3. VBA の基本

§3.1　VBA の構成

VBA(Visual Basic for Applications) は VBE (Visual Basic Editor) を通して編集される。VBE の操作法をマスターしながら、VBA の構成を学ぶ。

Check Point	1. VBE の構成を理解する。 2. プログラム作製の全体の流れをつかむ。

3.1.1　プロジェクトとモジュール

① VBE の起動

まず、VBE を起動する（図 3.1.1）。『ツール』のプルダウンメニューの『マクロ(M)』の『フィルタVisual Basic Editor (V)』を選択する。

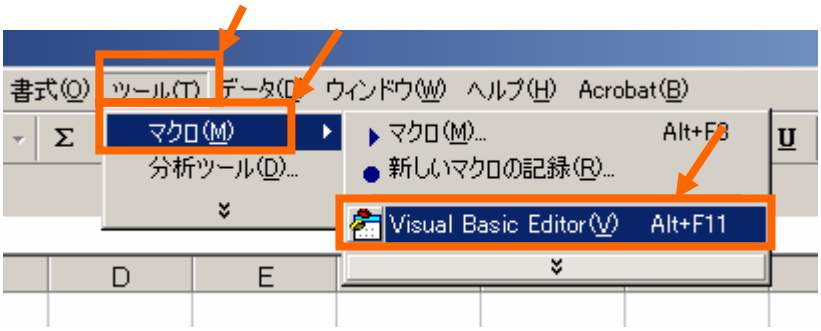


図 3.1.1

VBE が立ち上がる（図 3.1.2）。

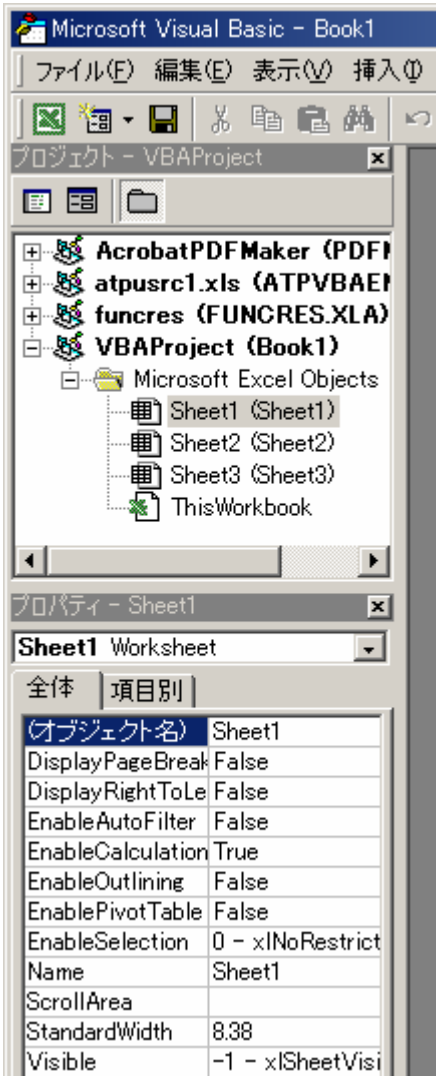


図 3.1.2

② プロジェクト

『**プロジェクト**』とは『**モジュール**』の集まりのこと。Excelで例えるなら『**ワークシート**』や『**グラフシート**』の集まりを『**ブック**』と呼ぶのに似ている。

プロジェクトは、ファイルとして存在するが、あくまでもモジュールを<u>まとめる（管理する）概念にすぎない</u>。実際には、図 3.1.3 のようにWindowsのファイル管理ツール『エクスプローラー』と同じような階層表示となっている。

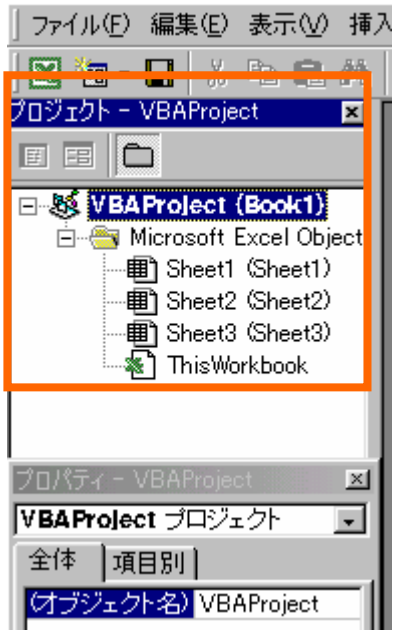


図 3.1.3

③ モジュール

マクロを記録すると VBA の記述が『**Macro1**』『**Macro2**』…に記述される。これが『**標準モジュール**』である。その他にも『**ユーザーフォーム**』、『**シート**』などもモジュールとして扱われる（図 3.1.4）。

