

The background of the page is decorated with a collection of 3D blocks in various colors (purple, blue, yellow, green, orange, pink, grey) arranged in a scattered, overlapping manner. Some blocks are solid, while others are hollow. Thin, curved lines in blue, red, and green are also visible, weaving through the blocks.

Apêndice 1

Linguagens de programação e exemplos de programas

Criação de exemplos:

Bruno Bove Barbosa, João Henrique Cunha Rangel e Ricardo Azzi Silva

Introdução

Os conceitos de lógica e de programação abordados neste curso são aplicáveis a diversas linguagens de programação, como Java, SQL, JavaScript, C#, PHP e Visual Basic for Applications (VBA).

Neste apêndice apresentamos uma breve descrição dessas linguagens, seguida de exemplos de declaração de variáveis, uso de operadores e de estruturas de decisão e repetição em simples programas desenvolvidos com cada uma delas.

Java

Criada na década de 90 por profissionais da Sun Microsystems, a **Java** é uma das linguagens de programação orientada a objetos mais utilizadas no mundo, popularizando-se no desenvolvimento de aplicações para Web.

Pelo fato de utilizar boa parte da estrutura da linguagem C++ e conceitos de segurança da linguagem SmallTalk, a Java é tida como uma linguagem de fácil aprendizado, relativamente familiar aos programadores e bastante eficaz, que possibilita o desenvolvimento maciço de aplicações, applets e sistemas embutidos.

Exemplos

A seguir, temos um exemplo de declaração de variáveis em um programa em Java. Note que há linhas que se iniciam com barras duplas (//), e linhas entre `/**` e `*/`. Utilizados para comentar as linhas de código, esses caracteres são inseridos pelo programador para facilitar o entendimento do código. Os comentários são ignorados pelo compilador ao executar o programa.

```
/** Classe de exemplo de variáveis */
public class Variaveis {

    /** Método principal Main, o primeiro método à ser executado */
    public static void main(String[] args) {

        //Tipos de dados primitivos
        byte A = 127;
        System.out.println("O valor de A é:"); System.out.println(A);

        short B = 32767;
        System.out.println("O valor de B é:"); System.out.println(B);

        int C = 2147483647;
        System.out.println("O valor de C é:"); System.out.println(C);

        long D = 9223372036854775807L;
        System.out.println("O valor de D é:"); System.out.println(D);

        float E = 1.12345678F;
        System.out.println("O valor de E é:"); System.out.println(E);

        double F = 1.123456789098765D;
        System.out.println("O valor de F é:"); System.out.println(F);

        boolean G = true;
        System.out.println("O valor de G é:"); System.out.println(G);

        char H = 'J';
        System.out.println("O valor de H é:"); System.out.println(H);

    }

}
```

Apêndice 1

Introdução à Lógica de Programação

4 / 22

Já no código a seguir, temos a utilização dos comandos de fluxo em um programa desenvolvido em Java:

```
/** Classe de exemplo de IF */
public class If {

    /** Método principal Main, o primeiro método à ser executado */
    public static void main(String[] args) {

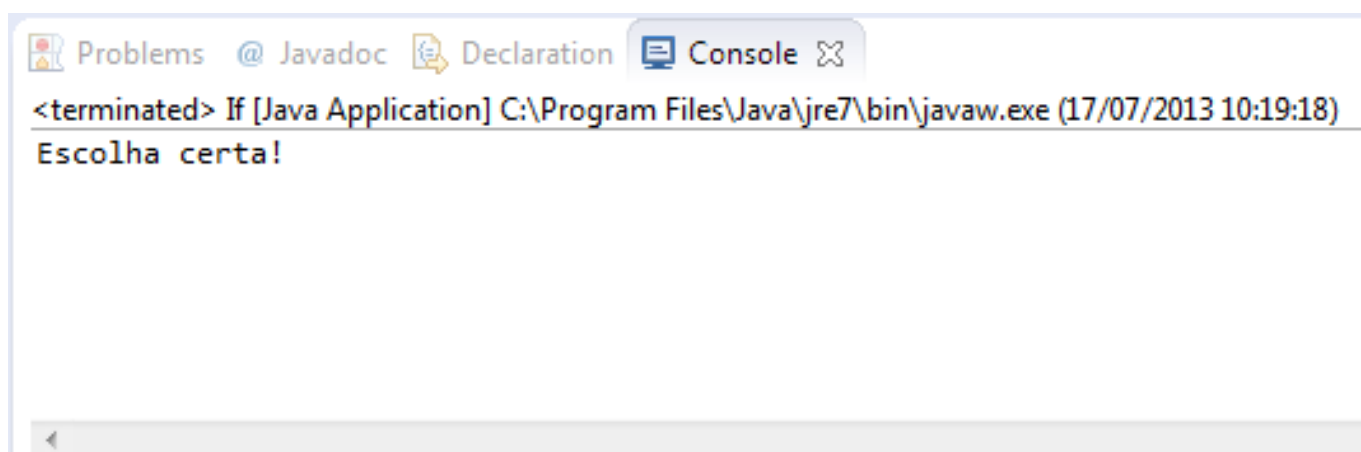
        //Declarando uma variável com o valor TRUE
        boolean alunoImpacta = true;

        //Usando a condição IF para validar o valor da variável
        if(alunoImpacta==true){
            //Resultado se a condição for verdadeira
            System.out.println("Escolha certa!");
        }else{
            //Resultado se a condição não for verdadeira
            System.out.println("Continue tentando!");
        }

    }

}
```

