Qualificação Profissional de Assistente de Desenvolvimento de Sistemas

LÓGICA DE PROGRAMAÇÃO

GEEaD - Grupo de Estudos de Educação a Distância
Centro de Educação Tecnológica Paula Souza
São Paulo – SP, 2019

Expediente

PROGRAMA NOVOTEC VIRTUAL
GOVERNO DO ESTADO DE SÃO PAULO
EIXO TECNOLÓGICO DE INFORMAÇÃO E COMUNICAÇÃO
QUALIFICAÇÃO PROFISSIONAL DE ASSISTENTE DE DESENVOLVIMENTO DE SISTEMAS
LÓGICA DE PROGRAMAÇÃO

PÚBLICO ALVO: ALUNOS DA 3ª SÉRIE DO ENSINO MÉDIO
TEMPO DE INTEGRALIZAÇÃO: 34 SEMANAS

Autores:
Eliana Cristina Nogueira Barion
Marcelo Fernando Iguchi
Paulo Henrique Mendes Carvalho
Rute Akie Utida

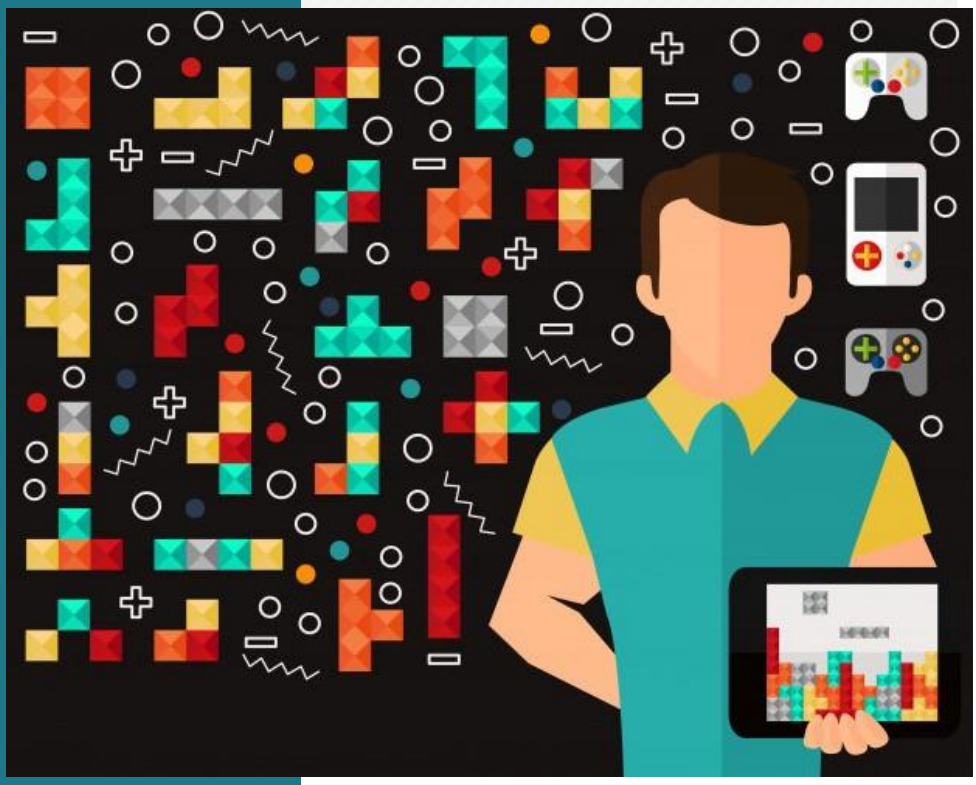
Revisão Técnica:
Sandra Maria Leandro

Revisão Gramatical:
Juçara Maria Montenegro Simonsen Santos

Editoração e Diagramação:
Flávio Biazim

AGENDA 2

DESENVOLVENDO A LÓGICA



MERGULHANDO NO TEMA...