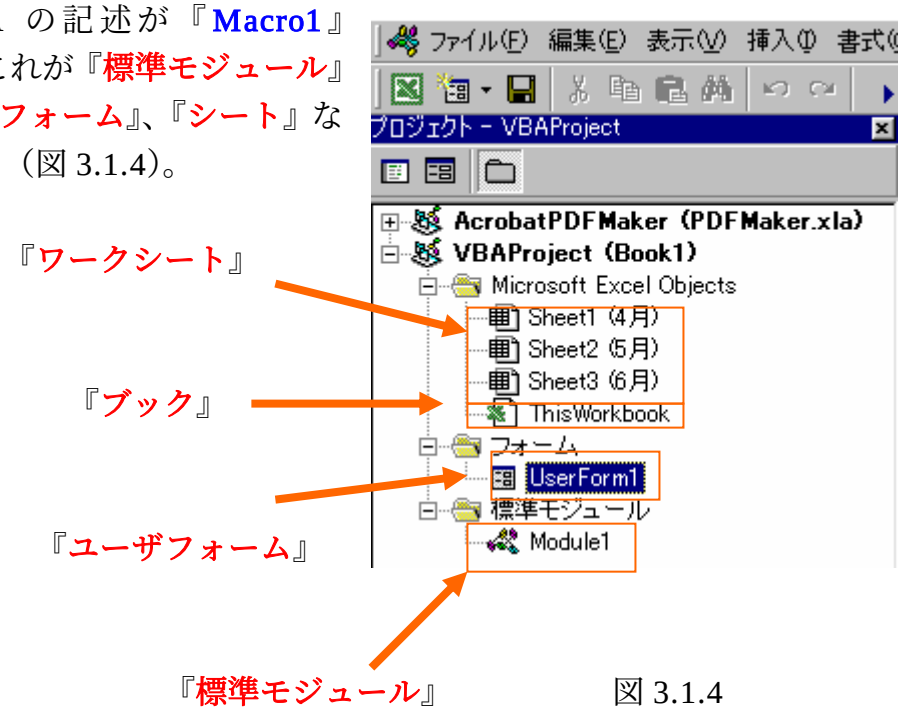


図 3.1.4

3.1.2　プロシージャ

マクロは、1 つまたは複数の『**プロシージャ**』で記述されている。また、プロシージャはプログラム言語の文法に従って記述された『**ステートメント**』と呼ばれる命令文、そのステートメントの集まりの『**コード**』によって構成される。

① Sub プロシージャ

『**Sub**』で始まって『**End Sub**』で終わる。この間に記述されたコードを実行する。

Subプロシージャは**戻り値を持たない**ため、処理した結果を呼び出されたルーチンに返すことができない。

```
Sub　プロシージャ名（引数　As　データ型、引数　As　データ型…）
　　…
　　…
End　Sub
```

② Function プロシージャ

『**Function**』で始まって『**End Function**』で終わる。

処理の結果を**戻り値**として呼び出し元に返す。ワークシート関数にない機能をユーザー定義関数として実現する場合に用いる。

```
Function　プロシージャ名（引数　As　データ型、引数　As　データ型…）
　　…
　　…
End　Function
```

③ プロシージャの有効範囲

Private

特定のモジュール内でしか呼び出すことができない。Sub または Function プロシージャの宣言の前に『Private』を付ける。

Public

プロジェクト内のすべてのモジュールから呼び出すことができる。Sub または Function プロシージャの宣言の前に『Public』を付ける。なお、Sub または Function プロシージャの宣言に指定しない場合、Public プロシージャとなる。

> プロシージャは、４章で詳しく説明する。

3.1.3　オブジェクトとプロパティ

①　オブジェクト

『オブジェクト』とは扱う **対象物** のことを意味する。Excel では『**ブック**』『**ワークシート**』など、実際に使用するものがオブジェクトとなる。VBA では、『**フォーム**』（図 3.1.5）『ボタン』『チェックボタン』『コンボボックス』などといったコントロール類が代表的なオブジェクトとなる。

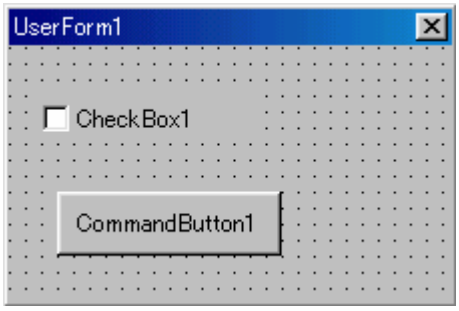


図 3.1.5

②　プロパティ

オブジェクトの性質や特徴を表すものが『**プロパティ**』である。図 3.1.6 のようにプロジェクトエクスプローラの下に表示されているのが『**プロパティウィンドウ**』である。このプロパティウィンドウを用いて現在選択されているオブジェクトをカスタマイズすることができる。

図 3.1.6

§3.2　VBA の操作法

実際に VBE (Visual Basic Editor) を使って、VBA を使ってみる。最初は理解できないことやうまくいかないことが多いが、とにかくやってみる！

