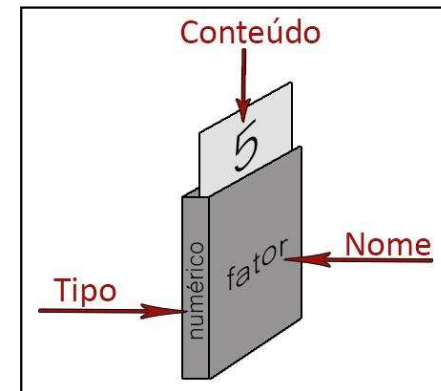
Variáveis

- ✔ Espaço de memória que pode receber um valor e sofrer alteração no decorrer do algoritmo/tempo.
- ✔ Toda variável tem um nome único que a identifica (**identificador**), um valor e o tipo correspondente à informação a ela atribuída.



Variáveis

- ✓ Nos algoritmos, cada variável corresponde a uma posição de memória.
- ✓ Embora uma variável possa assumir diferentes valores, ela só pode armazenar um valor a cada instante.
- ✓ Uma variável possui três atributos:
 - um nome (ou **identificador**),
 - um tipo de dado e
 - a informação por ela guardada.
- ✓ Cada linguagem de programação estabelece suas próprias regras de formação de nomes de variáveis.



Variáveis

O nome de uma variável deve ser representativo do seu conteúdo e possui as seguintes regras:

1. Não pode começar com números, apenas com letras
2. Não pode conter espaços em branco
3. Não pode conter caracteres especiais (#, ?, !, @, +, -, ...)
4. Não pode ser palavra reservada

Válidos	Inválidos
qtde_filhos	meu nome
idade	1tentativa
nota1	Real
Nome_Completo	ficha#2

Declaração de Variáveis

- ✔ Todas as variáveis utilizadas nos algoritmos devem ser definidas antes de serem utilizadas. Isto se faz necessário para permitir que o compilador reserve um espaço na memória para as mesmas.
- ✔ Para indicar o tipo de uma ou mais variáveis é feita a declaração de variáveis. A partir do momento da declaração das variáveis, é feita uma associação do nome escolhido, com a respectiva posição de memória.

Exemplo:

inteiro number1,number2

real arquivo

literal nome

lógico escolha

Constantes

- ✓ Valor fixo, numérico ou não, que deve permanecer inalterado no decorrer da execução do algoritmo.
- ✓ Em programação geralmente as constantes são declaradas com letras maiúsculas.
- ✓ Após sua declaração, fazemos a inicialização com o valor que será fixo em todo o nosso programa.
- ✓ As regras de criação do nome, seguem as mesma de variáveis.
- ✓ Podemos utilizar a palavra constante em pseudocódigo.
- ✓ A forma de criar uma constante em programação varia conforme a linguagem de programação.

Exemplo:

constante real $\text{PI} = 3.14$

constante inteiro $\text{VOLUME_MAX} = 100$

Inicialização de Variáveis

Existem várias maneiras de atribuir valores a variáveis:

- ✔ Dizendo no algoritmo qual o valor a variável deve assumir:

...

```
inicio
```

```
    real preco
```

```
    preco = 12.99
```

Inicialização de Variáveis

Existem várias maneiras de atribuir valores a variáveis:

- ✔ Definir que uma variável assuma o valor de uma outra variável:

...

`inicio`

`inteiro n1,n2`

`n1 = 10`

`n2 = n1`

Inicialização de Variáveis

Existem várias maneiras de atribuir valores a variáveis:

- ✔ Atribuir uma variável o resultado de uma expressão;

...

```
inicio
```

```
    real a,b,c
```

```
    a = 12.05
```

```
    b = 5.20
```

```
    c = a*b
```

- ✔ Usuário digitar o valor (comando de entrada, como veremos a seguir)

Comandos de Entrada e Saída (Input/Output)

Os algoritmos precisam ser “alimentados” com dados provenientes do meio externo para efetuarem as operações e cálculos e é necessário também mostrar os resultados.

Comando de entrada:

LEIA → tem como finalidade atribuir o dado a ser fornecido à variável identificada.

Exemplo: leia (variável)



Comandos de Entrada e Saída (Input/Output)

Comando de saída:

ESCREVA ➔ cuja finalidade é exibir uma mensagem, essa mensagem pode ser um texto ou o conteúdo de uma variável, ou ambos juntos

Exemplos:

escreva ("Mensagem")

escreva ("Mensagem" + variável)

escreva (variável)