Variável

Durante os seus estudos, você notou que até o momento trabalhamos apenas com números e palavras fixas? Ou seja, não pudemos alterar o seu conteúdo. Essa é a definição de uma **Constante**, isto é, um local na memória do computador que armazena um dado que não se altera ao longo da execução do programa.

Se um computador trabalha em constante interação com o usuário, por que até este momento não trabalhamos com esta interação?

A resposta é simples! Antes de iniciarmos a interação homem-máquina efetivamente, precisamos compreender os aspectos básicos de qualquer Linguagem de Programação, sendo assim, agora que você já conhece estes aspectos estamos prontos para trabalharmos um novo conceito: **Variável**.

O computador é um objeto que não possui a capacidade de pensar por conta própria, sendo necessário sempre uma programação para ensiná-lo a trabalhar.

Quando iniciamos uma interação homem-máquina, não há como o computador saber o que o usuário fará na sequência de seus atos, por isto, devemos "ensiná-lo" a se preparar para uma interação com o usuário do computador.

Esta interação é feita por meio de uma estrutura chamada **variável**. Ela consiste na alocação de um pedacinho da memória RAM do computador para que ele receba uma informação vinda de um dispositivo de entrada de dados, no nosso caso, o teclado.

Para entendermos este conceito, imagine a seguinte situação:

Em sua primeira aula de Lógica de programação, você conheceu um colega de turma chamado Juvenal. Para que em um futuro próximo vocês possam trocar informações sobre as atividades propostas, você decidiu pedir ao Juvenal seu número de telefone.

No momento em que você pede ao Juvenal o seu número de telefone, você inconscientemente prepara um lugar para armazená-lo, podendo ser em seu cérebro, celular ou em um papel, não é mesmo?

Você sabia qual seria o tipo de dado (Você saberia quais seriam os números) que o Juvenal lhe passaria? Provavelmente não, mas saberia com certeza que ele lhe passaria um número de telefone.

Resumindo: Se você fosse um computador, estaria utilizando uma variável (espaço na memória) do tipo numérica para receber o dado que seria passado pelo Juvenal (agente externo ao programa). Fácil, não?

Logo, se você estivesse criando um programa para que o Juvenal e outros amigos inserissem seus números de telefone para que você os consulte depois, precisaria ensinar o computador que ele deve reservar um pequeno espaço em sua memória RAM para receber estes valores.

Será que para o computador é tão simples somar 1+1 como para nós, seres humanos?

Vamos analisar ...

Primeiramente é necessário armazenar o primeiro número fornecido pelo usuário na memória do computador.

- Ler e armazenar o primeiro número seria a primeira tarefa.

Por que armazenar?

Porque ele somará o primeiro número com o próximo número fornecido! Enquanto isto, ele deve memorizar o primeiro número, senão, quando fornecermos o próximo, ele já esqueceu o primeiro!

- Ler e armazenar o segundo número pelo mesmo motivo que o primeiro!
- Executar a operação de soma
- Memorizar o resultado da soma para mostrar em sua tela!

Quando falamos em memorizar, estamos dizendo que o dado deve ser colocado na memória do computador. Inserimos valores na memória de um computador por meio de **VARIÁVEIS**.

