Fundamentos de Lógica de Programação

Aula 5

Prof Tanilson Dias dos Santos

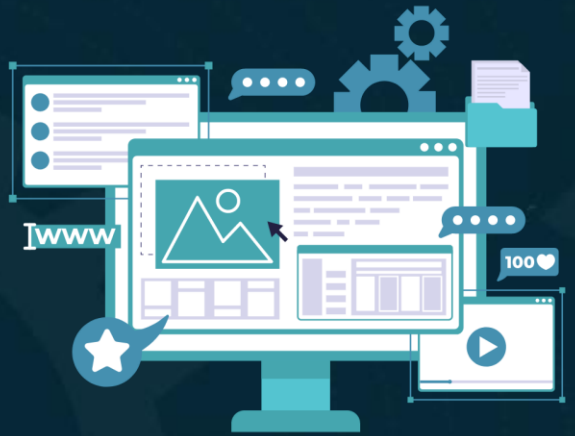
Universidade Aberta do Brasil – UAB
Universidade Federal do Tocantins - UFT

Relembrando Últimas Aulas

- Conceitos Básicos de Lógica de Programação;
- Fluxogramas e Tabela Verdade (conectivos AND, OR, NOT);
- Programação em Portugol:
  - Entrada/Saída de Dados;
  - Laços de Repetição;
  - Comando de Seleção Múltipla.

Roteiro da Aula 5

- **Feedback das Aulas Anteriores:**
  - **Fórum**: Ótimas respostas para a calculadora com **selecione... caso**
  - **Questionário**: Boas Respostas mas pouca Participação!
  - **Ritmo da Disciplina**: Algum problema até aqui?
- **Manipulação de Cadeias e Vetores;**
- **Mais conceitos de Portugol.**



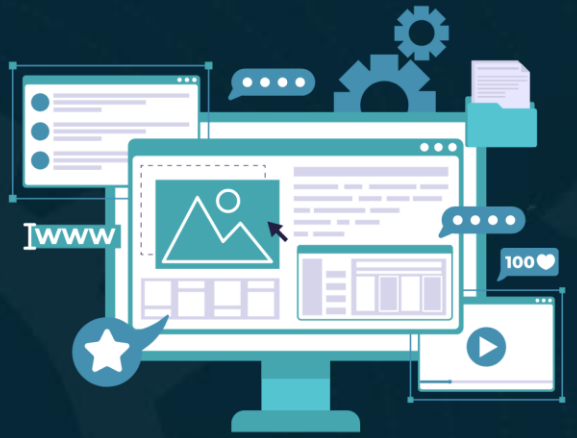
Programação em Portugol

Curiosidades Sobre Portugol

- O Portugol foi desenvolvido por Antônio Carlos Nicolodi (IFSC) e Cláudio Morgado de Souza (FURB);
- A primeira variação (dialeto) de Portugol foi criada pelos professores Antonio Carlos Nicolodi (Brasil) e António Manso (Portugal) em 1986;
- Possui mais de 1 milhão de downloads no baixaki;
- Projeto com 2.650 acessos no Source Forge em mais de 25 países no mundo;

Curiosidades Sobre Portugol

- O Portugol possui como principais dialetos:
- Portugol VisuAlg (Antônio Carlos Nicolodi e Cláudio Morgado de Souza);
- Instituto Politécnico de Tomar – IPT (António Manso);
- G-Portugol
- Portugol Viana
- P&G editor



Recaptulando . . .

Entrada/Saída de Dados

- **leia** - Lê um valor do usuário via teclado;
- **escreva** – Escreve um valor na tela;
- **limpa( )** – limpa o terminal de visualização.

Laços de Repetição

- **enquanto** – repete um bloco de código enquanto uma condição é satisfeita (repetição com teste no início);

- **faça ... enquanto** – executa um bloco de código e testa ao final, se a condição final for satisfeita, repete a execução do código (repetição com teste no final);

- **para (arg1; arg2; arg3)** – possui 3 argumentos/parâmetros: **arg1** é o inicializador, executado apenas 1 vez; **arg2** é a condição de execução do loop testada a cada iteração; **arg3** é o incremento, também executado a cada iteração (repetição contada).

