●VBAの主な「学習項目＝ステートメント（ひいてはプロシージャ、クラスモジュール）を構成する要素たち」のイメージ　（※VBA自体にいい加減なところがあり、また、命令の数が多すぎて、完全にカテゴリ分けするのは「かなりのプロ」でも「困難＝多分不可能」なので、あくまでも「イメージ」です。）

（＝表の水色とピンクの部分がそれです。　**特に、ピンクの部分は 「それをやらないと、たとえお金を払って習っていたとしても、いつまでたっても、上達が100%できない重要箇所」＝「真のスタートライン」 です。　でも、そのことを「ちゃんと」詳しく書いてくれている市販書籍は日本には、「1冊も」ありません。**

それはまるで、「わざと教えずにわざと挫折させられているかのよう」です。 「真のスタートライン」の話を、「初級の本が終わったあとの方向性として」すら、教えてくれませんのです。

このことは、Excelが発売されて以来、「20年以上も前から」「全く変わっておらず」、「全く改善無し」の「放置」状態です。 出版社や編集者、著者、資格試験実施者は、いったい何をしているのか？　大いなる疑問を覚えます。特に、子供たちの未来（プログラミング教育）を考えると。 また、私達初心者からすれば。）

※「プログラミング」とは、どんな言語であっても、「主に生データを変数と呼ばれるモノに」「代入」して、それを「操作」し、「自動化を図る作業」です。

やりたいこと・利用シーン			学習項目01 「生データ」 「生データ」、あるいは、 「生データに相当する」モノ 。 **ヘルプやウォッチウィンドウでは、 「値」と呼ばれています。**	学習項目02 「変数」 「生データ」を 代入する「変数」 **＝VBAにおける、 実質の「操作対象」**	学習項目03 「命令01（組込タイプ）」 ＝「キーワード」 「変数」を操作するための主な道具 （「絶対」ではない） **※Excelにあらかじめ 組み込まれている命令たち**	学習項目04 「命令02（自作タイプ）」 ＝「自作のキーワード（超意訳）」 自作できる・できない **※ユーザーが自作する命令たち**	ヘルプで言うところの 「キーワード」か否か？ （ムリヤリ意訳）	「返す」という仕組みを 持つ・持たない （あるいは「評価される」か否か）	戻り値（返り値） （＝「評価」されたモノ） （＝「評価」が確定したモノ）	「式」か「式じゃないモノ」か？ 式を作れるか作れないか？ （「ステートメント」内において）
VBAプログラミングの場合	一般データ操作（下ごしらえ的）	・「グラフやシートなどの各種オブジェクト」やその他のモノに対する、 各種設定値の作成。（各種「引数」等々の作成。） ・VBA関数やメソッド、プロパティに渡す「文字ベースの引数」の値の加工。 （各種「引数」の作成。） ・単純に計算（四則演算等々）。 ・開始日付と終了日付による期間日数の割り出し。 ・日付データからの生年月日や締め日、年度末、月度、年度、などの算出。 ・置換（変数内の。ある文字列内の、特定の短い文字列や単語などの。）。 ・検索（変数内の。ある文字列内の、特定の短い文字列や単語などの。）。 ・テキストファイルやCSVファイル処理の操作（読み書きその他。）。 ・フォルダ内の全てのサブフォルダのファイルのフルパスの取得。 ・フルパスからのファイル名のみやパスのみの切り出し。 ・文字列の結合、分離、切り出し、など。 ・True/Falseの判断。 ・フラグ処理。 ・文字コード処理。 ・16進数、2進数変換。その他の各種変換。 ・配列操作。 ・API利用、「SQL文」の作成。 などなど	一般データ （＝各種リテラル値） ・文字列 ・日付 ・論理値（ブール値＝TrueとFalseの2値） ・数値 などの 「文字系」のデータ。 言わば「2D」なデータ。 ※「型変換文字」を付加したモノも含む。 ※一般データは「値と型が確定する」ため、1単語であったとしても、「式」のかたちになります（「キーワード」には属しませんが）。それはウォッチウィンドウで明確に確認できます。**「式」は、ウォッチウィンドウでも、「式」と明示されます（エラー表示されないモノのみ）。** ※VBAにおける「式」とは、「数式」だけではありません。形としては、例えば「化学式」に少し似ているというか、そのように単語1語でも、また、日付や文字列でも、「式だ」ということになります。	一般変数 （「配列」も含む） ・文字列型 ・日付型 ・ブール型 ・数値型 整数型、長整数型、 単精度浮動小数点型、 倍精度浮動小数点型、 通貨型、 10進型、 etc などなど ※一般変数も同じ理由で「式」のかたちになります。これもウォッチウィンドウで明確に確認できます ※なお、ご自分が作った「変数」＝「変数名」は「キーワード」には属しません。ただ、上記の各・型の名前は、「キーワード」に属すると考えてOKです。	ステートメント名＝ステートメント用のキーワード （主に、フロー制御系、定義系、代入系、呼び出し系、など）	できない	キーワード	持たないものがほとんど	なし	基本、式じゃないモノ・式を作れない（例外有）
