Agenda 02: Desenvolvendo a lógica

Conceitos trabalhados:

Sumário

Momento de Reflexão

Olá, estudante ;)

Boas-vindas à segunda agenda do Módulo 1! Para começar, convidamos você a explorar a tirinha:

Você já passou por algum problema gerado por uma deficiência de comunicação em situações do cotidiano? Provavelmente, sim! Quando não há uma comunicação **clara**, **objetiva** e **padronizada** nas ações, a chance de que as informações sejam repassadas de forma errada é muito grande. Se pensarmos que até na comunicação entre pessoas é necessário estabelecer padrões para que exista entendimento, o que diremos da comunicação entre humanos e computadores?

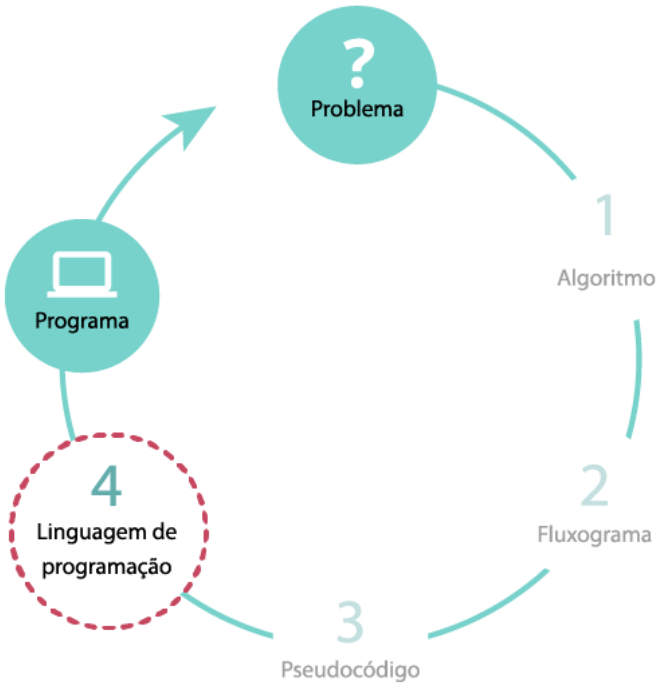
Assim, esta agenda irá apresentar algumas **técnicas para ajudar você no desenvolvimento da lógica de programação** e promover a "ordenação do pensamento", de forma que você possa enxergar essa lógica.

Boa sorte!

Por que Aprender?

Na agenda anterior, você teve seu primeiro contato com a lógica de programação. Você viu que os programas de computador são feitos para auxiliar as pessoas na realização de tarefas e que, como desenvolvedor, você deverá criar instruções que indiquem à máquina quais passos ela deve seguir para realizar tais tarefas.

Você também viu que existem quatro etapas para o desenvolvimento de um software, e conheceu as três primeiras. Nesta agenda, você avançará seus estudos, tendo um primeiro contato com a quarta etapa: **a linguagem de programação**.



O desenvolvimento de programas e algoritmos depende totalmente do **completo entendimento de sua estrutura de linguagem**. Saber os comandos que introduzem informações nos sistemas, bem como aqueles que exibem informações processadas é de vital importância para alguém que queira ingressar ou evoluir no ramo da programação de computadores. Nesta agenda, você verá como tudo isto se relaciona e qual é o procedimento que os programadores devem adotar para **alimentar um programa e extrair dele as informações requeridas**.

Para Começar o Assunto

Algoritmo, pseudocódigo e fluxograma são conceitos que você já conhece. À essa lista, iremos acrescentar ainda dados e variáveis. Quantos termos! Mas o que na verdade querem dizer todas essas palavras ainda “misteriosas”?

Antes de mais nada, precisamos entender que **cada uma delas ocupa um espaço bem definido** e importante dentro da lógica computacional. Então, é preciso entender direitinho o que cada uma significa. Já vimos, no capítulo anterior, que algoritmo é como se fosse uma **receita de bolo**. Para termos o bolo pronto, precisamos seguir alguns passos.

A maneira como esses passos são descritos é que indica se estamos usando um “pseudocódigo” (linguagem parecida com a linguagem humana) ou um “fluxograma” (diagrama com formas geométricas).

Mas e as **variáveis** e os **dados**? Vamos continuar nossa analogia culinária pensando nos armários da cozinha e em itens necessários para seu bolo! Os materiais a seguir auxiliarão você na compreensão desses conceitos.

Mergulhando no Tema

Um olhar inicial sobre variáveis e dados

Na agenda anterior, você viu que algoritmo é uma sequência de instruções ordenadas de forma lógica, como uma receita de bolo. Essa receita é escrita de uma determinada maneira, podendo ser registrada como pseudocódigo ou fluxograma. Agora, apresentamos os conceitos de **variáveis** e **dados**:

Bem, podemos entender uma **variável** como uma parte definida no armário **em que guardamos** alguns ingredientes e itens necessários à receita (**dados**). Para que qualquer pessoa consiga localizar os itens na cozinha, podemos etiquetar as áreas do armário indicando o nome do que se encontra ali. Da mesma forma, a variável precisa ser identificada (nome) e possuir um único tipo de dado (por exemplo, somente tipos de farinha).