Check Point	1. VBA の基本操作を学ぶ。 2. マクロと VBA の違いを理解する。

3.2.1　マクロからの起動

前節では、VBE を直接起動したが、ここではマクロからプロシージャを編集してみる。

① マクロの起動

『ツール(T)』のプルダウンメニューの『マクロ(M)』の『マクロ(M)…』を選択する（図 3.2.1）。

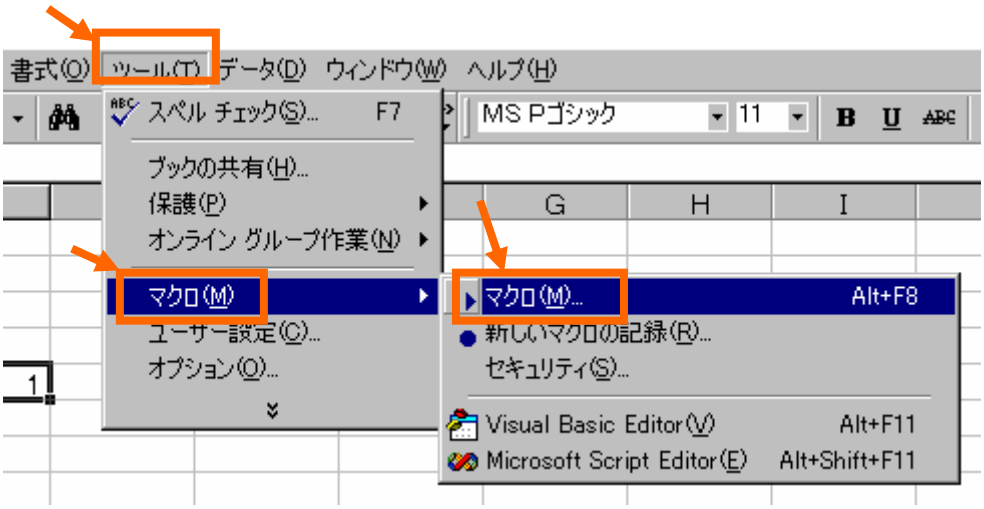


図 3.2.1

② マクロ名の設定

図 3.2.2 のウィザードが立ち上がるので、『マクロ名(M)』（プログラムの名前）の適当な名前を入力する。

その後『作成(C)』クリックする。マクロ名に日本語は使わないこと！必ず『英数半角』を用いなければならない。

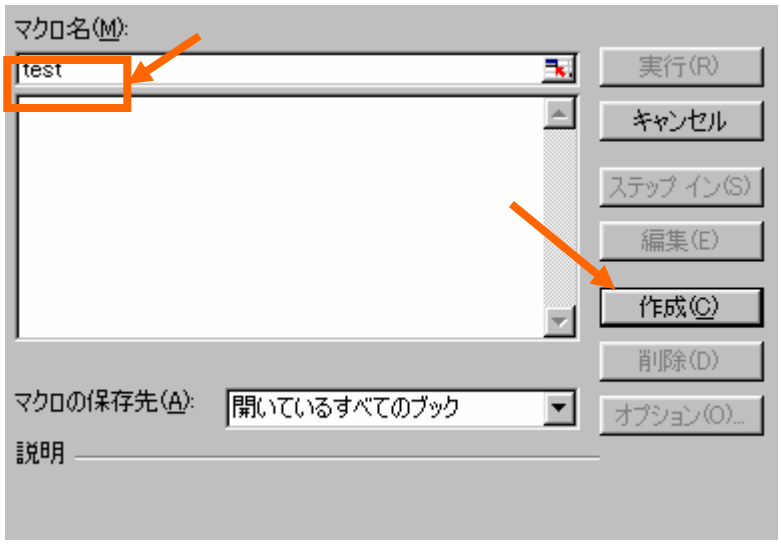


図 3.2.2

③　保存

まずプロシージャを保存する。

ボタンをクリックして Excel のブック（*.xls）として保存する（図 3.2.3）。

プログラムは Excel のファイルの中にマクロとして保存されている。

例えば、右図の場合、『book1.xls』のファイルの中に **test** という名前のマクロとして保存されている。

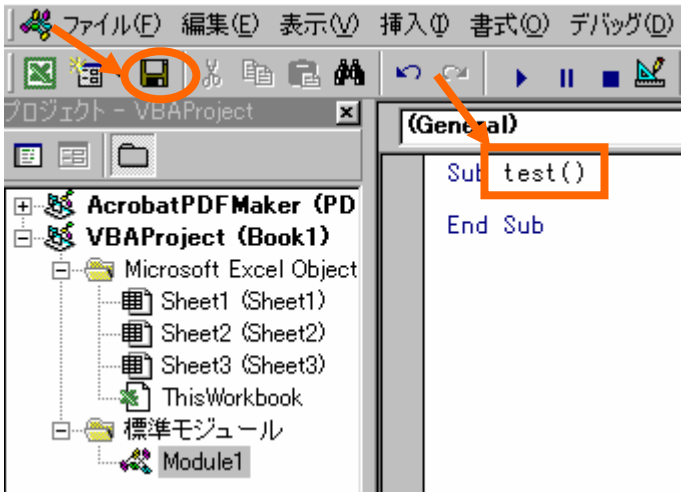


図 3.2.3

『**Sub**』はプログラムの始まりを表し、『**End Sub**』はプログラムの終わりです。『**Sub**』と『**End Sub**』で囲まれた範囲がひとつのプログラムである。

```
Sub プログラム名()


End Sub
```

