> **VBA基礎文法最速マスター**
> VBA

VBAの文法一覧です。他の言語をある程度知っている人はこれを読めばVBAの基礎をマスターしてVBAを書くことができるようになっています。簡易リファレンスとしても利用できると思いますので、これは足りないと思うものがあれば教えてください。

1. 基礎

Visual Basic Editorの起動

VBAはVisual Basic Editorで編集・実行します。Visual Basic Editorは次のように起動します。

```
ExcelやWordのメニューで[ツール]-[マクロ]-[Visual Basic Editor]を選択する
```

標準モジュールの追加

VBAは標準モジュールに記述します。標準モジュールは次のように追加します。

```
Visual Basic Editorのメニューで[挿入]-[標準モジュール]を選択する。
```

変数宣言の強制

変数宣言を強制するため、モジュールの先頭に必ず以下の行を書くようにします。プログラムが動けばよいからという考えでこの行を書かないというのはよくないです。この行を加えることで変数宣言が強制されて変数名の誤入力を防ぐことができるため、コードの品質が上がります。

```
Option Explicit
```

関数の作成

プログラムは関数の先頭から実行されます。関数は次のように書きます。

```
Sub Main()

    ' 処理を書く

End Sub
```

MsgBox関数

メッセージボックスを表示する関数です。

```
MsgBox "Hello world"
```

Debug.Printメソッド

イミディエイト ウィンドウに文字列を出力する関数です。Debug.Printメソッドは名前のとおりデバッグ時に役に立ちます。

```
Debug.Print "Hello world"
```

コメント

コメントです。

```
'コメント
```

変数の宣言

変数の宣言です。VBAには通常の変数と、配列変数があります。変数の宣言時にはデータ型を指定します。

```
' 変数
Dim num As Integer

'配列変数
Dim students() As String
```

データ型

データ型です。変数の宣言時に以下のデータ型を指定できます。

```
' Boolean(ブール)型
Dim flag As Boolean
' Inteegr(整数)型
Dim num As Integer
' Long(長整数)型
Dim row As Long
' Single(単精度浮動少数点)型
Dim value As Single
' Double(倍精度浮動少数点)型
Dim value As Double
' Date(日付)型
Dim today As Date
' String(文字列)型
Dim str as String
```

プログラムの実行

プログラムを実行するには、Visual Basic Editorで以下のいずれかを行います。

```
(1) メニューバーで[実行]-[マクロの実行]を選択する。
(3) ツールバーの[マクロの実行]ボタンを押す。
(2) F5ボタンを押す。
```

コンパイルチェック

事前にコンパイルチェックを行うには次のようにします。

```
Visual Basic Editorのメニューバーで[デバッグ]-[VBAProjectのコンパイル]を選択しま
```

2. 数値

数値の表現

Integer、Long、Single、Double型の変数に数値を代入できます。Integer、Long型には整数だけ代入できます。Single、Double型には整数でも小数でも代入できます。

```
Dim num as Integer
num = 1

Dim num As Long
num = 100000000

Dim num as Single
num = 1.234

Dim num as Double
num = 1.2345678
```

四則演算

四則演算です。

```
num = 1 + 1
num = 1 - 1
num = 1 * 2
num = 1 / 2
```

余りと商の求め方です。

```
' 商
div = 3 ¥ 2

' 余り
m = 3 mod 2
```

インクリメントとデクリメント

インクリメントとデクリメントです。

```
' インクリメント
i = i + 1

' デクリメント
i = i - 1
```

3. 文字列

文字列の表現

文字列はダブルクォートで囲みます。

```
Dim str As String
str = "abc"
```

文字列操作

各種文字列操作です。

```
' 結合
Dim join1 As String
join1 = "aaa" & "bbb"

Dim ary(2) As String
Dim join2 As String
ary(0) = "aaa"
ary(1) = "bbb"
ary(2) = "ccc"
join2 = Join(ary, ",")

' 分割
Dim record() As String
record = Split("aaa,bbb,ccc", ",")

' 長さ
Dim length As Integer
length = Len("abcdef")

' 切り出し
Dim left_str As String
left_str = Left("abcd", 3)   ' abc

DIm mid_str As String
mid_str = Mid("abcd", 1, 2)  ' ab

Dim right_str
right_str = Right("abcd", 1) ' c

' 検索
Dim result As Integer
result = InStr("abcd", "cd") '見つかった場合はその位置、見つからなかった場合は0
```

4. 配列

配列変数の宣言

配列です。配列の宣言時に要素数を指定すると静的配列になり、省略すると動的配列になります。静的配列は一度宣言すると要素数を変更できません。動的配列は宣言した後で要素数を変更できます。

```
' 静的配列の宣言
Dim ary(3) As String

' 動的配列の宣言
Dim ary() As String
```

配列の要素の参照と代入

配列の要素を参照と代入です。

```
' 要素の参照
ary(0)
ary(1)

' 要素の代入
ary(0) = 1
ary(1) = 2
```

配列の要素番号の最大値

配列の要素番号の最大値を取得するには以下のようにします。

```
ary_max = UBound(ary)
```

動的配列の要素数の変更

動的配列の要素数を変更するには以下のようにします。

```
'要素数の変更
Dim ary() as String
Redim Preserve ary(3)
Redim Preserve ary(5)
```

5. ユーザ定義型

ユーザ定義型の宣言

ユーザ定義型です。ユーザ定義型を宣言すると、複数の要素を持つ変数を作成できます。ユーザ定義型を宣言するには以下のようにします。

```
Type Person
  name As String
  age As Integer
End Type
```