					演算子	できない	キーワード	持つ （計算の答えやTrue/Falseなど）	一般データ	式・引数めいたものがあれば単独で式を作れる
					VBA関数・ほどんどの。　多分9割以上。 （Excel専用、Office共通、その他）	できる。SubやFunctionのプロシージャ（＝一般データを返す自作関数）として。引数ある/なし。	キーワード	持つ （同上）	ほとんど一般データ。 （例外的にオブジェクトもあるかも。 未調査）	式・引数があれば単独で式を作れる
					WorksheetFunctionオブジェクトの関数 （原則、「ワークシート関数」と同じ動きをVBA上でも行えるようにしたもの ）	できる。SubやFunctionのプロシージャ（＝一般データを返す自作関数）として。引数ある/なし。引数ある/なし。	キーワード	持つ （同上）	一般データ （同上）	式・引数やオブジェクト装飾子（＝WorksheetFunctionオブジェクトを表す式やオブジェクト変数）があれば単独で式を作れる
					各種引数名	できる	キーワード	持たない	なし	式じゃないモノ・単独では式を作れない
					（一般引数の値 自体）	できる、というか、するしかない	キーワードじゃない。式。	持つ（評価される）	主に一般データ	式
					（一般引数に相当するモノ（特に演算子の場合））	できる、というか、するしかない	キーワードじゃない。式。	持つ（評価される）	一般データ	式
					（オブジェクト引数の値 自体）	できる、というか、するしかない	キーワードじゃない。式。	持つ（評価される）	オブジェクト	式
					「:=」のような単なる「記号」	できない	キーワード	持たない	なし	式じゃないモノ・単独では式を作れない
					データの型の名前（一般データ用のもの）	できる	キーワード	持たない	なし	式じゃないモノ・式を作れない
					定数	できる	キーワード	持つ	一般データ（数値）	式
	オブジェクト操作（メイン的）	・単一ブックの単一操作や複数ブックの同時操作。 ・単一シートの単一操作や複数シートの同時操作。 ・『「オブジェクトそのもの」を「引数」として使用する関数、 プロパティ、メソッドなど 』の、その「引数」の作成。 ・Excelファイル以外のファイルの操作（テキストファイル、 画像ファイル、Wordファイル、PwPointファイルなど）。 ・フォルダ内の全てのサブフォルダの各種ファイルの一括操作。 ・単一セルの操作や、セル範囲の操作。　・ウィンドウの操作。 ・グラフの操作。　・ピボットの操作。　・名前定義の操作。 ・フィルタの操作。　・並べ替え機能の操作。 ・「Excel上での操作としての」、検索や置換、その他。 ・Microsoft Query（簡易SQL）の操作。 ※Accessの「クエリ」や、SQL Serverの「ビュー」と同じような機能です。 ・数式の操作。　・色や罫線、各種書式設定の操作。 ・オートシェイプ（図形）や、画像（図、写真等々）の操作。 ・印刷レイアウトの操作。 ・印刷設定の操作や印刷実行や連続印刷。 ・テキストファイルやCSVファイル処理の操作（読み書きその他。）。 ・Webブラウザ操作。　・クリップボードの操作。 ・Word操作、Access操作、PwPoint操作、Outlook操作、など。 ・「VBAコードやVBE画面自体」などの操作。　・API利用。 **・「オブジェクト式」の作成** **※「生のオブジェクト」＝「オブジェクト式」は、プロパティやメソッド、一部の関数などを使って表現しますので、「オブジェクト名」というモノを直接、「オブジェクト式」の中に書くことはありません。** **（「オブジェクト名」を書くのは変数宣言のステートメント内だけです。）** などなど	オブジェクト ＝オブジェクト式　（「生データに相当する」モノ） ＝特に、「階層構造」や「Itemプロパティ」を 「省略しない」モノ、 巻の書籍は省略だらけでドンボにハマるため。 「装置系」のイメージのデータ。 言わば「3D」なイメージのデータ。 ※「生オブジェクト＝オブジェクト式」のため、「オブジェクト」は「式」のかたちになります。 