3.2.2　マクロの有効

2 章で説明したが、Excel を終了した後に、このファイルを開くと、図 3.2.4 のようなウィザードが現れる。

『**マクロを有効にする(E)**』を、必ず、クリックする。そうしないと test というマクロ（プログラム）を認識しないのでVBAのエディタで開けない。

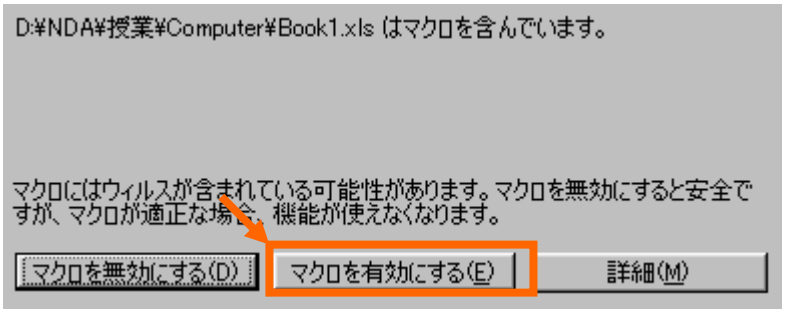


図 3.2.4

電子メールの添付ファイルに注意！

最近のウィルスは、Excelなどのマクロを使って自動的にコンピュータに感染します。知らない人からの添付ファイルは開かないこと。もし、ファイルを開いて上のようなウィザードが現れたら、『**マクロを無効にする(D)**』をクリックするか「**Ctrl**」+「**Alt**」+「**Delete**」でプログラムを強制終了させること。

重要！

＊セキュリティについての補足

『ツール(T)』のプルダウンメニューの『マクロ(M)』の『セキュリティ(S)…』を選択する（図 3.2.5）。

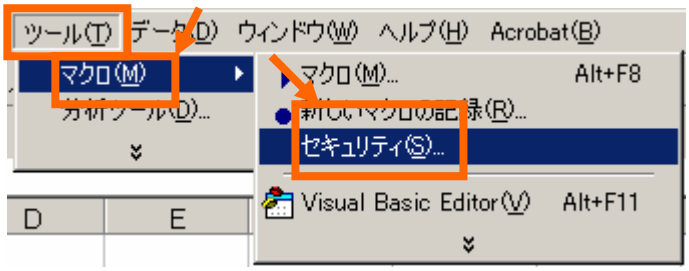


図 3.2.5

図 3.2.6 のようなウィザードが立ち上がる。『セキュリティ　レベル(S)』は必ず『高』か『中』でなければならない。

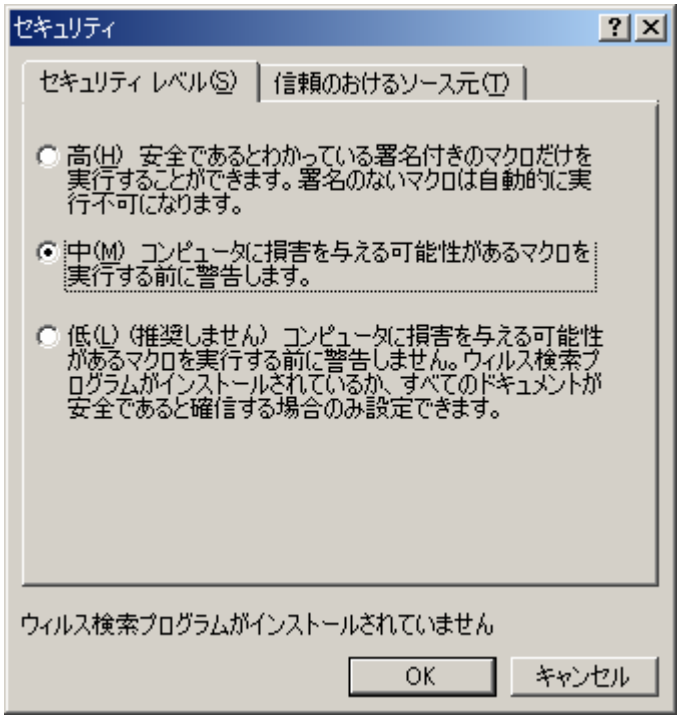


図 3.2.6

コンピュータウィルスはマクロを実行してウィルスを感染させる。

マクロで簡単にプログラムができることの短所である。

他の人のマクロを使う場合には最新の注意が必要になる。

『署名つきのマクロ』を自分で勉強すること！（時間がないのでここでは説明しない）

3.2.3　簡単なプログラム

①　数値表示

図 3.2.7 は数値を表示させるプログラムである。『 Cells(i, j) 』は、『 i 行 』『 j 列 』のセルという意味である。例えば、『 i = 5 』, 『 j =3 』の場合は、C5 セルになる（図 3.2.8）。

