FUNDAMENTOS EM LÓGICA DE PROGRAMAÇÃO

Atividade: Começando a pensar logicamente e a desenvolver softwares utilizando o Portugol Studio

Tema: Simplificando o código e repetindo estrutura – MÉTODOS/FUNÇÕES

INDICADORES ASSOCIADOS

2 - Analisa e avalia o funcionamento de computadores e periféricos em ambientes computacionais.
3 - Codifica programas computacionais utilizando lógica de programação e respeitando boas práticas de programação.
5 - Desenvolver capacidades linguísticas de modo a saber usar adequadamente a linguagem oral e escrita em diferentes situações e contextos.
8 - Utilizar estruturas de dados definindo-as e aplicando-as adequadamente nos programas.

SIMPLIFICANDO O CÓDIGO E REPETINDO ESTRUTURA – *MÉTODO/FUNÇÃO*

- Existe algum jeito de fazer os programas de modo mais simplificado?
- Se eu repito uma determinada ação, posso repeti-la sem o uso de uma forma recursiva, ou usar uma repetição?
- A resposta é: SIM!
- Podemos usar algo conhecido como MÉTODOS/FUNÇÕES para que um determinado trecho seja executado diversas vezes;
- E, aliás, escolher até o momento em que ele será executado;
- “Métodos (chamados de funções ou procedimentos em algumas linguagens) ajudam um programa separando suas tarefas em unidades autocontidas” (DEITEL; DEITEL, 2010, p.156):
    - Ou seja, são pequenas rotinas criadas para separar e tornar independente um determinado trecho de um código-fonte.
- Entendeu? Possivelmente nada ou muito pouco;
- Vamos para uma prática e daí, esperamos, que ficará tudo mais evidente;

- Como exemplo, construiremos um programa que fará a soma de dois números:
    - Lembre-se que agora usaremos MÉTODOS/FUNÇÕES.
- É a partir desse exemplo, que faremos as explicações de todos os conceitos necessários;
- Por ser um conteúdo mais amplo e complexo, a qualquer dúvida que você tenha pode nos chamar;
- Inicialmente faremos um código-fonte mais simples, no modelo que você já conhece;
- Assim, por momento, faremos a soma de dois valores de um modo mais simplificado;
- Copie o código-fonte abaixo e teste, pois é a partir dele que faremos nossas modificações:

Figura 1 – Código-fonte simplificado

```
programa
{
    funcao inicio()
    {
        inteiro valor1, valor2, resultado

        escreva("Este programa faz a soma de dois números!!!")

        escreva("\nInforme o primeiro valor:")
        leia(valor1)

        escreva("\nInforme o segundo valor: ")
        leia(valor2)

        resultado = valor1 + valor2

        limpa()

        escreva("\nO resultado desta soma é: " + resultado)

    }
}
```

Fonte: Autores, 2023.

- É um código-fonte bastante simples, certo?
- Então como podemos tornar isso mais ágil e mais dinâmico?
- Ao longo das alterações você verá como faremos isso;
- Por enquanto, vamos ao primeiro passo;
- Alteraremos nosso *MÉTODO/FUNÇÃO* principal, o inicio;

- Ele receberá uma formatação diferenciada, a fim de que utilizemos um *MÉTODO/FUNÇÃO* chamado solicitaNro;
- Mas daí você pergunta: o que esse *MÉTODO/FUNÇÃO* fará?
- Lembra que nosso objetivo é somar dois valores?
- Pois então, nossa ideia é simplificar o modo com que solicitamos o valor de cada um dos números ao usuário;
- Então, faremos a criação de um *MÉTODO/FUNÇÃO* chamado solicitaNro*:*

Figura 2 - Preparando o código-fonte para usar *MÉTODO/FUNÇÃO*

```
funcao inicio()
{
    inteiro valor1, valor2, resultado

    valor1 = solicitaNro("1")
    valor2 = solicitaNro("2")

    resultado = valor1 + valor2

    limpa()

    escreva("\nO resultado desta soma é: " + resultado)

}
```

Fonte: Autores, 2023.