オブジェクトは、**大別すると、 「単一オブジェクト」と 「コレクションオブジェクト」の 「2つ」しかないです。 また、オブジェクトモデルのその「階層」には、 「単一オブジェクトのみしか無い階層」と、 「コレクションオブジェクトと対になっている階層」とがあります。 「新規作成」や「追加」ができる単一オブジェクトの階層には、それと対になるコレクションオブジェクトも同居しており、逆に、新規追加などができない単一オブジェクトの階層にはコレクションは在りません。 また、超重要事項として、コレクションは「単一と同じ階層に居る」と考えると階層構造を理解しやすくなります。**	オブジェクト変数 （「配列」も含む） ・総称型 Object型、Variant型 ・固有型＝オブジェクト名 Application型、 Workbooks型、Workbook型 Worksheets型、Workbook型、 Sheets型、Range型、Font型、 Charts型、Chart型、 AutoFilter型、Sort型、 etc などなど ※「オブジェクト変数」も他と同じ理由で「式」のかたちになります。これもウォッチウィンドウで明確に確認できます。 ※なお、こちらも、ご自分が作った「変数」＝「変数名」は「キーワード」には属しません。ただ、上記の各・型の名前は、「キーワード」に属すると考えてOKです。	ステートメント名＝ステートメント用のキーワード （主に、フロー制御系、定義系、代入系、呼び出し系、など）	できない	キーワード	持たないものがほとんど	なし	基本、式じゃないモノ・式を作れない（例外有）
					演算子	できない	キーワード	持つ	一般データ	式・引数めいたものがあれば単独で式を作れる
					VBA関数・ほんの少し。　多分1割以下。 （CreateObject関数やGetObject関数など、その他）	できる。SubやFunctionのプロシージャ（＝オブジェクトを返す自作関数）として。引数ある/なし。	キーワード	大抵、持つ	オブジェクト。 （例外的に一般データもあるかも。未調査）	式・引数があれば式を作れる
					プロパティ	できる。Propertyプロシージャとして （クラスモジュールやシートモジュールなどにて）	キーワード	持つ	一般データやオブジェクト。	式・引数やオブジェクト装飾子（＝オブジェクト式やオブジェクト変数）があれば単独で式を作れる
					メソッド	できる。SubやFunctionのプロシージャとして。（＝一般データやオブジェクトを返す/返さない自作メソッド。クラスモジュールにて）	キーワード	一応持つけど、持たないモノの方が多い。多分。	「返す」仕組みを持つモノのみで、 一般データやオブジェクト。 何も返さないモノの方が多い。	式 or 式じゃないもの 式を作れたり作れなかったり。
					イベント	できる（クラスモジュールにて）	キーワード	持たない	なし	式じゃないモノ・式を作れない
					列挙（「生データ」に含むべきかも。直接は書かないので）	できる	キーワード	持たない	なし	式じゃないモノ・式を作れない
					定数	できる	キーワード	持つ	一般データ（数値）	式
					各種引数名	できる	キーワード	持たない	なし	式じゃないモノ・単独では式を作れない
					（一般引数の値 自体）	できる、というか、するしかない	キーワードじゃない。式。	持つ（評価される）	一般データ	式
					（オブジェクト引数の値 自体）	できる、というか、するしかない	キーワードじゃない。式。	持つ（評価される）	オブジェクト	式
					「:=」のような単なる「記号」	できない	キーワード	持たない	なし	式じゃないモノ・単独では式を作れない
					データの型の名前（オブジェクト用のもの、総称型・固有型）	できる（クラスモジュールにて）	キーワード	持たない	なし	式じゃないモノ・式を作れない
					↑↑これが俗に言う「オブジェクト名」です。					
					※「生オブジェクト＝オブジェクト式」は、					
					プロパティやメソッドなどを使って表現しますので、					
					俗に言う「オブジェクト名」は 使いません。					
お料理の場合		下ごしらえの種類 ========== メイン調理の種類	素材。 お肉・お魚 野菜 お米 パン 穀物 くだもの	素材を仮置き、 下ごしらえ置きする ための小皿、ボウルなど ※あるいは、「名前のついた計算用のメモ紙」、または「名前の付いたクリップボード」、のようなものです。	包丁 まな板 ナベ 菜ばし おたま フライパン 電子レンジ オーブン	自作の道具 自作の調理器具				