プログラム上の『＝』は代入なので、『 Cells(1, 1) = 1 』は、A1 セルに数値『 1 』を代入しなさいというコマンドである。

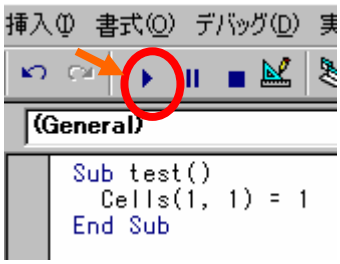


図 3.2.7

▶ をクリックするとプログラムを実行する。

実行する前に必ず保存すること。

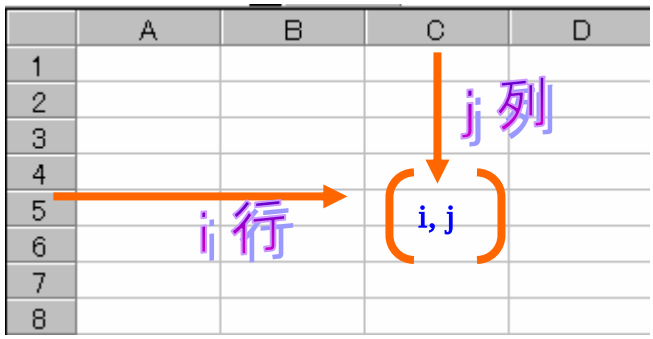


図 3.2.8

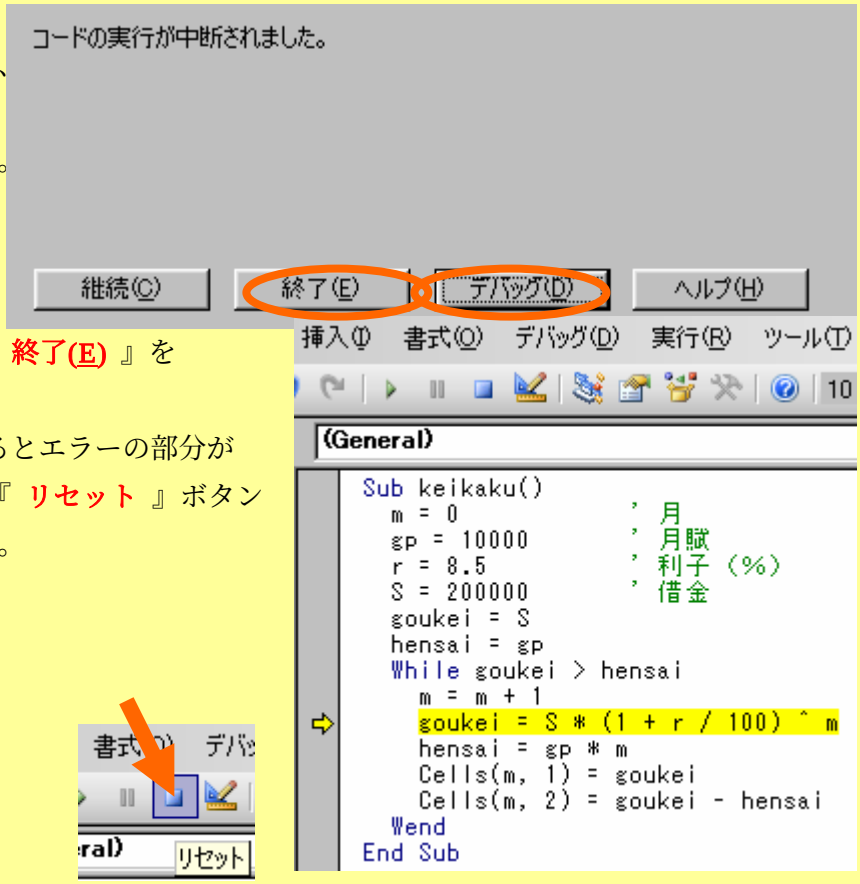
コマンドの先頭の文字が自動的に大文字にならない場合は、スペルが間違っている。

プログラムのエラーは、ほとんどがコマンドの『スペルの間違い』や、『全角』を用いてコマンドを書いているなどの簡単なミスの場合が多い。

プログラムが暴走した場合は、キーボードの 『Esc』 を押してプログラムを強制終了すること。

右図のウィザードが現れるので『 終了(E) 』をクリックする。

『 デバッグ(D) 』をクリックするとエラーの部分が黄色いラインで示される。その後、『 リセット 』ボタンをクリックして黄色いラインを消す。



② 他のマクロの追加

既存のマクロの下に、『 namae 』というマクロを加える。

『sub namae()』 と入力した後、キーボードの『Enter』ボタンを押す（図 3.2.9）。

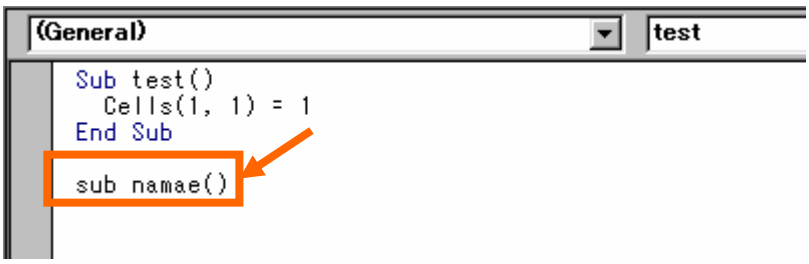


図 3.2.9