Laços de Repetição - Forma Geral

- **enquanto – repete um bloco de código enquanto uma condição é satisfeita (repetição com teste no início);**

enquanto (condição **) { //faça**
comandos;
} //fim enquanto

Laços de Repetição - Forma Geral

- **faça ... enquanto – executa um bloco de código e testa ao final, se a condição final for satisfeita, repete a execução do código (repetição com teste no final);**

```
faça { //faça
        comandos;
} enquanto ( condição )
```

Laços de Repetição - Forma Geral

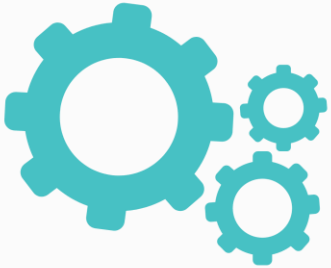
- **para (arg1; arg2; arg3) – possui 3 argumentos/parâmetros: arg1 é o inicializador, executado apenas 1 vez; arg2 é a condição de execução do loop testada a cada iteração; arg3 é o incremento, também executado a cada iteração (repetição contada).**

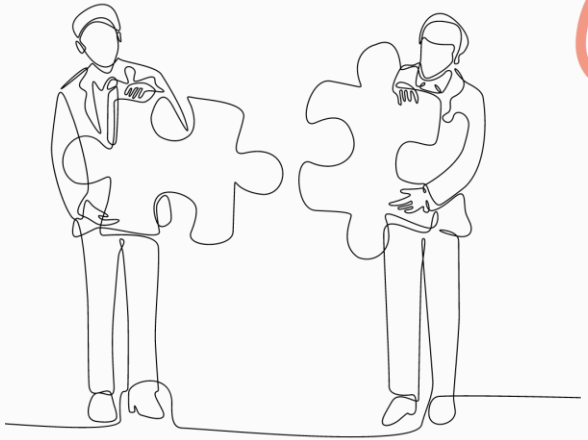
Para (inicializa **; (** condição **);** incremento**){**
comandos;
}//fim para

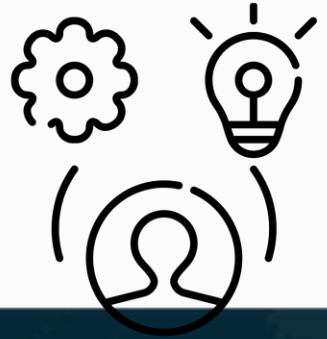
Um problema muito comum na programação!

Problema Aula Anterior

- **Leia 2 valores inteiros do usuário e armazene nas variáveis var1 e var2. Em seguida troque os valores de forma que o valor da variável 1 fique armazenado na variável 2, e vice-versa.**





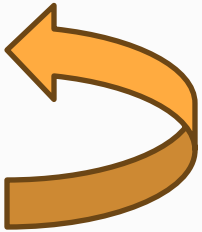


Problema Aula Anterior

- Leia 2 valores inteiros do usuário e armazene nas variáveis var1 e var2. Em seguida **troque os valores** de forma que o valor da variável 1 fique armazenado na variável 2, e vice-versa.

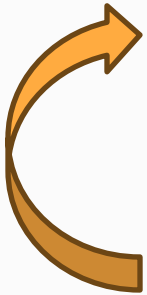
1) **Solução Ingênua**:

```
var1 = var2
var2 = var1
```



Não Funciona!

2) **Solução Correta**:



```
aux = var1
var1 = var2
var2 = aux
```

Funciona!

Outro problema comum na programação!

LICENCIATURA EM COMPUTAÇÃO

Escopo de Existência

- **Variáveis existem dentro do escopo em que foram declaradas;**
  - Não podem ser "enxergadas" fora deste escopo.
- **O que é escopo da variável?**
  - É o nome dado à propriedade que determina onde cada variável é visível ou não dentro do meu código.
- **O escopo pode ser local ou global.**

Escopo de Existência

- **Variáveis Globais -** são aquelas declaradas no início de um algoritmo. São visíveis, ou seja, podem ser acessadas em todo corpo do algoritmo.

- **Variáveis Locais -** são aquelas declaradas dentro de um bloco de código ou de um subalgoritmo. São visíveis, ou seja, podem ser acessadas somente dentro do bloco de código onde foram declaradas.

Vantagens vs Desvantagens

- **Vantagens**

1. **Variável local**

a. **O principal benefício de uma variável local é que não há alteração acidental dos dados. A variável é declarada dentro de um bloco e esse bloco de código usa a variável e evita efeitos colaterais indesejáveis.**

b. **A variável local consome memória por um período limitado do período, somente quando o bloco que contém a variável é executado.**

Vantagens vs Desvantagens

- **Vantagens**

1. **Variável global**

a. **Variáveis globais são muito úteis quando você está lidando com várias funções no programa que manipulam os mesmos dados.**
b. **As mudanças que precisavam ser aplicadas em todo o programa seriam mais fáceis através da implementação de uma variável global.**
c. **Podemos acessar de qualquer lugar ou através de qualquer função aleatória do programa.**

Vantagens vs Desvantagens

- **Desvantagens**

1. **Variável local**

a. **O escopo da variável local é restrito;**
b. **Proíbe o compartilhamento de dados;**
c. **Não retém os dados entre as chamadas, porque variáveis locais são geradas e removidas a cada entrada e saída do bloco.**