- Primeiramente, vamos explicar o que fizemos nesse código-fonte;
- Substituímos as funções *ESCREVA* e *LEIA* por uma chamada de *MÉTODO/FUNÇÃO* para as variáveis *valor1* e *valor2*;
- Ou seja, o *MÉTODO/FUNÇÃO* solicitaNro fará a solicitação dos valores necessários e encaminhará o valor indicado para essas variáveis;
- Não se prenda a isso ainda, mas perceba que temos *"1"* e *"2"*;
- Posteriormente, isso fará uma composição numa frase que queremos mostrar ao usuário;
- Ficará claro mais a frente;
- Porém, temos um ponto de atenção;
- Repare que temos alguns itens sublinhados em vermelho;
- Por que isso acontece?
- E aí respondemos fazendo uma nova pergunta:
    - Chamamos o *MÉTODO/FUNÇÃO* solicitaNro, mas ele existe?
    - Criamos esse *MÉTODO/FUNÇÃO* em nosso código-fonte?

- Precisamos criar um *MÉTODO/FUNÇÃO* chamado solicitaNro para que nosso código-fonte seja corrigido;
- Esse *MÉTODO/FUNÇÃO* será responsável por solicitar ao usuário um valor;
- Com isso, adicionaremos em nosso código-fonte o trecho abaixo:
    - Importante, você não deve substituir o código, mas adicionar abaixo do *MÉTODO/FUNÇÃO* inicio:

Figura 3 - Criação do *MÉTODO/FUNÇÃO* solicitaNro

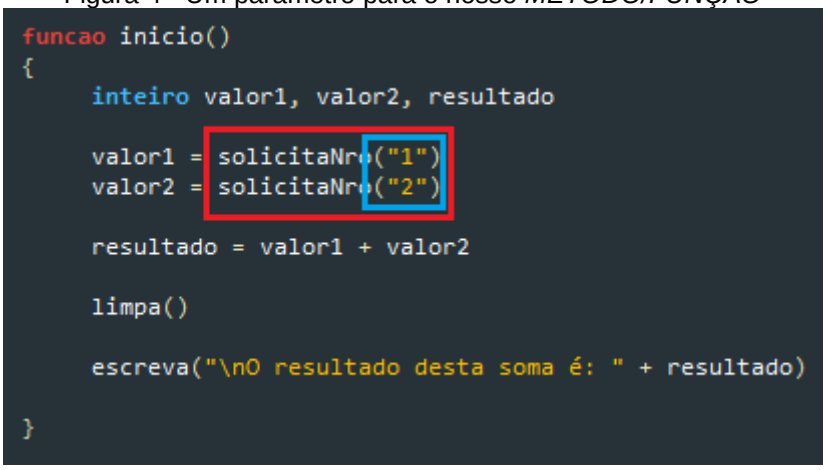
```
funcao inteiro solicitaNro(cadeia ordem)
{
    inteiro valor
    escreva("\nInforme o " + ordem + "º valor:")
    leia(valor)

    retorne valor
}
```

Fonte: Autores, 2023.

- Vamos entender cada ponto?
- Na primeira linha:
    - Temos a declaração da função, chamada de solicitaNro;
    - Essa função é declarada como *inteiro*, ou seja, ele devolverá para quem o chamou um resultado com esse tipo:
        - Essa informação é tida como o tipo de dados de retorno.
    - Por fim, entre os parênteses há a variável *ordem* do tipo *cadeia*:
        - Ou seja, quando se chama a função solicitaNro você deve enviar obrigatoriamente um dado do tipo *cadeia (ou seja*, um texto);
- Após a criação da variável *valor*, há um *ESCREVA*:
    - Você se lembra que quando chamamos o nosso *MÉTODO/FUNÇÃO* solicitaNro colocamos uma informação entre parênteses?
    - Isto se encontra lá no *MÉTODO/FUNÇÃO* inicio;
    - Nossa chamada está assim:

Figura 4 - Um parâmetro para o nosso *MÉTODO/FUNÇÃO*



Fonte: Autores, 2023.

- Aquelas duas informações, “1” e “2”, respectivamente, são conhecidos como parâmetros;
- Certo, mas por que temos um parâmetro quando queremos chamar nosso *MÉTODO/FUNÇÃO*?
- É mais simples, do que você pode imaginar: porque nós queremos isso;
- Perceba que nosso *MÉTODO/FUNÇÃO* é definido assim:

Figura 5 - O parâmetro do nosso *MÉTODO/FUNÇÃO*

```
funcao inteiro solicitaNro(cadeia ordem)
```

Fonte: Autores, 2023.