Memória do Computador

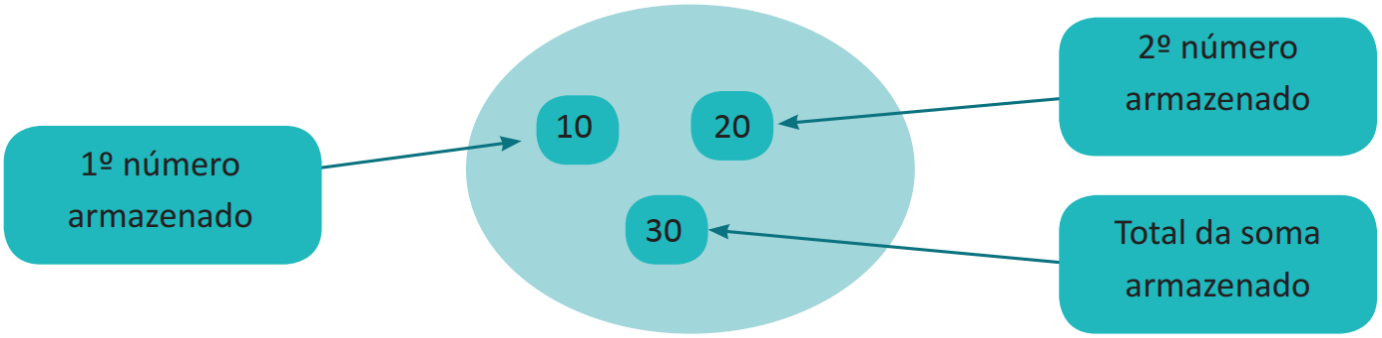


Imagem 03 – GEEaD – Memória do Computador. A imagem faz uma analogia com a memória do computador armazenando três números: 1º número armazenado é o 10, o 2º número armazenado é o 20 e o 3º número é a soma dos dois primeiros, armazenando o número 30.

Portanto, temos que dar nome aos lugares onde estes valores estão armazenados e indicar qual tipo de dado eles podem. Fazemos isto por meio de declaração de Variável. Veja:

Para a caixinha onde está armazenado o 1º número (10), poderia escolher um nome qualquer desde que:

- Não seja número ou que comece com um número;
- Não tenha acento;
- Não tenha espaços em branco!
- Não utilize caracteres especiais

Estas são algumas regras para declaração de nome de variável!

Operadores

Quando utilizamos a lógica, sendo ela em pseudocódigo ou em uma Linguagem de Programação, devemos utilizar alguns símbolos, ou palavras, para que o computador entenda o que queremos que ele faça, estes símbolos são chamados de Operadores.

Existem muitos tipos de operadores, mas neste momento estudaremos os operadores aritméticos, relacionais e lógicos.

Operadores Aritméticos

Se, por acaso, for necessário desenvolver uma operação matemática utilizando pseudocódigo ou uma linguagem de programação, você precisará dos operadores aritméticos para criar o seu programa. Apesar do computador utilizar a mesma regra da matemática para a resolução de cálculos, a simbologia utilizada nem sempre é a mesma da matemática. A seguir, você conhecerá os operadores aritméticos e sua simbologia em pseudocódigo em Java, linguagem que utilizaremos a partir da próxima agenda para implementar os pseudocódigos. Veja também a sua utilização e exemplo:

Nome da operação	Sintaxe em pseudocódigo	Sintaxe em Java	Função	Exemplo
Soma	+	+	Efetua a soma entre 2 valores numéricos.	3 + 2 = 5
Subtração	-	-	Efetua a subtração entre 2 valores numéricos.	3 – 2 = 1
Multiplicação	*	*	Efetua a multiplicação entre 2 valores numéricos.	2 * 4 = 8
Divisão	/	/	Efetua a divisão entre 2 valores numéricos.	4 / 2 = 2
Potenciação	exp(b,e) (b = base e e = expoente)	^	Efetua a potenciação entre 2 valores	Exp(3,2) = 9 (Pseudocódigo) 3 ^ 2 = 9 (Java)
Resto da Divisão	mod	%	Efetua a divisão entre 2 valores, mas retorna o valor do resto da divisão.	3 mod 2 = 1 (Pseudocódigo) 3 % 2 = 1 (Java)
Concatenação	+	+	Junta 2 valores do tipo caractere.	"Lógica" + "Programação" = "LógicaProgramação"

Imagem 04 - GEEaD – Tabela de Operadores Aritméticos - a sintaxe usada no pseudocódigo e na linguagem Java é a mesma da matemática para os operadores soma, subtração, multiplicação, divisão. Para representar a potenciação no pseudocódigo usa-se a função exp onde b é a base e o expoente, em Java usa-se o circunflexo. Para o resto da divisão em pseudocódigo usa-se o mod em Java a porcentagem. A concatenação é representada pela soma.