ユーザ定義型の変数の宣言

ユーザ定義型の変数の宣言です。ユーザ定義形の変数の宣言は以下のようにします。

```
Dim person1 As Person
```

ユーザ定義型の参照と代入

ユーザ定義型の参照と代入です。

```
' 要素の参照
person.name
person.age

' 要素の代入
person.name = "bob"
person.age = 35
```

6. 制御文

If文

If文です。

```
If 条件 Then

End If
```

If ～ Else文

If ～ Else文です。

```
If 条件 Then

Else

End If
```

if 〜 Elseif 文

If 〜 Elseif文です。End Ifと異なり、ElseとIfの間には空白を入れないことに注意しましょう。

```
If 条件 Then

ElseIf 条件 Then

End If
```

Do While文

Do While文です。

```
i = 0
Do while i < 5

    ' 処理

    i = i + 1
Loop
```

For文

For文です。

```
For i = 0 To 4

Next
```

7. Subプロシージャ

VBAでは戻り値を返さない関数をSubプロシージャといいます。Subプロシージャを作るには次のようにします。

```
Sub print_sum(num1 As Integer, num2 As Integer)
    Dim total As Integer

    total = num1 + num2

    MsgBox total
End Sub
```

8. Functionプロシージャ

VBAでは戻り値を返す関数をFunctionプロシージャといいます。Functionプロシージャを作るには次のようにします。戻り値を返却するには、Functionプロシージャ名に戻り値を代入します。

VBAでは戻り値として配列を返すことはできません。

```
Function sum(num1 As Integer, num2 As Integer) As Integer
    Dim total As Integer

    total = num1 + num2

    '戻り値
    sum = total
End Sub
```

9. ファイル入出力

ファイル入出力です。以下が雛形になります。ファイルをオープンすると、fileno内の番号がファイルに割り当てられます。For Inputは読み込みモードです。書き込む場合はFor Outputとします。

```
Dim fileno As Integer

fileno = FreeFile
Open filename For Input As fileno

Do While Not Eof(fileno)
    Line Input fileno, line

Loop

Close fileno
```

知っておいたほうがよい文法

VBAでよく出てくる知っておいたほうがよい文法の一覧です。

Do Until文

Do Until文はDo While文の否定を表現します。

```
Do Until 条件

Loop
```

関数の途中で抜ける

Subプロシージャの途中で抜けるにはExit Subステートメントを使用します。

```
Sub proc()

  If 条件 Then
    Exit Sub       '条件を満たす場合、Subプロシージャを抜ける。
  End If

End Sub
```

Functionプロシージャの途中で抜けるにはExit Functionステートメントを使用します。

```
Function proc() As String

  If 条件 Then
    proc = "exit"
    Exit Function      '条件を満たす場合、Functionプロシージャを抜ける。
  End If

 proc = "end"

End Function
```

繰り返し文の途中で抜ける

For文の途中で抜けるにはExit Forステートメントを使用します。

```
For i = 0 To 4

    If 条件 Then
        Exit For    ' 条件を満たす場合、For文を抜ける。
    End If

Loop
```

Do While文やDo Until文の途中で抜けるにはExit Doステートメントを使用します。

```
i = 0
Do While i < 5

    If 条件 Then
        Exit Do    ' 条件を満たす場合、Do While文を抜ける。
    End If

Loop
```

エラー処理

実行時エラーを投げるにはErr.Raiseを使用します。一つ目の引数にはエラー番号を513～65535の範囲で指定します。三つ目の引数にはエラーメッセージを指定します。

```
Err.Raise 513, , "Error message"
```

実行時エラーの発生時にエラー処理をするにはOn Errorステートメントを使用します。

```
On Error GoTo ERR_PROC

  '実行時エラーが発生する可能性のある処理

ERR_PROC:

  'エラー時の処理
```

関数の引数で値を返却

Subプロシージャと Functionプロシージャの引数は、値を返却するためにも使用できます。

```
Sub main()

    Dim out_value as Integer

    sql 4, out_value    ' Subプロシージャの処理結果が引数out_valueに格納される。

END Sub

Sub sqr(in_value As Integer, out_value As Integer)

    out_value = in_value * in_value

End Sub
```

関数の引数で配列を返すこともできます。

```
Sub main()

    Dim out_ary() as Integer

    get_ary out_ary    ' Subプロシージャの処理結果が配列引数out_aryに格納される。

END Sub

Sub get_ary(out_ary() As Integer)

    Dim i As Integer
    For i = 0 To UBound
        out_ary(i) = i * i
    Next

End Sub
```

VBA参考資料

- VBAで覚えておくデータ構造 - 静的配列・動的配列・ディクショナリ
- VBAを使うなら理解しておきたいアルゴリズム - 抽出・結合・集計

基礎文法最速マスターリンク集

この記事は最近の基礎文法最速マスターの流れに便乗して作成したものです。

以下、各種基礎文法最速マスターへのリンクです。

- Perl基礎文法最速マスター - Perl入門～サンプルコードによるPerl入門～
- Route 477 - Ruby基礎文法最速マスター
- PHP基礎文法最速マスター | Shin x blog
- Python基礎文法最速マスター - D++のはまり日誌
- Java基礎文法最速マスター - 何かしらの言語による記述を解析する日記
- VBA基礎文法最速マスター - 何かしらの言語による記述を解析する日記
- Brainf*ck基礎文法最速マスター - 医者を志す妻を応援する夫の日記
- Haskell基礎文法最速マスター - think and error
- Bash基礎文法最速マスター - 何かしらの言語による記述を解析する日記
- VBScript 基礎文法最速マスター - CX's VBScript Diary - VBScript グループ
- JavaScript基礎文法最速マスター - なんとなく日記