O resultado do código anterior é o seguinte:



Apêndice 1

Introdução à Lógica de Programação

5 / 22

Este outro exemplo demonstra o uso de um laço de repetição **FOR** em um programa feito em Java:

```
/** Classe de exemplo de laço de repetição */
public class For {

    /** Método principal Main, o primeiro método à ser executado */
    public static void main(String[] args) {

        //Declarando variável a com o valor 3
        int a = 3;

        //Iniciando o valor de i em 0 e repetindo o processo até 10
        for(int i=0; i<=10; i++){

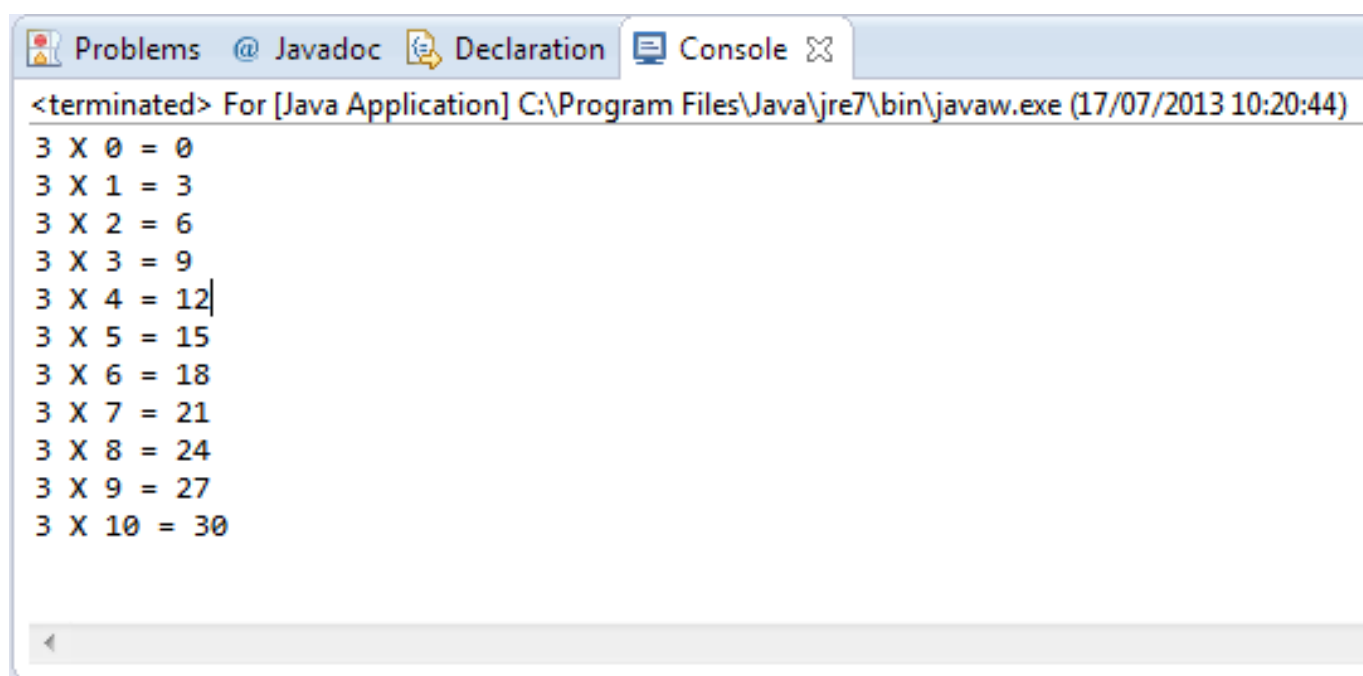
            //Imprimindo mensagem no console
            System.out.println(a+" X "+i+" = "+(a*i));

        }

    }

}
```

O resultado da execução do código que acabamos de ver é o seguinte:



```
<terminated> For [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (17/07/2013 10:20:44)
3 X 0 = 0
3 X 1 = 3
3 X 2 = 6
3 X 3 = 9
3 X 4 = 12
3 X 5 = 15
3 X 6 = 18
3 X 7 = 21
3 X 8 = 24
3 X 9 = 27
3 X 10 = 30
```

SQL

Desenvolvida no início da década de 70, a **SQL (Structure Query Language)** é uma linguagem de programação utilizada para manipulação de banco de dados. É caracterizada por sua exigência sintática rígida e por basear-se no conceito de banco de dados relacional. Na década de 80, a SQL foi adotada como linguagem padrão pela ANSI (American National Standard Institute) e ISO (International Organization for Standardization).

