BCC 201 - Introdução à Programação

Portugol

Guillermo Cámara-Chávez
UFOP

Introdução I

Lógica

A **lógica** é usada no dia a dia das **pessoas** que trabalham com **computação** para **solucionar problemas** de forma **eficiente**.

Algoritmo

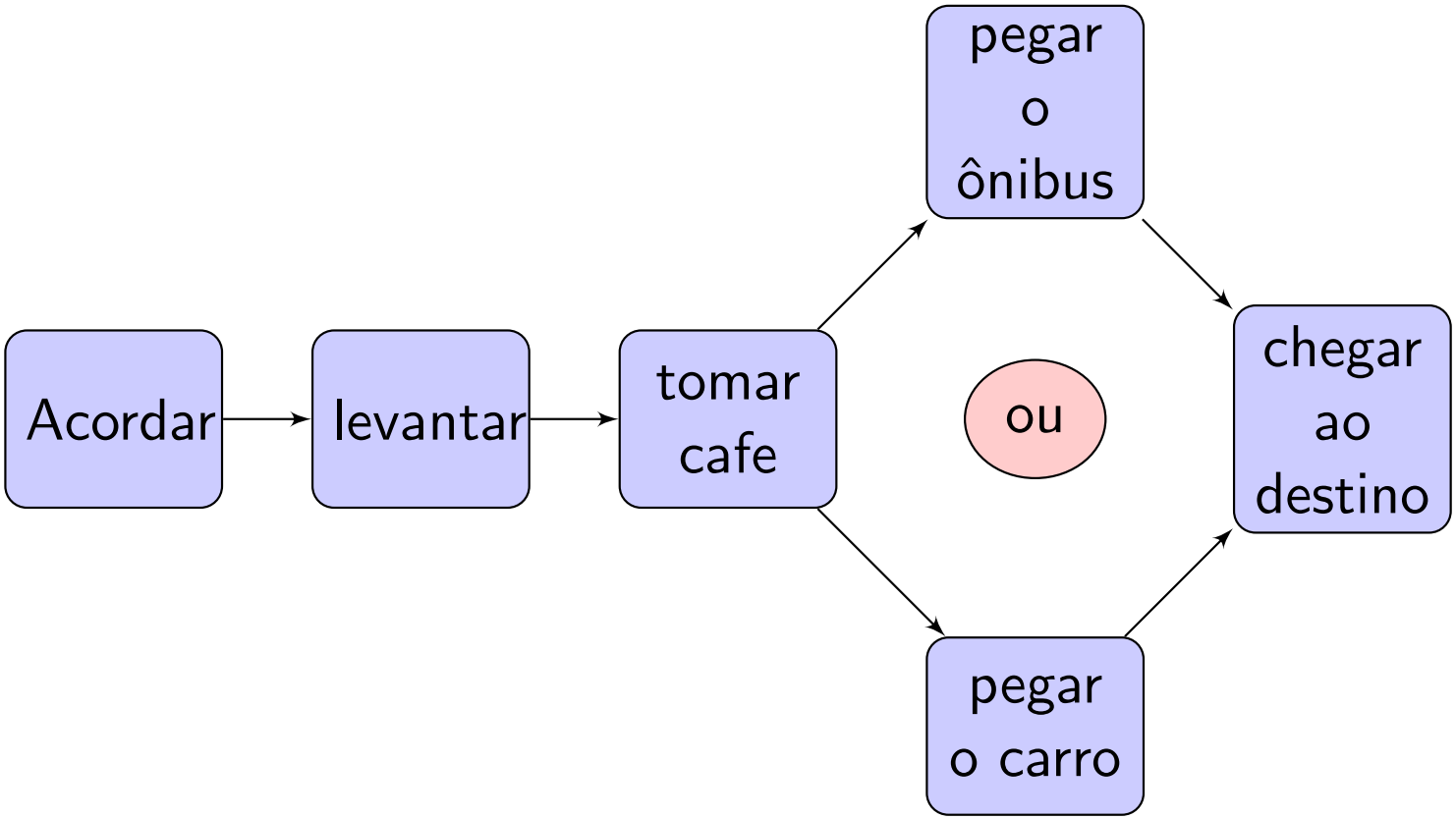
Um algoritmo representa de forma estruturada, uma **seqüência de ações**, que levam a um **resultado esperado**.