Operadores Relacionais

Estes operadores são os responsáveis por efetuar comparações entre dados, com o objetivo de mostrar ao programa como proceder dependendo da situação apresentada. O programa de computador verá o resultado de uma comparação em duas situações lógicas, sendo verdadeiro ou falso. Assim como os operadores aritméticos, segue uma pequena tabela indicando suas finalidades:

Nome do Operador Relacional	Sintaxe em pseudocódigo	Sintaxe em Java	Função	Exemplo	Resultado
Maior	>	>	Compara se o primeiro valor é maior que o segundo valor.	7 > 3	Verdadeiro
Menor	<	<	Compara se o primeiro valor é menor que o segundo valor.	(3+1) < (5*0)	Falso
Maior ou Igual	>=	>=	Compara se o primeiro valor é maior ou igual ao segundo valor.	(4 + 4) >= 8	Verdadeiro
Menor ou Igual	<=	<=	Compara se o primeiro valor é menor ou igual ao segundo valor.	4 <= 4	Verdadeiro
Igual	=	==	Compara se o primeiro valor é igual ao segundo valor.	3 = 2 (Pseudocódigo) 3 == 2 (Java)	Falso
Diferente	<>	!=	Compara se o primeiro valor é diferente do segundo valor.	7 <> 3 (Pseudocódigo) 7 != 3 (Java)	Verdadeiro

Imagem 05 - GEEaD – Tabela de Operadores Relacionais – os operadores maior e menor são os mesmos da matemática, já os operadores maior ou igual e menor ou igual são representados pela sintaxe de maior e o igual na frente, o mesmo ocorre para o menor com sinal de menor e igual na frente, isto para o pseudocódigo e para o Java. No sinal de igual usa-se a mesma sintaxe da matemática para o pseudocódigo e em Java usa-se duas vezes o sinal de igual. No pseudocódigo o sinal de diferente é representado pelo sinal de menor e maior e em Java é o ponto de exclamação e igual.

Operadores Lógicos

Os operadores lógicos são responsáveis pela elaboração de comparações especiais, possibilitando que uma única expressão de comparação receba mais de um operador relacional. Ele geralmente é utilizado em situações complexas, por exemplo:

Jovanir, um jovem de 20 anos, gostaria de saber se na próxima eleição ele será obrigado a votar ou se poderá votar de modo facultativo.

Para ser obrigado a votar, o eleitor deve ter a idade maior ou igual a dezoito anos e também o eleitor deve ter menos que 70 anos.

Neste caso, para que a expressão seja corretamente resolvida, necessitamos de um operador Lógico, que serão apresentados a seguir:

Nome do Operador Lógico	Sintaxe em pseudocódigo	Sintaxe em Java	Função	Exemplo	Resultado
E	E	&&	Para que o resultado da comparação seja verdadeiro, os dois lados da expressão devem ser verdadeiros.	(16 >= 16) E (16 < 18) (Pseudocódigo) (16 >= 16) && (16 < 18) (Java)	Verdadeiro
OU	OU	||	Para que o resultado da comparação seja verdadeiro, apenas um dos lados da expressão deve ser verdadeiro.	((3 + 4) < 5) OU ((3*2)=6) (Pseudocódigo) ((3+4) < 5) || ((3*2)==6) (Java)	Verdadeiro
NÃO	não	!	Inverte o resultado da expressão, ou seja, caso a expressão seja verdadeira, se tornará falso e vice-versa.	NAO(4 < 8) (pseudocódigo) ! (4 < 8) (Java)	Falso

Imagem 06 - GEEaD – Tabela de Operadores Lógicos – no pseudocódigo os operadores e, ou e não são representados na forma que são escritos, em Java o e é representado pelo e comercial duas vezes, o ou é representado por duas barras verticais e o não pelo ponto de exclamação.