A Microsoft desenvolveu uma implementação para a linguagem SQL padrão ANSI denominada **T-SQL (Transact-SQL)**, que oferece opções adicionais para os comandos SQL, além de novos comandos que permitem o recurso de programação, como os de controle de fluxo, variáveis de memória, entre outros.

Exemplos

O código a seguir exemplifica, em linguagem de banco de dados SQL, a declaração de tipos de dados, a definição de valores para os tipos declarados e a exibição em tela desses tipos. Os trechos precedidos por hífen duplo (--) são comentários, e serão ignorados ao executar o script.

```
--Declarando um tipo de dado
DECLARE @id int --Tipo de dado inteiro
DECLARE @nome varchar(20) --Tipo de dado caractere com tamanho no maximo de 20
caracteres
DECLARE @data datetime
DECLARE @dinheiro money

--Definindo o valor do dado
SET @id = 1
SET @nome = 'BRUNO'
SET @data = GETDATE() --Função GetDate() recupera a data e hora atual.
SET @dinheiro = 11.35

--Mostrando o resultado na tela
PRINT @id
PRINT @Nome
PRINT @data
PRINT @dinheiro
```

No exemplo a seguir, temos a criação de um banco de dados, a criação de uma tabela para o banco de dados e a inserção de dados nessa tabela. Na sequência, temos o trecho de código que realiza uma filtragem para exibição de dados do banco utilizando operadores lógicos estudados durante o curso:

```
--Criando um banco de dados
CREATE DATABASE DatabaseName

--Criando uma tabela
--A propriedade identity define que a coluna somente terá valores unicos
--É definido o nome da coluna e depois o tipo na construção.
CREATE TABLE tb_tablename (id int identity, nome varchar(200))

--Inserindo dados em uma tabela
INSERT INTO tb_tablename(id,nome) VALUES(1,'Bruno')
INSERT INTO tb_tablename(id,nome) VALUES(1,'Tassia')

--Listando os dados
--O caractere * corresponde a todos as colunas da tabela que serão listadas.
SELECT * FROM tb_tablename
SELECT id FROM tb_tablename --irá listar apenas a coluna ID

--Filtrando uma lista utilizando operadores logicos. (=, <=, <, >, >=, !=)
SELECT * FROM tb_tablename WHERE id = 1 --Irá listar o registro com ID igual a 1
SELECT * FROM tb_tablename WHERE id != 1 --Irá listar o registro com ID diferente que 1
SELECT * FROM tb_tablename WHERE id <= 1 --Irá listar o registro com ID menor ou igual a 1
SELECT * FROM tb_tablename WHERE id < 1 --Irá listar o registro com ID menor que 1
SELECT * FROM tb_tablename WHERE id >= 1 --Irá listar o registro com ID maior ou igual a 1
SELECT * FROM tb_tablename WHERE id > 1 --Irá listar o registro com ID maior que 1
```

Apêndice 1

Introdução à Lógica de Programação

8 / 22

Por fim, temos um código que demonstra o uso de uma estrutura condicional **IF** em um programa de banco dados com SQL:

```
--Utilizando estrutura de condição
DECLARE @n int
SET @n = 1
IF @n = 1
    BEGIN
        PRINT 'Condição Verdadeira'
    END
ELSE
    BEGIN
        PRINT 'Condição Falsa'
    END
```

JavaScript

Com possibilidade de execução em diversas plataformas, o **JavaScript** é uma linguagem de script orientada a objeto considerada leve, concisa e de fácil integração com outras aplicações. Vale dizer que JavaScript não possui relação qualquer com Java, sendo esta última uma linguagem mais complexa. Tem grande utilidade do lado do cliente, em um sistema cliente-servidor, como a Internet, em que é possível estender a linguagem básica por meio da disponibilização de objetos de controle para os navegadores (browsers). Do lado do servidor, um dos possíveis benefícios obtidos com a extensão básica da linguagem é a comunicação entre uma aplicação e um banco de dados.

Apêndice 1

Introdução à Lógica de Programação

9 / 22

Exemplos

O exemplo de código de programa em JavaScript a seguir demonstra como é feita a declaração de variáveis nessa linguagem:

```
/* Os tipos de dados no javascript são:
   undefined, boolean, number, string, function e object */

// Toda declaração de variável no javascript deve ser precedida por 'var'
var meuNumero = 10;

// O tipo de dado no javascript é assumido de acordo com o valor inserido na variavel
var minhaString = "Olá mundo!";

// Você pode declarar várias variáveis seguidas, mesmo de tipos diferentes;
var minhaVerdade = true, meuFalso = false;

// Se você não der um valor, a variavel por padrão é do tipo e valor undefined
var minhaVariavel, outraVariavel = undefined;
```