Resumindo:

- algoritmo: exercício de raciocínio (**definir o problema**);
- técnicas de programação: exercício da implementação

Introdução II

- Exemplo: Seqüência de ações para chegar ao trabalho/universidade



Introdução III

- Para cada ação acontecer, é necessário que a **ação anterior** tenha sido executada
- Cada **ação** pode **conter outros eventos** associados (outros algoritmos)

Portugol I

Portugol

É uma **pseudolinguaguem** que permite ao programador **pensar no problema** em si e não no equipamento que irá executar o algoritmo.

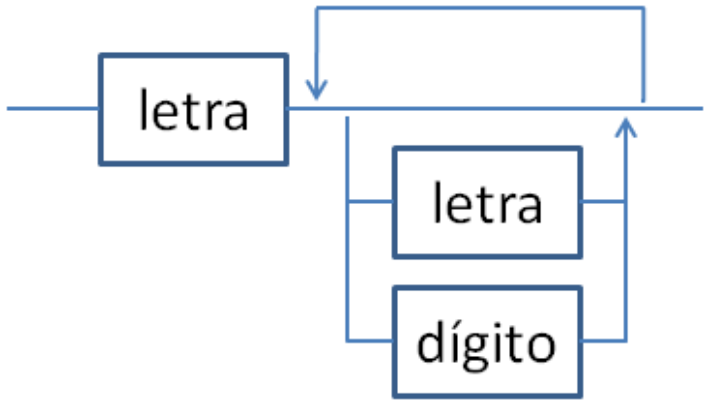
Portugol II

Estrutura de um algoritmo

```
Inicio
    <declarações de variáveis>
    <comandos>
Fim
```

Portugol III

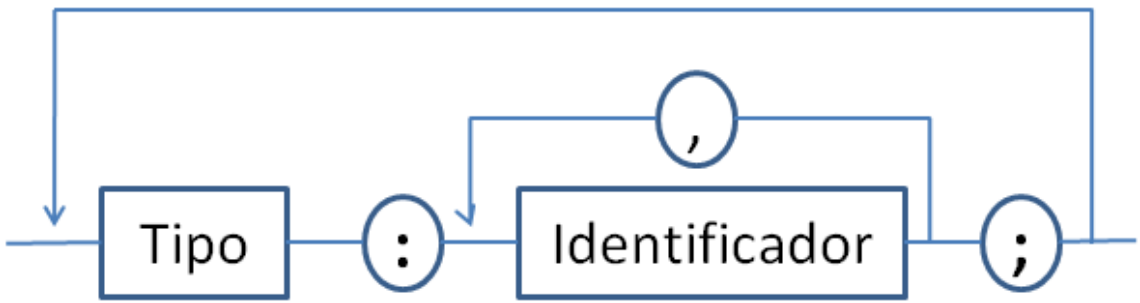
- **Identificadores**: elemento básico da linguagem, a sua sintaxe é definida por



  - Exemplos: area, nota1, i, N1, ...