- Dentro dos parênteses temos um “cadeia ordem”;
- O que isso quer dizer?
- Nosso *MÉTODO/FUNÇÃO* solicitaNro SEMPRE espera receber um valor do tipo cadeia, o qual será guardado na variável *ordem*;
- Essa variável ordem será criada a cada vez que esse *MÉTODO/FUNÇÃO* for invocado;
- A cada vez que eu acessar o *MÉTODO/FUNÇÃO* a variável *ordem* está vazia esperando receber algo;
- Ou seja, para que possamos usar o *MÉTODO/FUNÇÃO* solicitaNro SEMPRE precisamos informar alguma informação do tipo cadeia;
- Vamos voltar para o nosso código-fonte:
- Por enquanto, ele está assim:

Figura 6 - Código-fonte com a aplicação de *MÉTODO/FUNÇÃO*

```
programa
{
	funcao inicio()
	{
		inteiro valor1, valor2, resultado

		valor1 = solicitaNro("1")
		valor2 = solicitaNro("2")

		resultado = valor1 + valor2

		limpa()

		escreva("\nO resultado desta soma é: " + resultado)

	}

	funcao inteiro solicitaNro(cadeia ordem)
	{
		inteiro valor
		escreva("\nInforme o " + ordem + "º valor:")
		leia(valor)

		retorne valor
	}
}
```

Fonte: Autores, 2023.

- Então, bora prosseguir;
- O que faremos agora?
- Percebeu que temos um escreva ali no final do *MÉTODO/FUNÇÃO* inicio?
- Vamos criar um novo *MÉTODO/FUNÇÃO* para realizar aquela saída de dados;
- Logo abaixo de tudo o que já foi construído, faremos a criação do seguinte trecho:

Figura 7 - Um novo MÉTODO/FUNÇÃO sendo criado

```
funcao imprimeTxt(inteiro soma)
{
	escreva("\nO resultado desta soma é: " + soma)
}
```

Fonte: Autores, 2023.

- Pronto! Temos um novo *MÉTODO/FUNÇÃO*!
- Nossa ideia é só fazer a saída da variável *soma*;
- Mas peraí, que variável *soma*;
- Perceba que nossa variável lá no inicio era *resultado*, mas no imprimeTxt é *soma*;

- O que aconteceu?
- Quando eu digo imprimeTxt(resultado) **NÃO ESTOU ENVIANDO VARIÁVEL *RESULTADO***;
- O que estou dizendo é que eu quero que o *MÉTODO/FUNÇÃO* imprimeTxt receba o **VALOR DA VARIÁVEL *RESULTADO***;
- Pense, por exemplo:
    - *valor1* = 7
    - *valor2* = 5
    - *resultado* = Valor1 + Valor1
        - *resultado* = 12
    - Eu não envio resultado para imprimeTxt;
    - Eu envio o valor contido em *resultado*;
    - Ou seja, eu envio o valor 12.
- Foi possível compreender?
- Se sim, excelente!
- Se não, chama a gente!
- Por fim, esse será nosso código-fonte:

Figura 8 - Código-fonte final de *MÉTODO/FUNÇÃO*

```
programa
{
    funcao inicio()
    {
        inteiro valor1, valor2, resultado

        valor1 = solicitaNro("1")
        valor2 = solicitaNro("2")

        resultado = valor1 + valor2

        limpa()

        imprimeTxt(resultado)
    }

    funcao inteiro solicitaNro(cadeia ordem)
    {
        inteiro valor
        escreva("\nInforme o " + ordem + "º valor:")
        leia(valor)

        retorne valor
    }

    funcao imprimeTxt(inteiro soma)
    {
        escreva("\nO resultado desta soma é: " + soma)
    }
}
```

Fonte: 2023.

- Isso aí jovem *padawan*!
- Se o seu código-fonte está igual ou parecido, mas funcionando corretamente, excelente!
- Se não, tente corrigir, observando onde você se perdeu.
- Ficou com dúvidas?

CHAMA A GENTE!

REFERÊNCIAS BIBLIOGRÁFICAS

DEITEL, Paul; DEITEL, Harvey. **Java**: como programar. 8. ed. São Paulo, SP: Pearson Prentice Hall, 2010. 1144 p.