Neste outro exemplo de programa em JavaScript, temos o uso de IF e de loop com FOR e WHILE:

```
// O if no javascript pode ser feito com variáveis de tipo diferente
if (false == "") {
    console.log("Texto vazio é equivalente a falso!");
} else if (false == 0) {
    console.log("Número zero é equivalente a falso!");
} else if (false == []) {
    console.log("Um array vazio é equivalente a falso!");
}

// A ausencia de um objeto é representada por null em javascript
if (null == true || null == false) { // retorna falso
    console.log("null não é equivalente nem a verdadeiro nem a falso.");
} else if (undefined == true || undefined == false) { // retorna falso
    console.log("undefined também não!");
}

for (var i = 0; i < 10; i++) {
    // concatenação no javascript é feita com o sinal de mais
    console.log("Girando: " + i);
}

// Soma também é exercida com o sinal de mais
var dois = 1 + 1; // recebe dois
// mas cuidado!!
var resultado = "1" + 1; // aqui recebe "11" e não 2!

// enquanto você não digitar "minhasenha" na janela, o while irá rodar
var senha = "";
while(senha != "minhasenha") {
    senha = prompt("Digite a sua senha");
}
```

Apêndice 1

Introdução à Lógica de Programação

10 / 22

C#

O **C#** é uma linguagem orientada a objeto utilizada para a criação de diversos tipos de programas, como programas cliente-servidor, programas tradicionais do Windows, programas de banco de dados, além de componentes distribuídos, XML Web Services, entre outros.

Em conjunto com o Visual Studio e outras ferramentas da plataforma .NET, mostra-se uma linguagem de grande utilidade para desenvolver, de maneira fácil e eficiente, vários tipos de programas.

Exemplos

O exemplo de código a seguir demonstra como é feita a declaração de variáveis em um programa desenvolvido com a linguagem C#.

```
//      Não se assuste! Essas bibliotecas básicas são criadas pelo
//      Visual Studio, assim que você cria um novo projeto.
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace ExemploDeLogica1
{
    class Program
    {
        //  TODO programa em C# é iniciado pela função Main()
        static void Main(string[] args)
        {
            //  A declaração de uma variavel no C# é feita definindo o tipo de
            //  variavel e um nome:
            bool verdadeiro_ou_falso; // bool -> 1 bit

            //  Você pode definir um valor para a variavel no momento de sua
            //  criação
            byte AlgunsNumeros = 200; // byte -> 1 byte

            //  C# é CASE SENSITIVE, ou seja, para ele maiúsculas e minúsculas
            //  são diferentes;
            short algunsnumeros; // short -> 2 bytes
```

Apêndice 1

Introdução à Lógica de Programação

11 / 22

```
// Você pode declarar mais de uma variável por linha
int primeiraVariavel, segundaVariavel; // int -> 4 bytes

// Veja o exemplo da declaração de um array de 30 posições
long[] meuLongo = new long[30]; // long -> 8 bytes cada posição do
array

// Um caractere e uma frase
char mander = 'A';
string mensagem = "Olá mundo!";
    }
}
}
```

Este outro exemplo mostra o uso de IF e loop em C#:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
// Adicionando a biblioteca de manipulação de arquivos para o exemplo com o
DO WHILE
using System.IO;

namespace ExemploDeLogica2
{
    class Program
    {
        static void Main(string[] args)
        {
            // Para você escrever um texto comum numa aplicação básica:
            Console.WriteLine("Olá mundo");

            // Console.ReadLine permite que o usuário escreva um valor para o
programa:
            string valor_entregue_pelo_usuario = Console.ReadLine();

            // Comparações em C# devem ser feitas sempre com variáveis do
mesmo tipo, no caso abaixo, string e string
            if (valor_entregue_pelo_usuario == "Olá")
            {
                Console.WriteLine("Bem vindos à Impacta!");
            }
        }
    }
}
```

Apêndice 1

Introdução à Lógica de Programação

12 / 22

```
// Quando o inteiro é declarado dentro do FOR, ele se extingue
quando o FOR é concluído:
for (int i = 1; i <= 10; i++)
{
    // A concatenação em C# é feita com o sinal + e só é possível
com strings, se você deseja
    // concatenar um valor numérico com uma string, deve con-
verte-lo antes para texto
    Console.WriteLine("Você está dentro de um loop! Volta número "
+ i.ToString());
}

// Exemplo de DO WHILE, enquanto o nome do arquivo já existir,
ele tentará um novo,
// com um novo número: arquivo 1.txt, arquivo 2.txt, arquivo
3.txt, arquivo 4.txt...
string meuArquivo; int j = 0;
do
{
    meuArquivo = "arquivo " + j.ToString() + ".txt";
    j++;
} while (File.Exists(meuArquivo));

