●ExcelVBA命令単語5パターンの、「大雑把」な、分類の一覧表（あくまでも大雑把なイメージ。厳密には区分が難しいので。紐付くオブジェクトによっても動きが変わる。なお、分類の考え方としては、一番下の「●アとイ、あるいは、ウとエの区分けの考え方の事例」を参考にして下さい。

（「大雑把」ではありますが、しかし、実際でコードを書いたりデバッグする上で 何らかの破綻やエラーが起きることは無いレベルにはなっているとは思います。また、一般的な教え方通りに「演算子、VBA関数、プロパティ、メソッド、ステートメント用キーワード、などの「命令の各カテゴリ」通りの「細分化された分類」を「全部理解しようとする」と、「結局、この単語はコードの中で何者なのか」がかえって見えなくなってしまうので、この表のような「動きベース」の分類で最初は理解して下さい。

「命令の各カテゴリ」的な理解は「後付け」で理解を深めていって「全然違くありません」（各種カテゴリ名はデバッグ時には、中くらいの意味はありますが、「動き」のほうがデバッグに大きな意味があるので。カテゴリ名は学術的な整理・基本動作の「検索、情報整理、につかう意味のほうが大きいので」。なので「命令単語の理解は、まずは（先に）「動きベース」で考えましょう。「カテゴリベースではなく」。）

※この表は、初心者の方においては、「１年かけて」「初心者本でVBAの概要を掴んだ後に」、「なぜこうなるのか？」と「学習すべきこと」です（丸暗記する必要はなくて、なぜ、これがこのマス目に入るかが自分で調べられればいいのです。ヘルプやオブジェクトブラウザ、ウォッチウィンドウ、最後にAIを使って）。慌てずにゆっくり、学習のたたき台にしてください。

ただし、「なぜ？」を調べるには、VBA書籍やサイト、動画、記事、は使わないでください。ウソが書いてあることも少なくないので。「式」の「意味」を理解後に、ヘルプやオブジェクトブラウザ、ウォッチウィンドウ、最後にAIを使って、にしてください。　なお、この表が理解できると「１年後の爆発的な成長」に結びつく可能性が高いです。

※なお、プロパティ名やメソッド名は、本当はオブジェクト名とセットでないと正しくありません。（表の下を参照）
例えば「Value」なら、「Range.Value」「Name.Value」「PivotField.Value」「Application.Value」などがあります。それぞれ返ってくるものや動き、性格、なども変わります。
ただこれ、セルの大きさが小さすぎて書ききれないので、しかたなく省略してあります。ただ、オブジェクトブラウザまはAIに聞く、などで正しい書きかたを確認できます。（AIに聞く時は「ExcelVBAにて、×××プロパティ（or ×××メソッド）と紐付くオブジェクトを全部列挙して下さい」と聞いて、その動き聞いたり、実際にウォッチウィンドウなどでご自分で動きを見てみてください。）
なので、ご自分でも「紐付くオブジェクトをAIに聞くなどしてから」、「アーエ」のどこに区分けされるか考えてみてください。AIに聞いたり、ヘルプを読んだり、デバッグしたり、ウォッチウィンドウを見たりして、実際に動きを確認してみて。ただし、AIに色んなことを聞く時は、聞く前に、「先に」「オブジェクトブラウザとウォッチウィンドウが使えないと」、AIがウソをもとして学習しちゃう（つまりウソ）を教えてくれることがあるので、注意してください。「オブジェクトブラウザとウォッチウィンドウではこうだけどなんてあんたはそんなこと言うの？」みたいな質問の仕方がいいです。
なお、私の考えは「何らかの動き」とは、「本来人間がポチポチと行う面倒な作業をパソコンが代行・自動化してくれている」、というリベルの「動き」を意味しています。CPUが内部・裏側でやる細かい動きは含めません。具体的に言うと、「文字列の中だけでの検索、置換、トリミング、結合、判定」や「数値での四則演算、判定」、「Excel画面における、セル上での検索、置換、結合、判定、各種新規作成、条件設定、選択、入力、削除、コピー、貼り付け、切り取り、集計操作、フィルタ、並べ替え、図形操作、図操作、グラフ操作、ピボット操作、パワークエリ操作、
あるいは、Excel画面のウラに隠れての、各種通信、エラー処理、ネットワーク処理、コマンド実行、OS連携、他のソフトとのCOM連携、などなど、「全て」です。（また、例えば、Excelだけでなくパソコンでも、例えば壁に打込まれた鉄筋コンクリートのように、「目には見えない隠れたもの」もあります。このようなものを理解がなかなか難しいです。つまり「目には見えていないもの」「オブジェクト」「機能」「表面」と「ユーザ」というケースがある、ということです。それも結構多いです。）