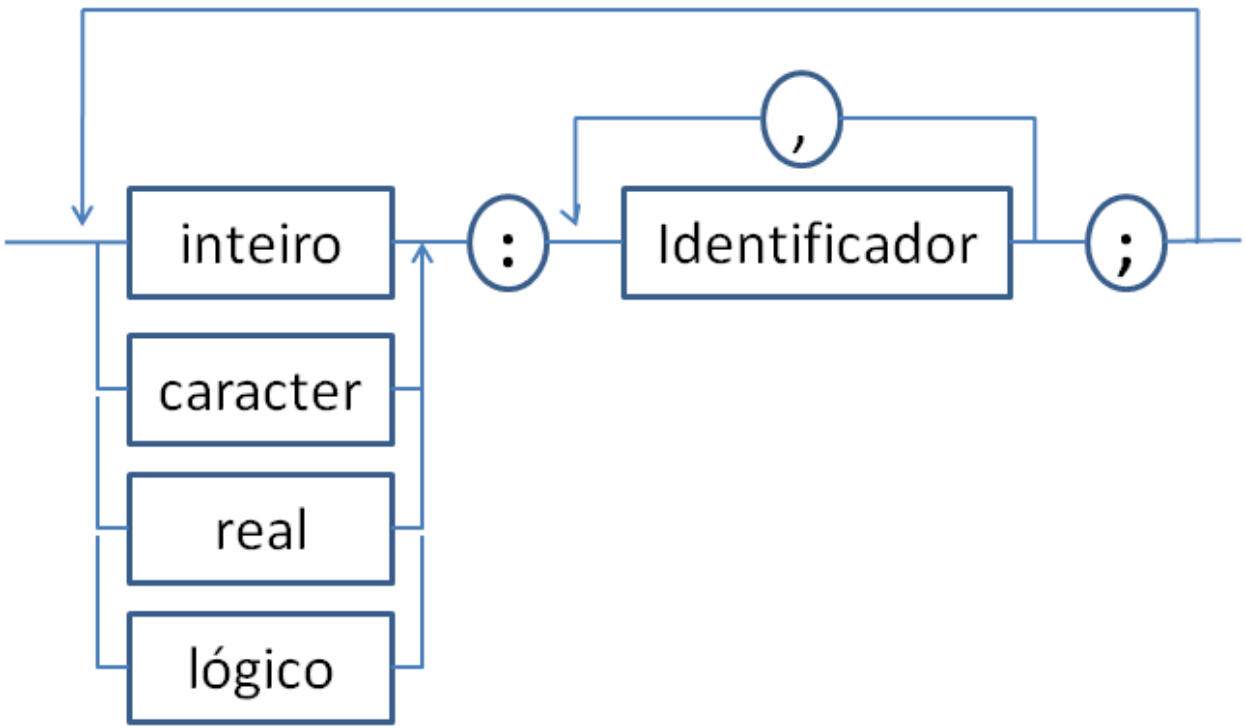
Portugol IV

- Declaração de variáveis
  - É um local que armazena um tipo específico de conteúdo
  - Contém um valor que se modifica durante a execução de um programa.
  - Possue um identificador (nome), que pode ser representando:



Portugol V

- No Portugol existem quatro tipos básicos de dados utilizados: INTEIRO, REAL, CARACTER e LÓGICO



Portugol VI

- Inteiro: qualquer número inteiro (negativo, nulo ou postivo)
  Exemplo: 100, 0, 1, 2, 1250
- Real: qualquer número real (negativo, nulo ou postivo)
  Exemplo: -10, -1.5, 11.2, 0, 1, 2, 50
- Caracter: caracteres alfanuméricos
  Exemplo: casa, Win31, 123
- Lógico: valor lógico verdadeiro ou falso
  Exemplo: $x > y$?

Portugol VII

- Exemplo:

```
inteiro: idade;
real: nota1, nota2, media;
caracter: nome_aluno;
logico: maior;
```

Portugol VIII

- É importante não esquecer
  1. Não é possível definir variáveis de diferentes tipos com o mesmo identificador.
     Exemplo: *real A; inteiro A;* causaria erro na programação.
  2. Tomar cuidado em relação à sintaxe da linguagem. Não é possível ter identificador como: *caracter ?nome; real valor*; inteiro 1x;*
  3. Letras maiúsculas e minúsculas são tratadas de forma diferente.
     Exemplo: *Media* é diferente de *media*, como também de *MEDIA*;

Portugol IX

- **Constantes**
  - Uma constante é um valor fixo que não se modifica ao longo do tempo
  - Em algoritmo representaremos constantes pelo tipo **const** ou `#define` (eventualmente alguns elementos da linguagem C poder ser escritos no algoritmo)

```
const M 10;
```

Portugol X

- **Comandos básicos**:
  - O comando de **atribuição** é utilizado para atribuir um valor a uma variável.
  - Para isso usamos o símbolo "$\leftarrow$"

Portugol XI

- A notação usada para expressões é básicamente uma forma **linear** comumente usada na matemática, que pode conter operadores:
  - Aritméticos: $+$, $-$, $*$, $/$, raiz(), $\wedge$, sen(), cos(), mod, div, ...
  - Lógicos: e, ou, não
  - Relacionais: $=$, $\neq$, $>$, $\geq$ (ou $>=$), $<$, $\leq$ (ou $<=$)