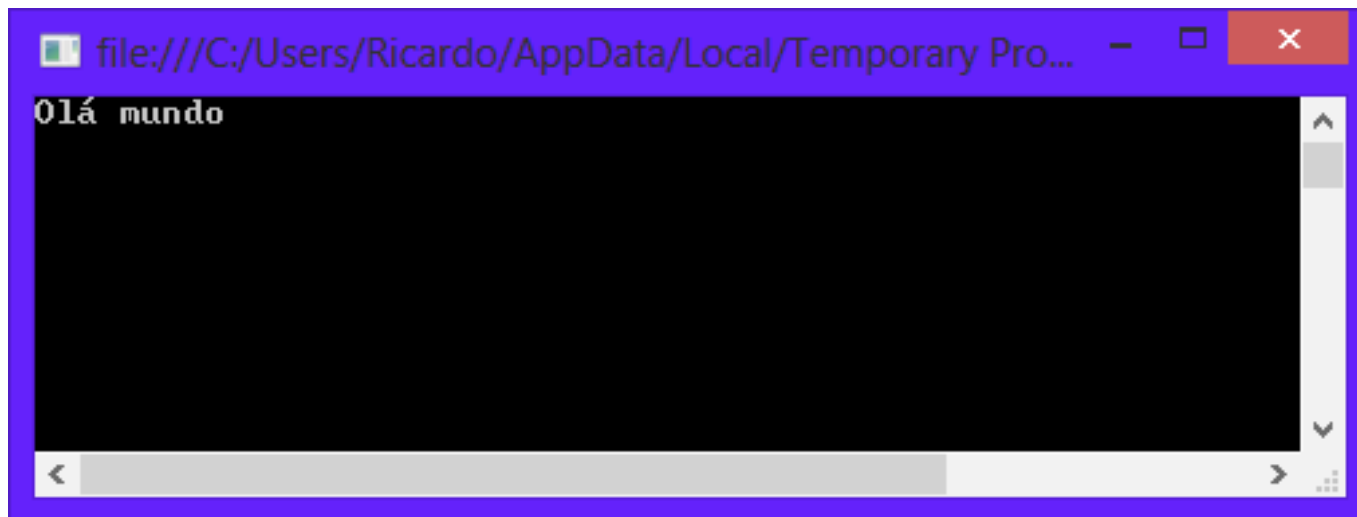
// As aplicações em Console do C# costumam ser concluídas com um
Console.ReadKey(),
// dessa forma o programa só irá encerrar se o usuário precionar
alguma tecla,
// isso dará tempo à ele de ler o que foi escrito na tela
Console.ReadKey();
}
}
}
```

Apêndice 1

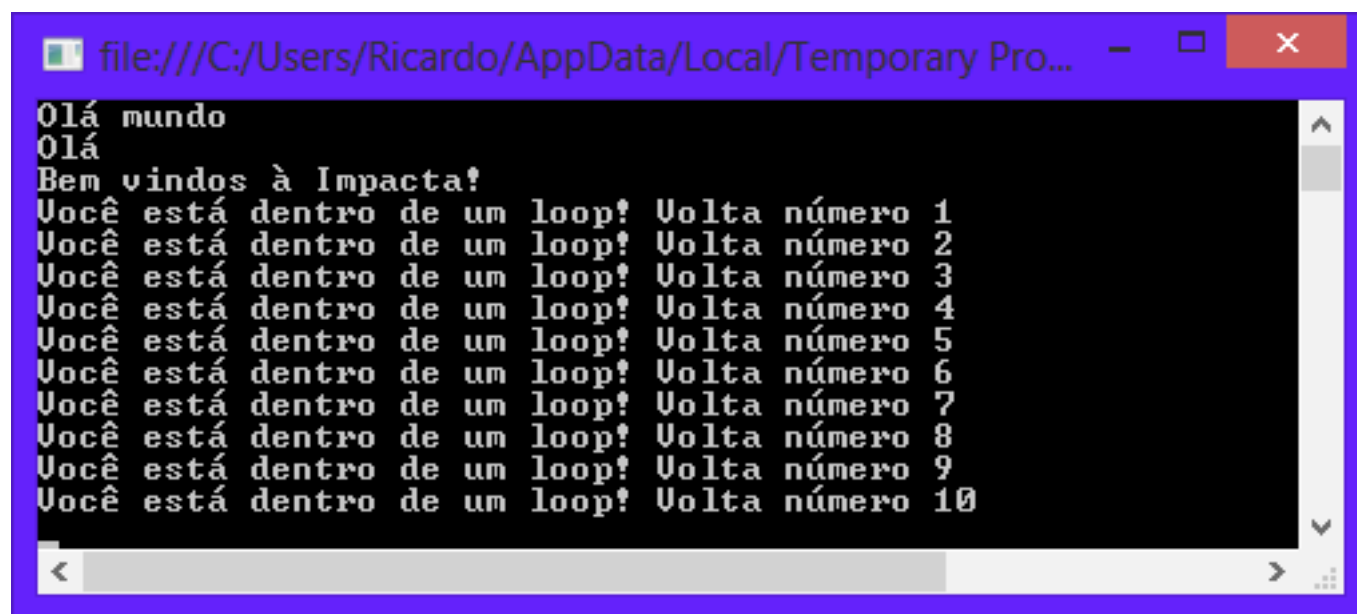
Introdução à Lógica de Programação

13 / 22

Temos, a seguir, o resultado da execução do código anterior. Ao executar o programa, você verá a seguinte tela, com a mensagem **Olá mundo**:



Se digitarmos **Olá** na tela anterior, a mensagem **Bem vindos à Impacta!** será exibida, e os comandos de loop são executados, conforme mostrado a seguir:



PHP

Destinado primordialmente ao desenvolvimento de scripts do lado do servidor, o **PHP (Hypertext Preprocessor)** é uma linguagem de programação open source (código aberto) muito popular na área de desenvolvimento Web, tanto para aplicações quanto para Websites. Uma das vantagens das páginas em PHP é possuir HTML com código embutido, diferentemente de linguagens como C e Perl, que exigem muitos comandos para a criação de HTML. Também é utilizado para desenvolvimento de scripts de linha de comando e aplicações GUI (interface gráfica do usuário) do lado cliente.

No exemplo de código a seguir, podemos ver como é feita a declaração de variáveis no contexto da linguagem PHP:

```
<html>
<head>
  <title>Conheça o PHP: Uma linguagem web</title>
</head>
<body>
  <p>O php trabalha de uma forma que faz com que pareça que<br />
  ele está dentro do html, mas na verdade é o html que está dentro dele!</p>
  <?php // O sinal <? ou <?php indica que iremos iniciar código em php

  // A declaração de uma variável no php é feita apenas escrevendo seu
  // nome com a opção de definir à ela um valor, de acordo com o valor
  $meuboleano = TRUE;

  // Toda variável no php é precedida pelo sinal $ e o segundo caractere
  // deve ser uma letra ou underline
  $meuinteiro = 13;

  // O tipo da variável no php é definido de acordo com o valor que é definido para ela
  $minhastring = "Olá mundo!";

  // as únicas divisões de números no php são int e float
  $meufloat = 10.01;

  // Se você declarar uma variável e não der um valor à ela, ela iniciará com o valor nulo
  $meunulo;
  $sou_meu_outro_nulo = null;

  // O php é case insensitive: ele não diferencia maiúsculas de minúsculas
  // Você pode alterar o tipo de uma variável no php se desejar
  $MeuInteiro = False;

  // echo no php, permite que você escreva no documento
  echo $minhastring.", bem vindos à Impacta!"; // Php usa ponto para concatenar
?>
</body>
</html>
```

Apêndice 1

Introdução à Lógica de Programação

15 / 22

Vejam os outros trechos de programa em PHP, no qual uma estrutura condicional é estabelecida:

```
<?php
/* A comparação de valores no php não
precisa ser feita com valores do mesmo tipo */
if (TRUE == 1) {
    echo "retorna verdadeiro!";
} else if ("Olá mundo!" == true) {
    echo "retorna verdadeiro também";
}

if (false == null) {
    echo "retorna verdadeiro também";
} else if ("" == false) {
    echo "Todos nós retornamos verdade!";
}
?>
```

Visual Basic for Applications (VBA)

Com características semelhantes à Visual Basic (VB), a **Visual Basic for Applications**, ou **VBA**, é uma linguagem de programação visual orientada a objetos implementada nos softwares que compõem o pacote Microsoft Office.

Sua utilização permite criar macros para aperfeiçoar tarefas que são realizadas frequentemente, fazendo com que várias ações sejam executadas em uma determinada sequência, por meio de um único atalho de teclado, por exemplo.

Vejam os, a seguir, exemplos de utilização da linguagem VBA nos softwares Excel e Access.

Apêndice 1

Introdução à Lógica de Programação

16 / 22

Excel

O Excel, um dos softwares de planilha eletrônica mais utilizados do mercado, permite o uso dos recursos da linguagem VBA. Vejamos um exemplo de um programa em VBA no Excel que calcula a média de quatro notas. Essas notas devem ser entre zero (0) e dez (10). Cada nota lida é verificada individualmente se está entre zero (0) e dez (10).

```
Option Explicit
Sub CalculaMedia01()

    'Versão 1, com cada nota sendo lida e verificada individualmente
    'A linhas abaixo indicam que as variáveis serão do tipo dinâmicas,
    'isto é, serão apagadas da memória do aplicativo/computador,
    'e suas informações são do tipo STRING = Texto,
    'SINGLE = Um dos subtipos do tipo Numérico

    Dim Nota1 As String
    Dim Nota2 As String
    Dim Nota3 As String
    Dim Nota4 As String
    Dim Media As Single
    Dim Msg As String
    Dim Tit As String

    Tit = "Cálculo da Média"
    'A planilha PLANILHANOTAS é selecionada
    Sheets("PlanilhaNotas").Select
    'As variáveis são carregadas com o conteúdo das células indicadas
    Nota1 = Range("A2")
    Nota2 = Range("B2")
    Nota3 = Range("C2")
    Nota4 = Range("D2")

    If Not IsNumeric(Nota1) Then
        Msg = "A rotina aceita apenas números"
    ElseIf Nota1 < 0 Or Nota1 > 10 Then
        Msg = "A rotina aceita apenas números entre 0 e 10"
    End If
    'Se o conteúdo da variável MSG for diferente de vazio,
    'isto é, se ocorreram erros, então
```



```
If Msg <> "" Then
    'O conteúdo das variáveis MSG e TIT são apresentadas em tela
    'juntamente com o endereço da célula onde o erro foi encontrado
    MsgBox Msg, , Tit & " em A2"
    'A célula A2 é selecionada para que o operador possa corrigir o erro
    Range("A2").Select
    'O processamento é terminado
    Exit Sub
End If