Vantagens vs Desvantagens

- **Desvantagens**

1. **Variável global**

- **O uso de um grande número de variáveis globais pode resultar na geração de erros de programa;**
- **Ocorrência acidental de alterações devido às variáveis globais disseminadas ao longo do programa;**
- **Refatoração de código se torna extensa.**

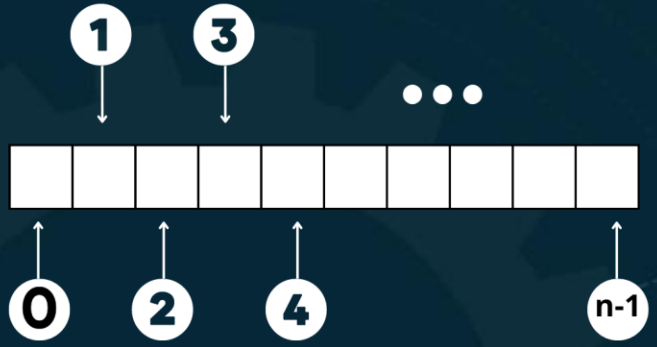
Exemplo 1

- Escrever um programa em Portugol que declare e inicialize uma variável global fora da funçao inicio, chamemos ela de **varGlobal**. Em seguida, declare uma variável local dentro da função inicio, chame ela de **varLocal1**. Por último, declare uma função teste que declara uma variável local dentro dela, digamos **varLocal2**, e retorna o varLocal2 operada com varGlobal e varLocal1. O que acontece?

*obs. Todas as variáveis envolvidas são do tipo real.

Resumo Variáveis Local e Global

VARIÁVEL GLOBAL	VARIÁVEL LOCAL
Visível em todo programa (**escopo amplo**)	Visível somente no escopo onde foram declaradas (**escopo restrito**)
Declaradas no início do programa ou em algum local especial	Declaradas dentro das funções, métodos ou blocos de código.
Existem durante todo tempo de execução do programa	Existe somente enquanto a função ou bloco de código na qual declarada estiver sendo executado
Permite compartilhamento de dados entre funções	Dados acessados somente dentro do bloco delimitado.



Caracteres e Cadeias

Caracteres e Cadeias

- **Caracter** - qualquer símbolo alfanuméricos da tabela ASCII;

Obs. 1: O caractere '5' é diferente do valor inteiro 5.

- **Cadeia** (“String”) - uma variável deste tipo poderá armazenar uma cadeia de caracteres de qualquer tamanho.

Obs. 2: Os textos deverão ser representados sempre entre apóstrofes/aspas para que não se confundam com os valores numéricos.
Obs. 3: Toda cadeia sempre tem a posição do primeiro caractere indexada com a índice zero.

Exemplo 2 - Cadeias

Obs. 4: o operador + também pode ser utilizado com operador de concatenação de cadeias;

- **Escrever um programa em Portugol que leia o nome e sobrenome do usuário e concatene em uma nova variável.**

Biblioteca Texto

- **Para outras funções de manipulação de texto deve-se utilizar a biblioteca Texto:**

  a) Tamanho do texto;

  b) Ler caracteres;

  c) Pesquisar no texto;

  d) Caixa Alta/Caixa Baixa;

  e) Preencher Texto, etc.

Exemplo 3

- Escrever um programa em Portugol que leia o nome do usuário e deixe as vogais em **CAIXA ALTA**.

  *obs. Usar a biblioteca **Texto**.

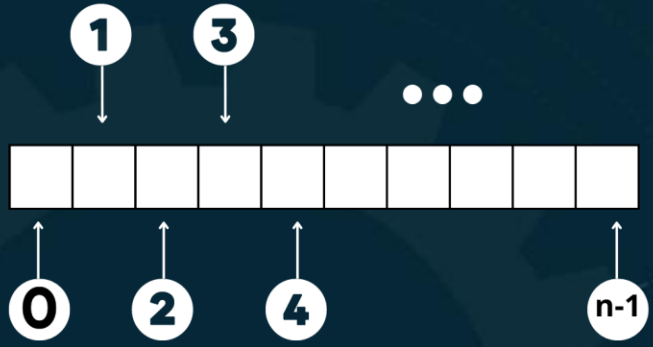
Desafio 4

- **Escrever um programa em Portugol que leia o nome do usuário escreva letra por letra, uma letra por linha.**

  ***obs. Usar a biblioteca Texto.**

Desafio 5

- **[*Linguagem do Cebolinha*]: Escrever um programa em Portugol que leia uma frase digitada pelo usuário e escreva como o Cebolinha, da turma da Mônica, diria essa frase. *Dica: é suficiente trocar 'r' por 'l'; e 'R' por 'L'.**
    - **Exemplo 1**: Usuário digita: "O problema é seu", **saída**: "O ploblema é seu"
    - **Exemplo 2**: Usuário digita: "Ruiva, a arara, riu e riu!", **saída**: "Luiva, a alala, liu e liu!"