Portugol XII

Exemplos:

- Atribuição de um valor constante

```
inteiro valor;
valor <- 10;
```

- Atribuição entre variáveis

```
inteiro valor;
inteiro x;
x <- 10;
valor <- x;
```

- Resultado de expressões

```
inteiro valor;
inteiro x, y;
x <- 10;
y <- 5;
valor <- x + y * 2;
```

Exercício 1 I

Desenvolva um algoritmo em portugol para somar dois valores inteiros (Ex. 10+5)

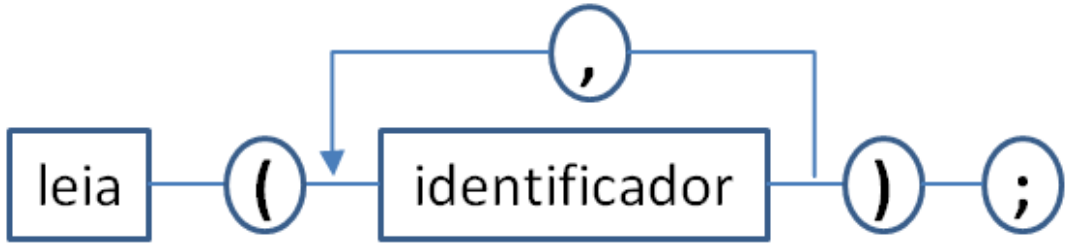
```
Inicio
    inteiro: x, y, z; //declara três variáveis inteiras
    x <- 10;  // atribui 10 para x
    y <- 5;  // atribui 5 para y
    z <- x + y;  // soma x e y, o resultado é atribuido a z
Fim
```

Entrada e Saída de dados I

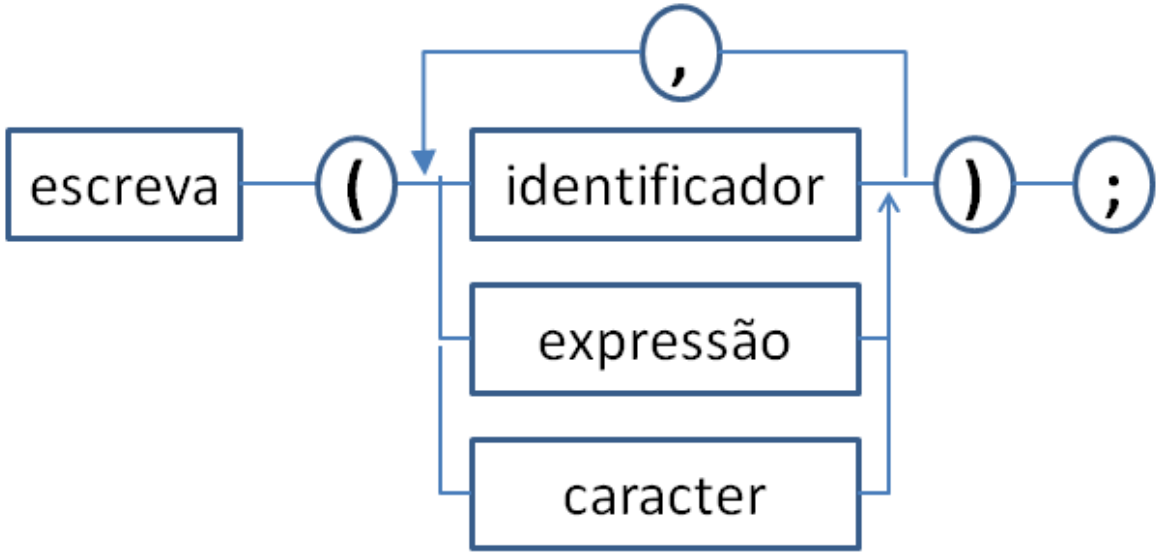
- Um programa pode receber um dado informado através de um comando de **leitura**
- Também pode ser necessário conhecer o resultado de uma determinada operação, nesse caso usaremos um comando de **escrita**