If Not IsNumeric(Nota2) Then
    Msg = "A rotina aceita apenas números"
ElseIf Nota2 < 0 Or Nota2 > 10 Then
    Msg = "A rotina aceita apenas números entre 0 e 10"
End If
If Msg <> "" Then
    MsgBox Msg, , Tit & " em B2"
    Range("B2").Select
    Exit Sub
End If

If Not IsNumeric(Nota3) Then
    Msg = "A rotina aceita apenas números"
ElseIf Nota3 < 0 Or Nota3 > 10 Then
    Msg = "A rotina aceita apenas números entre 0 e 10"
End If
If Msg <> "" Then
    MsgBox Msg, , Tit & " em C2"
    Range("C2").Select
    Exit Sub
End If

If Not IsNumeric(Nota4) Then
    Msg = "A rotina aceita apenas números"
ElseIf Nota4 < 0 Or Nota4 > 10 Then
    Msg = "A rotina aceita apenas números entre 0 e 10"
End If
If Msg <> "" Then
    MsgBox Msg, , Tit & " em D2"
    Range("D2").Select
    Exit Sub
End If

Media = (CSng(Nota1) + CSng(Nota2) + CSng(Nota3) + CSng(Nota4)) / 4
Msg = "A média das notas é " & Media

MsgBox Msg, , Tit
End Sub
```

Apêndice 1

Introdução à Lógica de Programação

18 / 22

A seguir, temos outro exemplo de um programa em VBA no Excel, em que cada nota lida tem sua verificação efetuada através de um loop:

```
Sub CalculaMedia02()  
  
    'Versão 2, com matriz e cada nota sendo lida e verificada dentro de um loop  
  
    Dim i As Byte  
    Dim Msg As String  
    Dim Tit As String  
    Dim Media As Single  
    Dim MediaFinal As Single  
    Dim Nota(3) As String  
    Tit = "Cálculo da Média "  
  
    'Conceitos  
    'Neste exemplo, se não houver erros de preenchimento das células, a rotina faz o cálculo da média.  
    'Ocorrendo um dos erros previstos, o operador é alertado sobre a natureza do erro, o local onde  
    'ele ocorreu é selecionado e a rotina é interrompida  
  
    For i = 0 To 3  
        Nota(i) = Cells(2, i + 1)  
        If Not IsNumeric(Nota(i)) Then  
            Msg = "A rotina aceita apenas números"  
        ElseIf Nota(i) < 0 Or Nota(i) > 10 Then  
            Msg = "A rotina aceita apenas números entre 0 e 10"  
        End If  
        'Se a variável MSG tiver um conteúdo (uma das mensagens de erro) então  
        If Msg <> "" Then  
            'O texto composto pelo conteúdo das variáveis MSG, TIT e pelas frases indicadas abaixo  
            'é apresentado ao operador  
            MsgBox Msg & vbCrLf & "Faça a correção e execute novamente a rotina", ,  
Tit & " Erro em Nota " & (i) + 1  
            'A célula onde o erro foi identificado é selecionada  
            Cells(2, i + 1).Select  
            'O processamento é encerrado  
            Exit Sub  
        End If  
    Next
```

Apêndice 1

Introdução à Lógica de Programação

19 / 22

```
`Não foram encontrados erros, a natureza das variáveis é convertida, a média é calculada
MediaFinal = (CSng(Nota(0)) + CSng(Nota(1)) + CSng(Nota(2)) + CSng(Nota(3))) / 4
`A variável é carregada com a frase indicada e com o resultado do cálculo
Msg = "A média das notas é " & MediaFinal
`O conteúdo da variável é apresentado ao operador
MsgBox Msg, , Tit
End Sub
```

Access

O Access, da Microsoft, é um software que permite construir, aplicar e distribuir banco de dados. É possível realizar muitas tarefas utilizando a interface do usuário, mas em muitos casos é necessária a programação. O Access também permite o uso dos recursos do VBA.

Vejamos um exemplo de um programa em VBA no Access. Ele calcula a média de quatro notas. Essas notas devem ser entre zero (0) e dez (10). Cada nota lida é verificada individualmente se está entre zero (0) e dez (10):

```
Sub CalculaMedia01()

`Versão 1, com cada nota sendo lida e verificada individualmente
`A linhas abaixo indicam que as variáveis serão do tipo dinâmicas,
`isto é, serão apagadas da memória do aplicativo/computador,
`e suas informações são do tipo STRING = Texto,
`SINGLE = Um dos subtipos do tipo Numérico

`Declaração das variáveis
Dim Nota1 As String
Dim Nota2 As String
Dim Nota3 As String
Dim Nota4 As String
Dim Media As Single
Dim Msg As String
Dim Tit As String
```

```
`Atribuição de valores ou carregamento
Tit = "Cálculo da Média"