Observando a tabela dos Operadores Lógicos, podemos concluir que a expressão que resolveria o problema do Jovanir seria: ((20 >=18) && (20 < 70)).

Como a idade dele é maior ou igual a 18 e também é menor que 70, Jovanir é obrigado a votar.

A tabela Verdade

Uma forma muito simples de lembrar qual é o operador correto para satisfazer uma expressão é utilizando a Tabela Verdade. Com ela podemos prever e entender melhor o funcionamento dos Operadores Lógicos.

A tabela consiste em separamos a comparação em dois blocos, sendo o primeiro antes do Operador Lógico, e o segundo logo após o operador. Para simularmos os resultados, definimos os resultados como verdadeiro (V) ou falso (F), para facilitar a simulação e não perdemos tempo com a resolução das expressões.

Na tabela, verificamos todas as possibilidades, sendo ambas as comparações verdadeiras, ambas as comparações falsas ou apenas uma das comparações verdadeira, assim testamos as possibilidades que o operador pode proporcionar, como as tabelas a seguir nos mostram:

Operador E		
Entrada 1	**Entrada 2**	**Saída**
V	V	V
V	F	F
F	V	F
F	F	F

Operador OU		
Entrada 1	**Entrada 2**	**Saída**
V	V	V
V	F	V
F	V	V
F	F	F

Imagem 07 - GEEaD – Tabela Verdade – no operador E a saída será verdadeira somente se todas as entradas forem verdadeiras, caso contrário a saída será falsa. Para o operador OU se tivermos uma entrada verdadeira a saída será verdadeira e falsa somente quando todas as entradas forem falsas.

*Saída é o resultado das entradas de dados.

Aplicando a tabela Verdade em Pseudocódigo e Java temos:

Operador NÃO	
Entrada	**Saída**
V	F
F	V

Para o operador NÃO, o resultado da expressão é apenas invertido.

Operador NAO (!)		
Expressão em Pseudocódigo	**Expressão em Java**	**Resultado**
NAO V	! V	FALSO
NAO F	! F	VERDADEIRO

Imagem 08 - GEEaD – Operador "NÃO"- para o operador Não usamos o inverso, se a entrada for verdadeira a saída será falsa, se a entrada for falsa a saída será verdadeira.

Operador E (&&)		
Expressão em Pseudocódigo	**Expressão em Java**	**Resultado**
V E V	V && V	V
V E F	V && F	F
F E V	F && V	F
F E F	F && F	F

Note que assim como na definição do operador E, o resultado da expressão foi verdadeiro, apenas quando ambos os lados da expressão são verdadeiros.

Imagem 09 - GEEaD – Operador "E" – a tabela apresenta para as entradas verdadeiras o resultado será verdadeiro, para uma entrada falsa o resultado será falso.

Operador OU (||)		
Expressão em Pseudocódigo	Expressão em Java	Resultado
V OU V	V || V	V
V OU F	V || F	V
F OU V	F || V	V
F OU F	F || F	F

Desta vez, apenas quando ambas as expressões são falsas o resultado foi falso.

Imagem 10 - GEEaD – Operador "OU" – a tabela apresenta para uma entrada verdadeira o resultado será verdadeiro, e falso somente quando todas as entradas forem falsas o resultado será falso.

Para entender mais sobre os Operadores Aritméticos, Relacionais e Lógicos, assista à videoaula do Prof. Sandro Valérius, disponível em https://www.youtube.com/watch?v=ntCQmyfhA30.