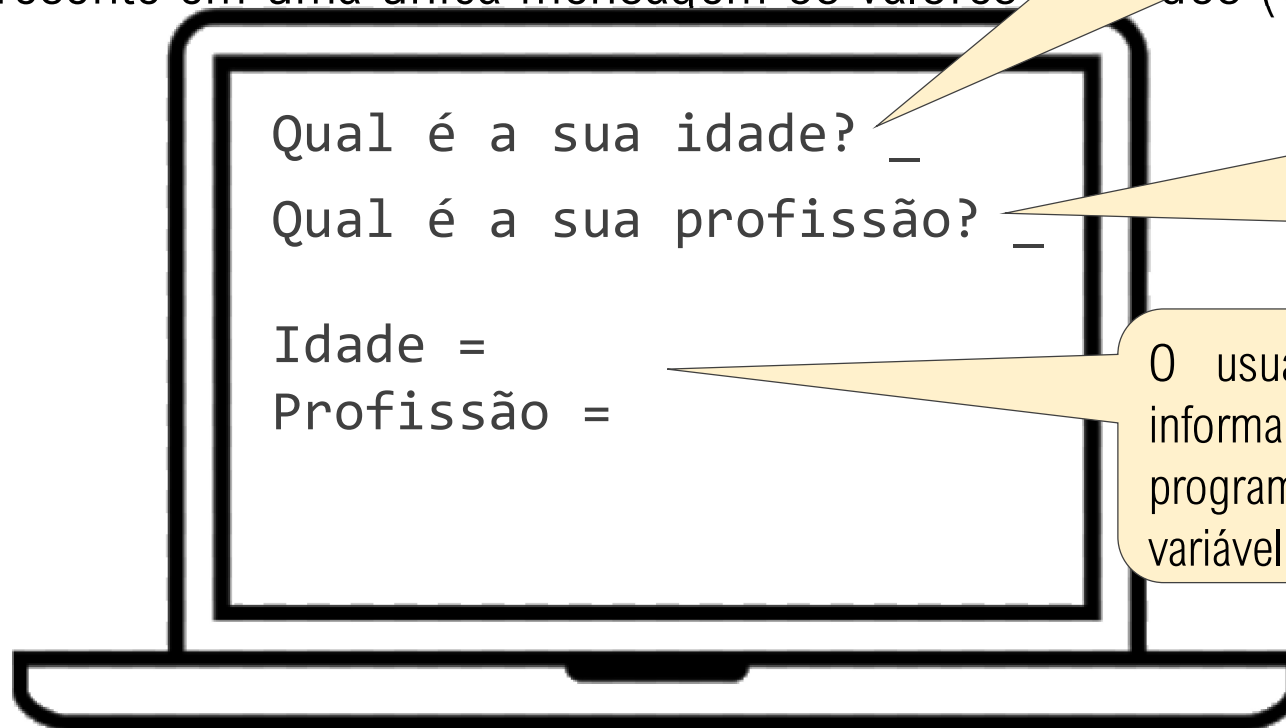
Exemplos de aplicação

1- Crie um algoritmo que solicita ao usuário a sua idade e armazena essa informação em uma variável, solicita também a profissão e armazena em outra variável. Após obter os dados, apresente em uma única mensagem os valores digitados (variáveis)

|

Exemplos de aplicação

1- Crie um algoritmo que solicita ao usuário a sua idade e o programa armazena em uma variável, solicita também a profissão e armazena em uma variável. Depois de obter os dados, apresente em uma única mensagem os valores digitados (variáveis).



Exemplos de aplicação

algoritmo Dados

início

inteiro idade

literal profissao

escreva (“Digite a sua idade”)

leia (idade)

escreva (“Digite a sua profissão”)

leia (profissao)

escreva (“Idade = “ + idade + “\nProfissão = “ +
profissao)

fim

RESOLUÇÃO EXEMPLO 1

Exemplos de aplicação

2- Crie um algoritmo que solicita ao usuário o modelo de um carro, a quantidade de quilômetros rodados e o seu valor. Após, mostre os dados do carro.

Exemplos de aplicação

RESOLUÇÃO EXEMPLO 2

algoritmo Carros

início

real valor, km

literal modelo

escreva (“Digite o modelo do carro”)

leia (modelo)

escreva (“Digite a quilometragem do carro”)

leia (km)

escreva (“Digite o valor do carro”)

leia (valor)

escreva (“Os dados do carro são: Modelo = “+ modelo+
 “Quilometragem = “+ km+ “Valor = “+ valor)

fim

Exemplos de aplicação

3- Crie um algoritmo que calcule a média aritmética de 4 números reais digitados pelo usuário e exiba o resultado.

Exemplos de aplicação

RESOLUÇÃO EXEMPLO 3

```
algoritmo media_quatro_valores
início
    real: n1, n2, n3, n4, media
    escreva ("Digite o 1º valor: ")
    leia (n1)
    escreva ("Digite o 2º valor: ")
    leia (n2)
    escreva ("Digite o 3º valor: ")
    leia (n3)
    escreva ("Digite o 4º valor: ")
    leia (n4)
    media ← (n1+n2+n3+n4)/4
    escreva ("A média dos valores é: " + media)
fim
```

Exemplos de aplicação

4- Crie um algoritmo que leia os valores dos lados de um retângulo e calcule/exiba o perímetro e a área do mesmo.

Exemplos de aplicação

algoritmo retangulo

RESOLUÇÃO EXEMPLO 4

início

real: ladoA, ladoB, perim, area

escreva ("Digite o valor de um lado (em cm): ")

leia (ladoA)

escreva ("Digite o valor de outro lado (em cm): ")

leia (ladoB)

perim \leftarrow 2*ladoA + 2*ladoB

escreva ("Perímetro: " + perim + " cm ")

area \leftarrow ladoA * ladoB

escreva ("Área do retângulo: " + area + " cm² ")

fim

Exemplos de aplicação

5- Elaborar um algoritmo que solicite os dados de altura (em metros) e peso (em Kg) de uma pessoa e calcule/visualize seu IMC (Índice de Massa Corporal).

Lembre que $IMC = \text{peso} / \text{altura}^2$