`Conceitos
`A estrutura de decisão IF...ELSEIF...ELSE...ENDIF permite verificar os vári-
os
`tipos de erros possíveis. O processamento sairá do laço quando
`não houver erros (ELSE).
`O laço Do...Loop é necessário caso os erros de digitação sejam repetitivos,
`isto é, enquanto houver erros no preenchimento das variáveis o processamento
`não sai do laço

Do While True
    `A variável NOTA1 é preenchida por digitação do operador, na caixa de en-
trada
    Nota1 = InputBox("Digite a Nota1", Tit)
    `O conteúdo digitado é verificado em relação a sua natureza,
    `se não for de natureza numérico, então
    If Not IsNumeric(Nota1) Then
        `A variável MSG é carregada com o texto indicado
        Msg = "Digite apenas números"
    `O conteúdo digitado é verificado em relação ao seu valor,
    `se for menor do que zero ou maior do que 10, então
    ElseIf Nota1 < 0 Or Nota1 > 10 Then
        `A variável MSG é carregada com o texto indicado
        Msg = "Digite números entre 0 e 10"
    `O conteúdo digitado não é como indicado nas alternativas anteriores
    Else
        `O processamento sai do laço e vai para a linha seguinte ao LOOP
        Exit Do
    End If
    `O conteúdo da variável MSG e da variável TIT são mostrados em tela
    MsgBox Msg, , Tit
Loop

Do While True
    Nota2 = InputBox("Digite a Nota2", Tit)
    If Not IsNumeric(Nota1) Then
        Msg = "Digite apenas números"
    ElseIf Nota2 < 0 Or Nota2 > 10 Then
        Msg = "Digite números entre 0 e 10"
    Else
        Exit Do
    End If
```

```
    MsgBox Msg, , Tit
Loop

Do While True
    Nota3 = InputBox("Digite a Nota3", Tit)
    If Not IsNumeric(Nota1) Then
        Msg = "Digite apenas números"
    ElseIf Nota3 < 0 Or Nota3 > 10 Then
        Msg = "Digite números entre 0 e 10"
    Else
        Exit Do
    End If
    MsgBox Msg, , Tit
Loop

Do While True
    Nota4 = InputBox("Digite a Nota4", Tit)
    If Not IsNumeric(Nota4) Then
        Msg = "Digite apenas números"
    ElseIf Nota4 < 0 Or Nota4 > 10 Then
        Msg = "Digite números entre 0 e 10"
    Else
        Exit Do
    End If
    MsgBox Msg, , Tit
Loop

'A média aritmética (variável MEDIA) é calculada após a conversão
'(CSNG = converter o conteúdo da variável para a natureza numérica, sub tipo
single)
'do conteúdo da variável (Nota1, Nota2, Nota3, Nota4) para natureza número
Media = (CSng(Nota1) + CSng(Nota2) + CSng(Nota3) + CSng(Nota4)) / 4
'A variável MSG é carregada com a frase indicada
Msg = "A média das notas é " & Media
'O conteúdo da variável MSG e da variável TIT são mostrados em tela
MsgBox Msg, , Tit
End Sub
```

Apêndice 1

Introdução à Lógica de Programação

22 / 22

A seguir, temos outro exemplo de um programa em VBA no Access, em que cada nota lida tem sua verificação efetuada através de um loop:

```
Sub CalculaMedia02()  
  
    'Versão 2, com matriz e cada nota sendo lida e verificada dentro de um loop  
    'A linha DIM NOTA(3) AS STRING indica a declaração de uma matriz de uma dimen-  
    são  
    'com 4 elementos numerados de zero a 3 e de natureza texto (como o nome das  
    variáveis)  
  
    Dim i As Byte  
    Dim Msg As String  
    Dim Tit As String  
    Dim Media As Single  
    Dim Nota(3) As String  
    Tit = "Cálculo da Média"  
    'Laço do tipo FOR...NEXT.  
    'Lê-se: Para i variando de 0 até 3, com incremento de 1  
    'O processamento é automaticamente levado para fora do laço  
    'quando a contagem de i for maior do que 3  
    For i = 0 To 3 Step 1  
        'Laço do tipo teste de condição, isto é, faça enquanto a condição for ver-  
        dadeira  
        'Neste caso o processamento é levado para fora do laço pelo comando EXIT  
        DO  
            Do While True  
                Nota(i) = InputBox("Digite a Nota " & i + 1, Tit)  
                If Not IsNumeric(Nota(i)) Then  
                    Msg = "Digite apenas números"  
                ElseIf Nota(i) < 0 Or Nota(i) > 10 Then  
                    Msg = "Digite números entre 0 e 10"  
                Else  
                    Exit Do  
                End If  
                MsgBox Msg, , Tit  
            Loop  
            'Quando i for igual a 3, o laço FOR...NEXT o processamento automaticamente  
            'passará para a linha de comando após a clausula NEXT  
        Next  
        MediaFinal = (CSng(Nota(0)) + CSng(Nota(1)) + CSng(Nota(2)) +  
        CSng(Nota(3))) / 4  
        Msg = "A média das notas é " & MediaFinal  
        MsgBox Msg, , Tit  
    End Sub
```