VOCÊ NO COMANDO

Aplicação da lógica na resolução de problemas

Imagem 11: Freepik – Homem pensativo escrevendo

Manoel é um comerciante, dono de uma loja de ferragens para construção onde ele vende pregos, parafusos, porcas e braçadeiras por unidade. Para que Manoel saiba que obteve lucro em sua venda ele deve ter vendido pelo menos 100 pregos e 70 parafusos, mas se ele vender 30 braçadeiras e 10 porcas também ficará no lucro. Certo dia, Manoel vendeu 300 pregos, 180 parafusos, 9 porcas e 32 braçadeiras, sendo assim ele obteve _____ (Lucro/Prejuízo).

A seguir, confira se você conseguiu resolver os desafios propostos!

Sabemos que a meta é 100 pregos e 70 parafusos, mas a meta também pode ser considerada caso ele venda pelo menos 10 porcas e 30 braçadeiras, neste caso podemos montar a seguinte expressão:

((?? >= 100) E (?? >= 70)) OU ((?? >= 10) E (?? >= 30))
PREGOS PARAFUSOS PORCAS BRAÇADEIRAS

Agora que indicamos a expressão, basta substituirmos as interrogações pelos valores que foram vendidos pelo Manuel, sendo assim:

((300 >=100) E (180 >= 100) OU ((9 >= 10) E (32 >= 30))

Resolvendo cada lado da expressão principal:

(VERDADEIRO E VERDADEIRO) OU (FALSO E VERDADEIRO)
VERDADEIRO OU FALSO
VERDADEIRO

Logo, no dia em questão, Manuel ficou com LUCRO (Verdadeiro)

Observe as seguintes expressões lógicas e indique se o resultado é Verdadeiro ou Falso.
a)

Linguagem	**Expressão**	**Resultado**
Pseudocódigo	((33*1) > 27) OU (NAO (18<20))	
Java	((33*1) > 27) || (!(18 <20))	VERDADEIRO*

*Passo a passo para a resolução do exercício:

Passo 1: Separamos as expressões pelo Operador OU, sendo assim, temos as expressões

((33 * 1) > 27) OU (NÃO (18 < 20))

Passo 2: Resolvemos as expressões individualmente:

Expressão 1:	Expressão 2:
((33*1) >27)	NAO (18<20)
33*1 = 33	NAO (18 é MENOR que 20?)
33 é maior que 27?	NAO (VERDADEIRO)
VERDADEIRO	FALSO

Logo, VERDADEIRO OU FALSO, segundo a tabela verdade é: VERDADEIRO

Os operadores lógicos são lidos e resolvidos pela ordem em que forem lidos, salvo no caso do operador estar dentro de parênteses, neste caso ele terá prioridade como o caso do operador NAO na resolução anterior.

b)

Linguagem	Expressão	Resultado
Pseudocódigo	NAO(3>4)	
Java	!(3>4)	

c)

Linguagem	Expressão	Resultado
Pseudocódigo	(5>(3+1)) OU (8 > 7)	
Java	(5>(3+1)) || (8 > 7)	

d)

Linguagem	Expressão	Resultado
Pseudocódigo	(((exp(3,2) +1) = 10) E (NAO (10 = 10))	
Java	(((3^2)+1) == 10) && (!(10==10))	

Confira a resolução dos desafios:

b)

Linguagem	Expressão	Resultado
Pseudocódigo	NAO(3>4)	Verdadeiro
Java	!(3>4)	

NAO(3>4)
NAO(FALSO)
VERDADEIRO

c)

Linguagem	Expressão	Resultado
Pseudocódigo	(5>(3+1)) OU (8 > 7)	Verdadeiro
Java	(5>(3+1)) || (8 > 7)	

(5 > (3 + 1)) OU (8 > 7)
VERDADEIRO OU VERDADEIRO
VERDADEIRO

d)

Linguagem	Expressão	Resultado
Pseudocódigo	(((exp(3,2) +1) = 10) E (NAO (10 = 10))	Falso
Java	(((3^2)+1) == 10) && (!(10==10))	

(((exp(3,2) + 1) = 10) E (NAO (10 = 10))
VERDADEIRO E FALSO
FALSO