↑全部、ウォッチウィンドウでチェックできます。

●「プロシージャ（＝命令ブロック）」、「ステートメント（＝命令文）」、「式」、について　https://thom.hateblo.jp/entry/2016/05/25/140304　←参考URL。ほんの少し「あれ？これってそうなのかな？」なところもありますが・・・

VBAプログラムの最小単位である、「プロシージャ（＝命令ブロック）」は、複数の「ステートメント（＝命令文）」の集まりでできています。
そして、その「ステートメント（命令文）」は、これまた複数の、「式」と「式じゃないモノ」（上表の一番最後の列を参照）でできています。

そして、「式」には、変数やリテラル値などの単語1語のモノもあれば、それらとプラス、演算子・関数・プロパティ・メソッドなどの複数の命令単語が組み合わされてできた長めのモノのもありますが、最終的には「評価が完了したモノ＝評価が確定したモノ＝**値と型が確定したモノ**」のことを言います。
その簡単なチェック方法としては、「ウォッチウィンドウ」に調べたい「式」を追加した状態にしてから F8キーでの1行ずつのステップ実行をするなかで、その「型」の列に明確に型の内容が表示され、かつ、「値」の列にエラー表示が無ければ、「それは式だ」・「式のていをなしている」と判断・断定できます。（「値のエラー」が無く「型」が明確に決まればOK！ 的な。）
基本、たとえば長い式の中には、複数の短い式（変数、定数、リテラル値など）が含まれることが多いのですが、それ自体が「数式、文字列式、日付式、ブール式」などであることが多い・・・という感じです。ちょっと説明が、難しいんですけれど・・・。

ただ、「式」単位で・・・つまり、「式」か「式じゃないか」の単位でプログラミングを考えると、エラー解決がしやすくなったり、VBAの仕様・構造そのものが理解しやすくなります。
つまり、「VBAの上達が早まります。」
人によっては後述の5つの自作関数の作成と共に、「式」を意識・頻繁にチェックすると、「ある地点から」「爆発的に」「上達」します。理由は・・・、まず、「式単位」でモノを見るには「ウィッチウィンドウ」を頻繁に使うことが近道なんですが、それを使って「式単位」でモノを見るクセにすると、「型のリレー」をミスることで出るエラー を相当数、減らせるからです。
基本、初心者のうちに出すエラーの半分以上が、その「型のリレーミス」かタイプミスでのエラーなので、特に、です。ですので、是非、ウォッチウィンドウで、「式」を調査するクセをつけてください。
また、演算子・関数・プロパティ・メソッドなどに使う「引数(or 引数に相当するモノ)」は、必ず「式」の形になっています。 その意味からも「式単位」でモノを見れるようになると、エラーが減りますし、VBA自体の構造を理解しやすくなります。

あと、「超」「重要なこと」として、「自作関数」には、大別して、以下の5つのタイプがあります。（「引数や既定値がある／なし」を含めて。）

(01) 一般データを返すこと「だけ」、しかしないタイプの自作関数
(02) 何らかの動きをしつつ、同時に、一般データをも返すタイプの自作関数
(03) オブジェクトを返すこと「だけ」、しかしないタイプの自作関数
(04) 何らかの動きをしつつ、同時に、オブジェクトをも返すタイプの自作関数
(05) 何も返さずに、何らの動きをする、だけのタイプの自作関数。

●「式」の種類と意味。（ヘルプより）
数式、文字列式、日付式、ブール式、オブジェクト式、時刻式、共通日付形式、など。「文字列、数値、オブジェクトのいずれかに評価される、キーワード、演算子、変数、定数の組み合わせ」のこと。

●「キーワード」の意味（ヘルプより）
キーワードとは、Visual Basic プログラミング言語の一部として認識される文字 列または記号。ステートメント名、関数名、演算子などがあります。（**※VBAにおいての「キーワード」とは、世間一般の常識とは異なり、単に「命令単語」というだけの意味であって、「重要語句」という意味あいはありません。**）

そして、VBAの命令は、演算子、各種VBA関数、WorksheetFunctionオブジェクトのメソッド、プロパティ、メソッド、ステートメント名、API、など、色んなカテゴリに分かれていて、合計で（Excelの場合だけでも）、少なくとも1000個以上の命令単語がありますが、実は・・・、それらは全部、上記の(01)～(05)の、5つの自作関数と同じ、あるいは似た動きをします。
「演算子、関数、プロパティ、メソッド、云々…」といったカテゴリ名に「過度」に惑わされないでください。その呼び名、カテゴリ名の壁・枠を飛び越えて、1000個以上の命令たちはすべて、この「5つのどれか」に当てはまります。 また、そう考えたほうが、VBA自体の構造やクラスモジュールの仕組みを理解しやすくなると思います。
そして、上記の(01)～(05)の5つの関数が、引数や既定値の有り無しを含め、色んなタイプのものが自由に作れることこそが、「初心者本を終えたあとの次なる目標」であり、「VBAの ”真のスタートライン” 」でもあります。それができれば、ある地点から「爆発的に」「上達する」確率が上がります。