Entrada e Saída de dados II

O comando de entrada é **leia**



O comando de saída é **escreva**



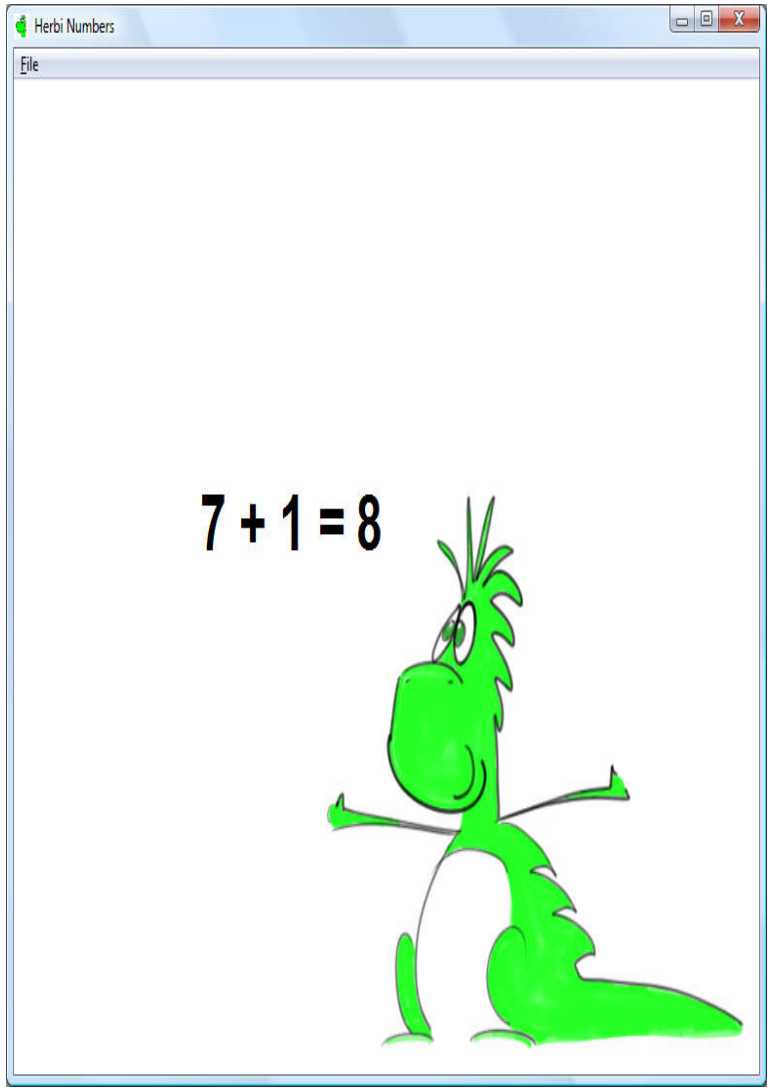
Exercício 2 I

De forma genérica, a construção de um algoritmo se resume às seguintes etapas:

1. entendimento do problema;
2. elaboração da solução algorítmica; e
3. codificação da solução no Português Estruturado

Exercício 2 II

Inserir dois números inteiros e encontrar a soma



Exercício 2 III

Etapa 1
Simples, sabemos que vamos a calcular a soma de dois números
Etapa 2
Os dados necessários serão os dois valores, que colocaremos em duas variáveis A e B, de tipo inteiro, e uma terceira variável, que chamaremos de *soma*, que armazenará a soma das duas variáveis.
Etapa 3
A obtenção dos dados neste programa é simples e direta. Basta pedir ao usuário que digite os valores.
Etapa 4
O processamento aqui é o cálculo da soma, usando o método citado na **etapa 1**. O resultado será armazenado na variável *soma*.
Etapa 5
Exibir o conteúdo da variável *soma*