Dado

Valor ou operação que uma variável pode armazenar.

Variáveis

Durante os seus estudos, você notou que até o momento não armazenamos nenhum dado, trabalhamos apenas com procedimentos, como o caso do algoritmo “Fritar ovos”.

A partir de agora, trabalharemos com valores **variáveis** e **fixos**, por exemplo, o nome do cliente de uma loja que varia de cliente para cliente (**variável**) ou valores que são fixos (**constantes**), como é o caso de um valor que se mantém para todos os registros, como o valor de PI (3.14159) ou a quantidade de meses do ano, que sempre será 12. Nesse caso, não há alteração de

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.

Constante

Local na memória do computador que armazena um dado que não se altera ao longo da execução do programa.

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

A resposta é simples! Antes de iniciarmos efetivamente a interação homem-máquina, precisamos compreender os aspectos básicos de qualquer Linguagem de Programação, sendo assim, agora que você já conhece esses 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 isso, devemos “ensiná-lo” a se preparar para uma interação com o usuário do computador.

Essa 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.

Variável

Área reservada na memória RAM do computador para armazenar o valor que conterá em determinado tempo de execução do programa.

Para melhor compreender esse conceito, imagine a seguinte situação:

Em sua primeira aula de Lógica de Programação, você conheceu um colega de turma chamado Caio. Para que em um futuro próximo vocês possam trocar informações sobre as atividades propostas, você decidiu pedir a ele seu WhatsApp, ou seja, seu número de telefone.

No momento em que você pede ao Caio o número do telefone, você inconscientemente prepara um lugar para armazená-lo: seu celular, um pedaço um papel ou até mesmo na sua memória, não é mesmo?

Você saberia, de antemão, qual seria o número que Caio lhe passaria, ou seja, quais seriam os tipos de dados que receberia? 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 Caio (agente externo ao programa).

Fácil, não?

Logo, se você estivesse criando um programa para que o Caio 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 esses valores.

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

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

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

Por que armazenar?

Porque o computador somará o primeiro número com o próximo número fornecido! Enquanto o usuário indica o segundo número, o computador deverá ter registrado na memória o primeiro número. Se esse passo não for feito, quando fornecermos o próximo número, o computador já terá esquecido o primeiro!

E depois?

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

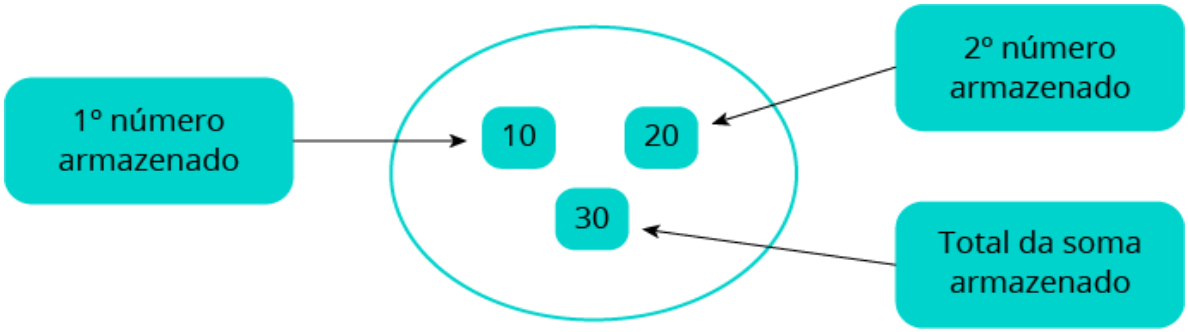
2. Ler e armazenar o segundo número, pelo mesmo motivo que o primeiro!

3. Executar a operação de soma.

4. 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 das **variáveis**.

Agora, veja esse esquema que representa a memória do computador:



Você percebeu que o computador precisará registrar três variáveis, correto? É necessário **dar nome** aos lugares onde esses valores estão armazenados e indicar **qual tipo de dado** eles podem ser. Fazemos isso por meio da **declaração de variável**, que conta com algumas regras:

Regras para a declaração de variáveis

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

~~1numero~~	Não seja número ou que comece com um número.
~~primeironúmero~~	Não tenha acento.
~~primeiro numero~~	Não tenha espaços em branco!
~~prim°numero~~	Não utilize caracteres especiais

Utilizando nosso exemplo, um nome de variável possível seria: primeironumero.