自動的に『 sub 』の先頭が大文字なり、青色に変わる『 Sub 』。また、『End Sub』も自動的に付け足される（図 3.2.10）。

カーソルが 『 namae 』のプログラムにあるときは、『 namae 』と表示される。この状態で実行するとこの表示 (namae) のマクロを実行する。

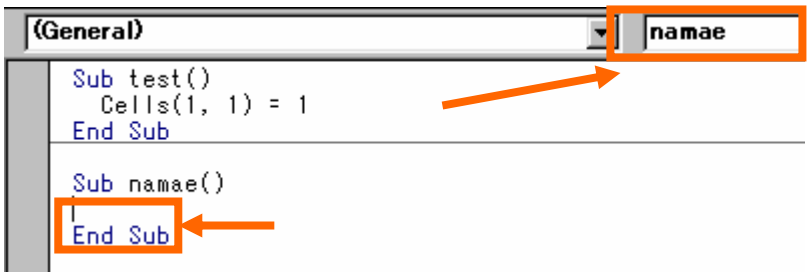


図 3.2.10

③ 複数のマクロの選択

プルダウンメニューで マクロ名 を選択する（図 3.2.11）。もしくは実行したいマクロの『Sub マクロ名()』と『End Sub』の間にカーソルを持っていく。

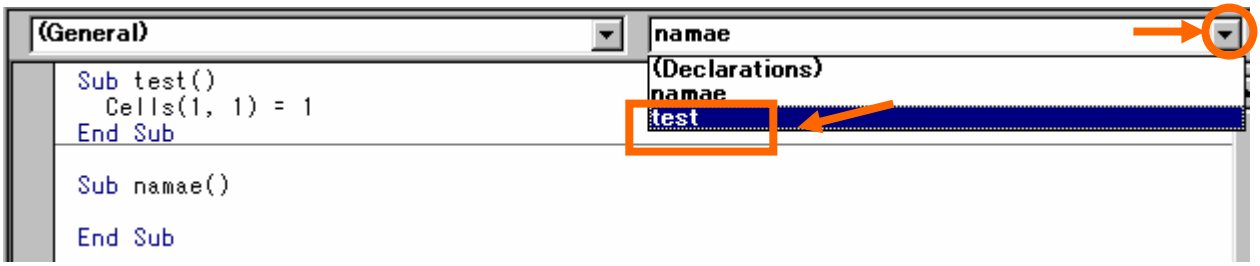


図 3.2.11

このように簡単にマクロを増やすことができる。実際に 2 つのマクロが存在しているのがわかる（図 3.2.12）。

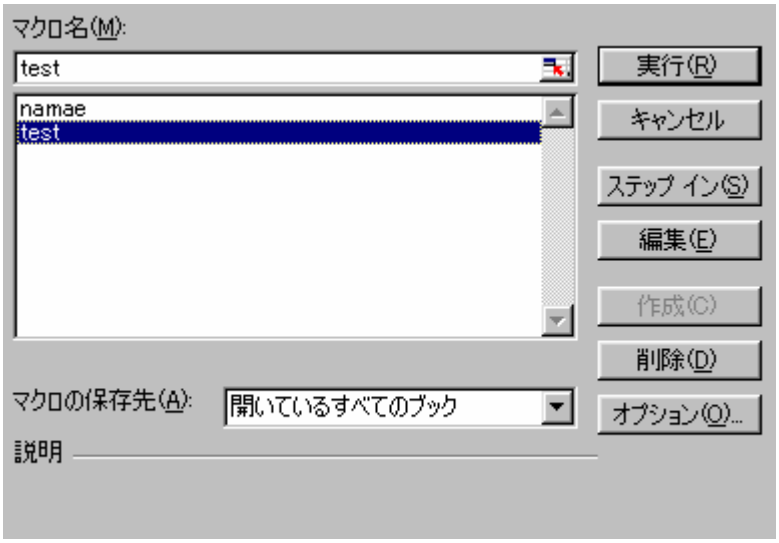


図 3.2.12

§3.3　プログラムを構成する単語

プログラムを作成するときにいろいろな道具を使わなければならない。それぞれの単語を学びながら文法をマスターする。

Check Point

1. 予約語と変数を学ぶ。
2. 演算子に関するきまりをマスターする。

プログラムはステートメント、関数、演算子、変数、定数で構成されている。

ステートメント	For　To　Next　Sub
関数	Sin()　Cos()　Tan()　Sqr()
演算子	+　-　*　^
変数	I, k, nx,
定数	5　　“Hello”

