INTRODUÇÃO À PROGRAMAÇÃO DO ACCESS BASIC

Access Basic é a linguagem de Programação para o Microsoft Access. Mais potente do que os objetos de Macro. O Access Basis foi projetado para controlar e estender o Access, e é muito parecida com a maioria das linguagens de programação estruturadas conhecidas. Você pode usá-la para:

- escrever funções personalizadas que podem ser usadas em expressões, em macros e etc..,
- automatizar a manipulação de objetos e de dados em seu banco de dados,

- construir aplicativos de banco de dados sofisticados.

Você escreve o código do Access Basic em unidades denominadas procedimentos que são armazenados nos objetos **Módulos**. Um procedimento contém uma série de instruções do Access Basic que desenvolvem uma operação ou calculam um valor. Embora você possa guardar todos os procedimentos em um único Módulo, pode ser mais conveniente dividi-los em grupos lógicos e guardar cada grupo em um Módulo separado. Há duas espécies de procedimentos:

- procedimentos **Function** que recebem zero ou mais argumentos e devolvem um valor **que podem** ser usados em uma expressão;

- procedimentos **Sub** que recebem zero ou mais argumentos e **não podem** ser usados em uma expressão.

Cada Módulo tem uma simples seção Declarações e zero ou mais procedimentos. A seção Declarações contém, como padrão, a instrução **Option Compare Database.**

Fundamentos do Access Basic

Declarando Variáveis

Deve começar com uma letra,

Deve conter apenas letras e números. Caracteres de pontuaçãoe e espaços não são permitidos,

Não deve ter mais do que 40 caracteres,

Não pode ser uma palavra reservada.

Você declara uma variável com a instrução **Dim:**

Dim nomedavariável

exemplo: **Dim Resultado**, cria uma variável denominada Resultado.

Declaração Implícita

Você não precisa declarar uma variável antes de usá-la. O Access Basic cria implicitamente uma variável com esse nome, e você pode usá-la como se a tivesse declarado explicitamente.

exemplo: Function SalvaRaiz(num)

ValTemp = ABS (num)
SalvaRaiz = Sqr(ValTemp)

End Function.

A variável **ValTemp** não foi declarada, e a função é executada normalmente.

Escopo e tempo de vida de variáveis

Declaração	**Escopo**
Dim (usado no interior de um procedimento)	Local
Dim (na seção declarações de um Módulo)	Módulo
Global (na seção declarações de um Módulo)	Global

Variáveis estáticas

Além de escopo as variáveis possuem tempo de vida. Os valores em variáveis de módulos e varáveis globais são preservados enquanto o banco de dados estiver aberto. Mas as variáveis locais só existem enquanto o procedimento no qual foram declaradas estiver sendo executado. Entretanto, você pode fazer com que o valor de uma variável local seja preservado tornando-a uma **variável estática.**

Static (nomeda variável)

exemplo: Function Total(num)

```
Static Acumulador
Acumulador = Acumulador + num
Total = Acumulador

End Function.
```

Declarando todas as variáveis locais como estáticas

Basta colocar a palavra reservada **Static** no começo de um procedimento.

Tipos de dados de variáveis fundamentais.

Ao declarar uma variável pode-se fornecer o seu tipo de dados associado. Mas, por padrão, quando não se fornece o tipo de dados associado à variável, o tipo de dados asoociado é o **Variant.**

Para se verificar o tipo de dado contido em variáveis **Variant**, pode-se usar algumas das funções:

- IsNumeric

- IsDate

- IsEmpty

- IsNull.

Outros tipos de dados fundamentais

Tipos de dados de argumento

Os argumentos para os procedimentos que você escreve têm o tipo de dados Variant por padrão. Contudo, você pode declarar outros tipos de dados para argumentos com a palavra reservada **By Val,** conforme o exemplo:

exemplo: By Val N as Integer

Tipos de dados de função

Por retornarem valores, as funções, como as variáveis, têm um tipo de dados. Também a exemplo das variáveis, as funções têm por padrão um tipo **Variant** . Mas, no entanto, uma função pode ser declarada para retornar um tipo especial.

exemplo: Function Reverso (S As String, ByVal n As Integer) As String

Matrizes

Há três maneiras de declarar uma Matriz ordinária (de tamanho fixo), dependendo do escopo desejado para a Matriz:

- Para criar uma Matriz Global, use a instrução GLOBAL

- Para criar uma Matriz de nível de Módulo, use a instrução Dim

- Para criar uma Matriz no interior de um procedimento, use a instrução STATIC.

exemplo: Dim Contadores (14) As Integer , (unidimensional)

Global Contadores(14) As Integer, (unidiimensional)

Static Contadores (14) As Integer, (unidimensional)

Static Matriz (9,9) As Double, (bidimensional)

O limite inferior padrão das matrizes é 0 (zero), logo, Contadores possui 15 elementos. Pode-se alterar o liimite padrão para 1 (um) com a instrução **Option Base** na declaração do Módulo.

exemplo: Option Base 1 ou

Dim Contadores (1 to 14) As Integer

Dim Matriz (1 to 10, 1 to 20) As Integer (bidimensional)

exemplo de operação com Matrizes:

```
Statiic Contadores (1 to 15) As  Integer
Dim  I  As  Integer

For  I  = 1  to  15
      Contador (I)  =  5
Next
```

As regras se alteram quando você cria uma **Matriz dinâmica** .

exemplo: Dim MatrizDinâmica ( )

E para se alocar um numero de elementos utiiliiza-se a instrução **ReDim.**

exemplo:

Sub CalcValoresAgora ( )

ReDim Matrz (19, 29), a instrução ReDim aloca uma

Matriz 20 x 30 (600 elementos)