Operadores

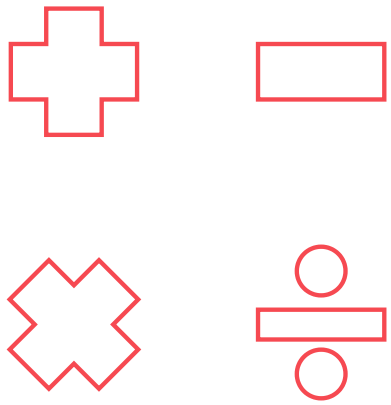
Quando utilizamos a lógica de programação, sendo ela em pseudocódigo ou em uma linguagem de programação, devemos utilizar alguns **símbolos** ou **palavras** para que o computador entenda o que queremos que ele faça, chamados **operadores**.

Operadores

Símbolo ou palavra específica que indica ao computador partes da instrução.

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

Operadores Aritméticos

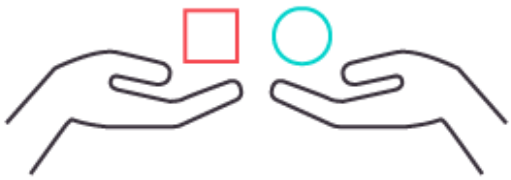


Se for necessário desenvolver uma **operação matemática** utilizando pseudocódigo ou uma linguagem de programação, você precisará dos operadores aritméticos para criar o seu programa.

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 unidade para implementar os pseudocódigos. Veja também a sua utilização e exemplos:

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”

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

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:

> Paulo, 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, é necessário utilizar Operadores Lógicos, 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 + 2) < 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

Observando a tabela dos Operadores Lógicos, podemos concluir que a expressão que resolveria o problema de Paulo seria:

((20 >=18) && (20 < 70))

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

A Tabela Verdade (operadores lógicos)

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 **separarmos a comparação em dois blocos**, sendo o **primeiro antes** do Operador Lógico, e o segundo **logo após** o operador. Para simularmos os resultados, definimos as respostas como verdadeiras (V) ou falsas (F), facilitando a simulação e evitando o uso de tempo na resolução das expressões.

Usando esse tipo de tabela verificamos todas as possibilidades de resultados: ambas as comparações podem ser verdadeiras, ambas falsas ou apenas uma delas verdadeira. Analise as tabelas a seguir:

Tabela verdade do operador E

TABELA VERDADE DO OPERADOR E

Entrada 1	Entrada 2	Saída
V	V	V
V	F	F
F	V	F
F	F	F

No operador E, a saída será verdadeira somente se todas as entradas forem verdadeiras, caso contrário a saída será falsa.

A entrada 1 é o primeiro bloco. A entrada 2, o segundo. A saída é o resultado das entradas de dados.

Aplicando a tabela verdade em **Pseudocódigo e Java** temos:

OPERADOR E (& &)

Expressão em Pseudocódigo	Expressão em Java	Resultado
VEV	V&&V	V
VEF	V&&F	F
FEV	F&&V	F
FEF	F&&F	F

Note que, assim como na tabela superior, o resultado da expressão foi verdadeiro apenas quando ambos os lados da expressão são verdadeiros.

Para entender mais sobre os **Operadores Aritméticos, Relacionais e Lógicos**, assista à videoaula do Prof. Sandro Valérius:

Informática - Módulo I - Agenda 11 - Operadores ...

Ampliando Horizontes

Onde encontro mais informações sobre linguagem de programação?

Para aprofundamento dos temas discutidos nesta agenda, seguem abaixo algumas dicas de vídeos e livros que se relacionam com o conteúdo estudado. Essas dicas são muito importantes para você :)

Desenvolvendo a lógica | Professor Anderson Oliveira

Nesta videoaula, o professor Anderson Oliveira retoma alguns conceitos que você estudou até este momento do curso.

Operadores aritméticos, relacionais e lógicos | Professor Sandro Valérius

Nesta videoaula, produzida originalmente para o curso técnico em Informática, o professor Sandro Valérius detalha os operadores aritméticos, relacionais e lógicos.

Curso de Java para iniciantes

Essa playlist (Links to an external site.) oferece um curso grátis para iniciantes em Java, assista às videoaulas 1 a 8.

Lógica de programação: a construção de algoritmos e estrutura de dados. FORBELLONE, André L. V; ELBERSPACHER, Henri Frederico. Editora Pearson, 2000.

Algoritmos: lógica para desenvolvimento de programação. MANZANO, José Augusto N. G; OLIVEIRA, Jayr Figueiredo. Editora Érica, 2007.

Lógica de programação e estruturas de dados com aplicações em Java. PUGA, Sandra; RISSETTI, Gerson. Editora Pearson, 2009.

Java para iniciantes. SCHILDT, Hebert. Editora Bookman, 2015.

Resumindo o Estudo

Nesta agenda, você aprofundou seus estudos em lógica de programação, conhecendo conceitos como variáveis, dados e tendo seu primeiro contato com a linguagem de programação por meio dos operadores. Na próxima agenda você verá a lógica aplicada em Java, aguarde!

Até lá!