① ステートメント

プログラム中に記述する BASIC の基本命令。

（図 3.3.1 のように自動的に**青色**になり先頭の文字が**大文字**になる）。

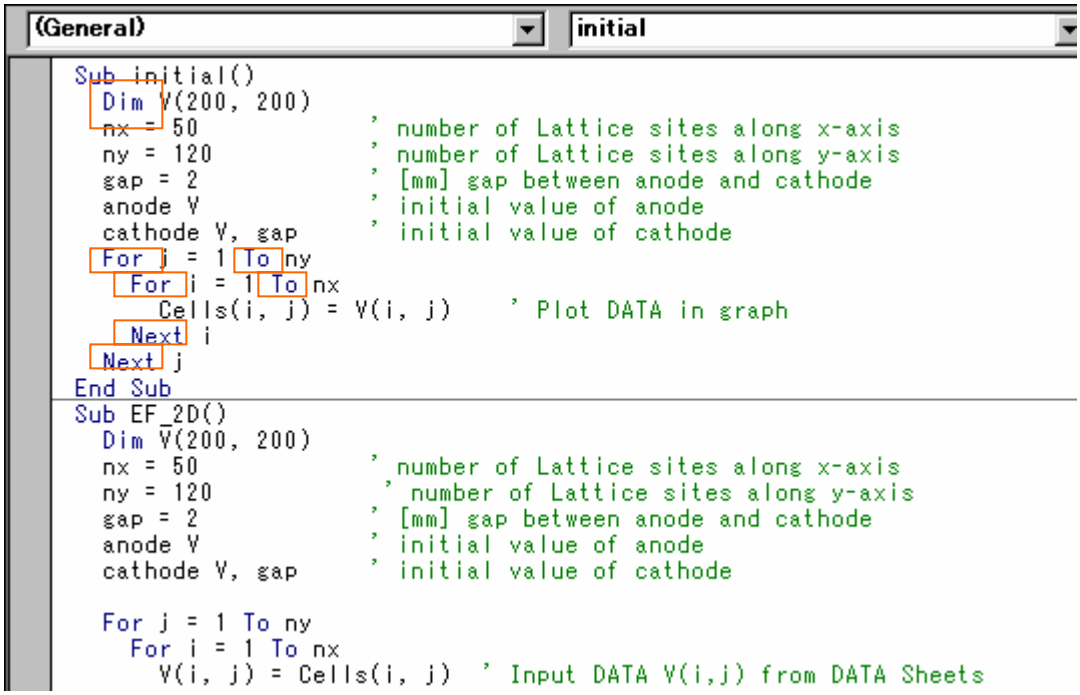


図 3.3.1

② 関数

() 内にデータを与え、その結果を得る命令を関数と呼ぶ（図 3.3.2）。（自動的に先頭の文字が大文字になる。文字の色は黒。）

数学の『 **f(x) = sin(x)** 』と同じ概念である。

プログラム上では、『 **Y = Atn(1)** 』となり、変数 **Y** に **0.785398** が代入される。

また、Visual BASIC の関数の種類は、ヘルプの目次に関数の項目があるので各自調べること（図 3.3.3）。

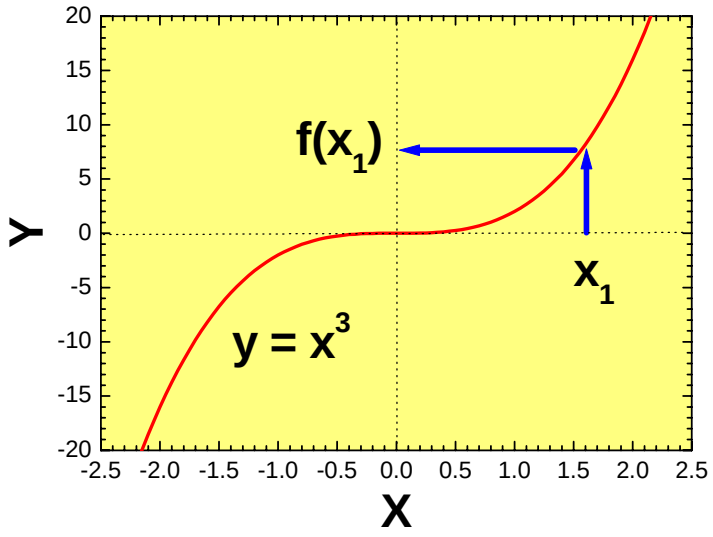


図 3.3.2

図 3.3.3

③　演算子

足し算や掛け算などの計算を行う記号のこと。演算子の優先順位は、以下のようになる。

また、演算子にも以下の３種類があります。

算術演算子	^　*　/　Mod　+　-
関係演算子	=,　<>,　<,　<=,　>,　>=
論理演算子	Not,　And,　Or

^	べき乗
*	乗算
/	除算
MOD	整数除算の余り
+	加算
-	減算

注意：

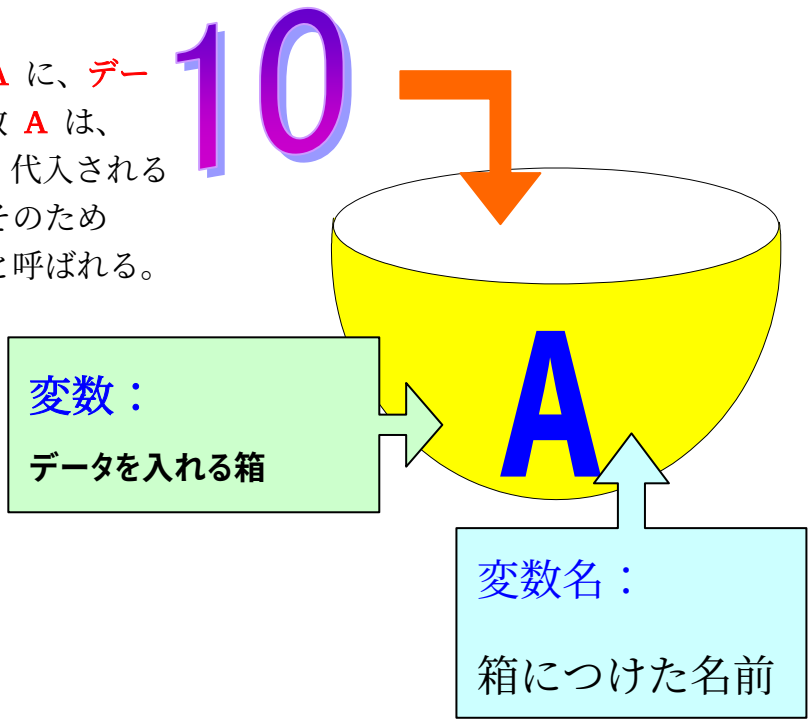
MOD 演算子の両脇は必ず 1 つ以上の空白をとる。

$\frac{a}{bc}$　は、プログラムでは　**a / b / c** となる。

課題 1.　**-2 ^ 2**　と　**(-2) ^ 2**　の計算をしなさい！

④　変数

『 A = 10 』の意味は、変数 A に、データ 10 を入れるということ。変数 A は、値を入れる容器みたいなもので、代入される値によってその内容が変わる。そのため『変わる数』という意味で変数と呼ばれる。



プログラム上では、
『 A = A + 1 』
の『＝』は代入するという意味である。数学の等式の意味とは異なる。例えば、