Exercício 2 IV

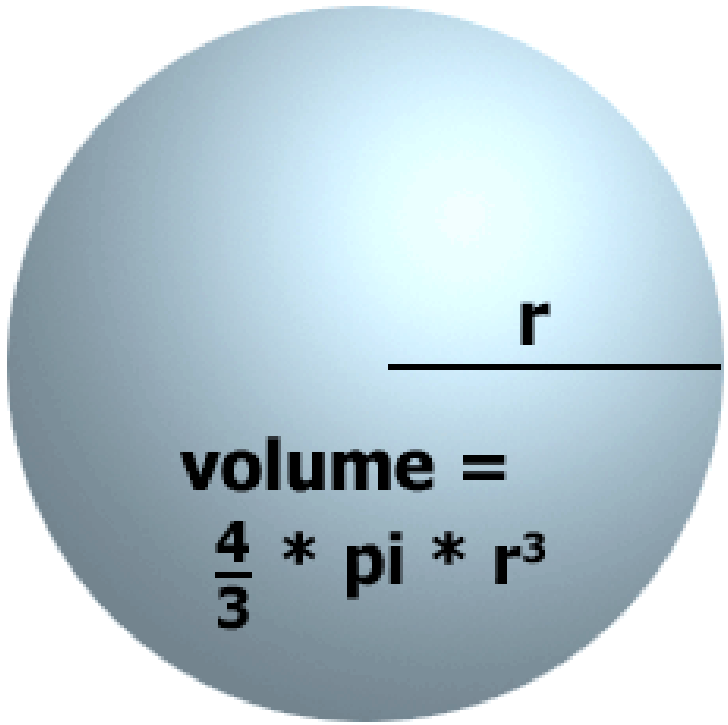
```
Inicio
    inteiro: x, y, soma;
    escreva("Inserir dois números: ");
    leia(x, y);
    soma <- x + y;
    escreva("A soma entre", x, " e ", y, " é ", soma);
Fim
```

Regras para escrever algoritmos I

- Incluir comentários
- Usar nomes significativos para as variáveis que possam identificar o conteúdo
- Identar os comandos **facilita a legibilidade** do algoritmo e **reduz** a possibilidade de **erros**

Exercício 3 I

Desenvolva um programa que calcule o volume de uma esfera de raio R, fornecido pelo usuário. ($V = 4/3\pi R^3$)



Exercício 3 II

```
Inicio
    const pi 3.14159;
    real: R, volume;
    escreva("Inserir raio da esfera");
    leia(R);
    volume <- 4/3 * pi * (R^3);
    escreva("O volume da esfera: ", volume);
Fim
```

Exercício 3 III

Desenvolva um algoritmo para encontrar a média entre 4 valores fornecidos pelo usuário

Exercício 3 IV

```
Inicio
    real: nota1, nota2, nota3, nota4, media;
    escreva("Inserir quatro notas");
    leia(nota1, nota2, nota3, nota4);
    media <- (nota1 + nota2 + nota3 + nota4) / 4;
    escreva("Valor da media: ", media);
Fim
```

Comandos de Controle I

- Permite
  - alterar a direção tomada por um programa, ou (desvio)
  - fazer com que partes específicas de um algoritmo seja executada mais de uma vez (*loop*)

Comandos de Controle II

- **Desvio condicional**: muitas vezes será necessário desviar a execução do programa segundo uma condição.
  - Por exemplo, ir a universidade de carro ou de ônibus?
  - Para se testar condições é necessário utilizar operadores lógicos e relacionais

Comandos de Controle III

- **Desvio condicional simples**

```
se (condição) então
    lista de comandos
fim_se
```

Inserir um número real, se ele for positivo imprimir o número

Comandos de Controle IV

```
Inicio
    inteiro: A;
    escreva("Inserir valor ");
    leia(A);
    se A > 0 então
        escreva(A);
    fim_se
Fim
```

Comandos de Controle V

- **Desvio condicional composto**
  - As condições, verdadeiras ou falsas, geram ações através de um único comando de desvio condicional

```
se (condição) entao
     lista de comandos
senão
     lista de comandos
fim_se
```

Inserir dois valores numéricos e encontrar o menor deles e a média

Comandos de Controle VI

```
Inicio
    real: num1, num2, media;
    escreva("Inserir dois valores");
    leia(num1, num2);
    se num1 < num2
        escreva("o menor é ", num1);
    senão
        escreva("o menor é ", num2);
    fim_se
    media = (num1 + num2) / 2;
    escreva("A media é: ", media);
Fim
```

FIM