Vetores

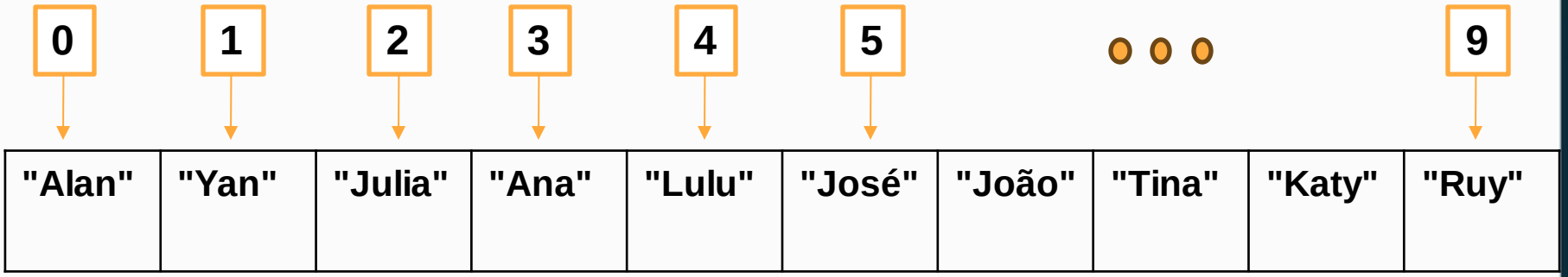
Vetores

- Suponha a seguinte situação, ao invés de ler o nome de 1 usuário em uma aplicação, você tenha que ler o nome de 10 usuários;
- Além dos 10 nomes, você também teria que ler 10 idades de cada um desses usuários;
- Depois de ler os nomes e as idades, você teria que dizer o nome do usuário mais velho;
- A ideia ingênua para resolver esse problema seria criar 10 variáveis nome e também criar 10 variáveis par armazenar as idades.

Vetores

- **A estrutura de dados do tipo vetor vem nos ajudar a resolver esse tipo de problema.**
- **Mas afinal, o que é um vetor?**
- **Resposta:** Um vetor pode ser visto como uma variável que possui diversas posições, e com isso armazena diversos valores, porém todos do mesmo tipo.
- **Um vetor sempre tem sua posição inicial indexada por zero;**
- **Um vetor do tipo cadeia que vai armazenar os nomes de 10 pessoas pode ser declarado da seguinte forma: cadeia pessoas[10]**

Vetores



- Vetor **pessoas**
- Tipo **cadeia**
- Tamanho de **10 posições**

Desafio 6

- Escreva um programa em Portugol que leia o nome e a idade de 10 pessoas fornecida pelo usuário.
- Depois de ler os nomes e as idades, seu programa deve dizer qual é o nome do usuário mais velho e qual a sua idade.

Obs. Essa implementação possui duas variações simples:

1 - no momento da inserção dos dados (em tempo de execução) já dá pra verificar quem é o usuário mais velho;

2 - depois dos dados inseridos, efetuar uma busca no vetor de idades buscando a idade do mais velho.

Desafio 7

- [*Algoritmo de Ordenação*]: Escrever um programa em Portugol que leia um número do usuário, digamos **quant**, esse número é a quantidade de elementos que ele vai inserir na sequência. Todos elementos são números inteiros positivos. Após ler os elementos e inserir em um vetor, digamos **elem**, ordenar os elementos no vetor e apresentar o vetor ordenado.

Dica: *usar bubble sort*.

Tarefas Semanais

- Refazer Exercícios da Aula;
- Responder Questionário Avaliativo;
  (**IMPORTANTE!** Estudem antes de tentar resolver o questionário!)
- Responder Fórum;
- Monitoria dia 14/06 às 19h;
- Tentar fazer os exercícios de Programação Sugeridos no material "exercicios-Aula5.pdf"

Conclusão e Próxima Aula

- **Aula de Hoje:**
  - **Exercícios sobre Escopo Variáveis;**
  - **Exercícios sobre Cadeias e Vetores;**
  - **Prática de Programação.**
- **Próxima Aula:**
  - **Prática de Programação com vetores e matrizes.**

Dúvidas até aqui? Muitas, Provavelmente

- Procurem o tutor!
- Mandem mensagem no fórum de dúvidas!
- Façam as atividades!
- Programação se aprende programando, então programem!

Boa Sorte!

Fundamentos de Lógica de Programação

Aula 5

Prof Tanilson Dias dos Santos

Universidade Aberta do Brasil – UAB
Universidade Federal do Tocantins - UFT