A = 1
A = A + 1

で計算結果は 2 となる。

変数名をつけるときの決まり

1. 名前の先頭は必ずアルファベットで始まらないといけない。
2. 名前の 2 文字以後はアルファベット、数字、＿　などが使用できる。
3. 予約語（コマンド、ステートメント、関数）は使用できない。
4. 大文字と小文字の区別はない。
5. “ 文字型変数 ” や “ コメント文（緑色）” 以外は、英数半角しか使用してはいけない。

課題 1.　自分自身に 2 を掛ける演算をプログラムで書きなさい！

⑤　データの型

　プログラムではデータの型にこだわる。原則的にデータの型の異なるもののやりとりは行わないように制限している。

整数型	２バイト	-32768　～　32767	
倍長整数型	４バイト	-2147483648　～　2147483647	
単精度実数型	４バイト	-3.402823E+38　～　3.402823E+38	
倍精度実数型	８バイト	-1.797693134862316E+308　～　1.797693134862316E+308	
文字型	可変長	最大 32767 文字	“Hello” のような文字を扱う。“”の中は全角でも日本語でもよい。
	固定長		

注意：

E+38 は 10^{38} を表します。
E-20 は 10^{-20} を表します。

変数の後に%、&、！、#、＄をつけて、それぞれの型を区別する。

A%　・・・　整数型
A&　・・・　倍長整数型
A！　・・・　単精度実数型
A#　・・・　倍精度実数型
A$　・・・　文字型

バイトの話

コンピュータの世界は 0 と 1 だけの値をとる２進法の世界です。各桁を１**ビット**と呼ぶ。

１ビットを 8 つ並べた大きさを 1 **バイト**と呼び、記憶領域の大きさを表す基本単位とする。

１バイトで表せる値は、00000000～11111111 の 256 種類！

$2^8=256$ となるためである。

２進法	１０進法
00000000	0
00000001	1
00000010	2
00000011	3
・・・	・・・
11111111	255