パターン	演算子 （自作は当然できない。）	VBA関数 （＝VBAの内部的にはいろんなオブジェクトのメソッド。自作は当然できない。）	WorksheetFunctionオブジェクトの関数 （＝VBAの内部的にはWorksheetFunctionオブジェクトのメソッド。こちらも自作は当然できない。）	組込のプロパティ （組込のものの内容変更は当然できないが、「自作のオブジェクト＝クラスモジュール」の中では「プロパティ」自体は自作できる。）	組込のメソッド （組込のものの内容変更は当然できないが、「自作のオブジェクト＝クラスモジュール」の中では「メソッド」自体は自作できる。）	ステートメント用キーワード （こちらも自作は当然できない。）	自作関数01（＝自作メソッド01） Functionプロシージャ	自作関数02（＝自作メソッド02） Subプロシージャ	自作のプロパティ01 PropertyLetプロシージャ	自作のプロパティ02 PropertySetプロシージャ	自作のプロパティ03 PropertyGetプロシージャ
(ア) プリミティブな型の値を返す「だけ」のもの（完全に無加工・そのまま返すもの） **※「＝式」となります。 変数の代入や引数、 条件式の作成、などに使えます。 なお、変数への直接代入時は、 「LET」が先頭に必要。 （はじめのうちは省略しないほうが 脳は上がる）**	(該当なしor未確認)	CurDir	Pi（ここかもしれない。）	Address, Count, Name, Value, Row, Column, Top, Left, Width, Height, Text, Type, Visible, Enabled, AllowFiltering, Creator, FontStyle, Size, Color, ColorIndex, Bold, Italic, Underline, NumberFormat, NumberFormatLocal, Formula, Formula2, FormulaR1C1, HasFormula, RowHeight, ColumnWidth, WrapText, MergeCells, PageSetup, PrintGridlines, Tab, DisplayAlerts, ScreenUpdating	(該当なしor未確認)	存在できない。 ステートメント用のキーワードは、「式じゃないもの」の代表格です。	自由に作れる Function test01() As String Let test01= ""あい"" End Function みたいな内容のものです。	存在できない	存在できない	存在できない	自由に作れる
(イ) 何らかの動き・計算・加工をしつつプリミティブな型の値を返すもの **※「＝式」となります。 変数の代入や引数、 条件式の作成、などに使えます。 なお、変数への直接代入時は、 「LET」が先頭に必要。 （省略しないほうが 脳は上がる）**	+, -, *, ¥, Mod, &, =, <, >, <=, >=, Like, Is, And, Or, Not, Xor, Eqv, Imp ※ただし、演算子単体ではなにもできず、オペランド（引数みたいなもの）とセットでないと値が返りません。	Abs, Asc, Chr, Cos, Date, DateAdd, DateDiff, DatePart, DateValue, Day, Dir, Error, Exp, FileAttr, FileDateTime, FileLen, Format, Hex, Hour, InStr, InStrRev, Int, Fix, IsArray, IsDate, IsEmpty, IsError, IsMissing, IsNull, IsNumeric, IsObject, LBound, LCase, Left, Len, Log, LTrim, RTrim, Trim, Mid, Minute, Month, MonthName, Oct, Replace, Right, Rnd, Round, Second, Sgn, Sin, Space, Split, Str, StrComp, StrConv, String, Tan, Time, TimeValue, TypeName, UBound, UCase, Val, VarType, Weekday, WeekdayName, Year, InputBox, MsgBox	Acos, Asin, Atan2, Average, Ceiling, Count, CountIf, Days, Db, Ddb, Match, Max, Min, Power, Product, Proper, Rand, RandBetween, Roman, RoundUp, RoundDown, Search, Slope, Sqrt, Sum, SumIf, VLookup, XLookup	(該当なしor未確認)	ExecuteExcel4Macro, Run, Evaluate, Compute, BreakSideBySide, CompareSideBySideWith, PointsToScreenPixelsX, PointsToScreenPixelsY, CanCheckIn, CanCheckOut, ExclusiveAccess, PublishToPBI, XmlImport, XmlImportXml, XmlMap.ExportXml, AdvancedFilter, AutoComplete, CopyFromRecordset	存在できない。 ステートメント用のキーワードは、「式じゃないもの」の代表格です。	自由に作れる Function test02() As String Dim Tmp01 As Integer Let Tmp01 = 100*15 Let Tmp01 = Tmp01+20 Let test02="あ，" & Tmp01 End Function みたいな内容のものです。	存在できない	存在できない	存在できない	自由に作れる
(ウ) オブジェクト型の値を返す「だけ」のもの（完全に無加工で 既存のオブジェクトを特定・参照するもの） **※「＝式」となります。 変数の代入や引数、 条件式の作成、などに使えます。 なお、変数への直接代入時は、 「SET」が先頭に必要。 （省略不可能。）**	(該当なしor未確認)	(該当なしor未確認) ※VBA関数全体の補足 VBA関数は、実は、「RangeやWorksheetなどが属するExcelライブラリ」には属していません。「VBAライブラリ」に属するメソッドやプロパティです（ExcelだけでなくWordやAccessでも使えるようにするため）。 なので、本来は呼び出しコードの記述として、例えば、「VBA.Strings.Len,VBA.Math.Abs, VBA.DateTime.Date」などと書かないといけませんが、でも、実際には、「VBA.Strings.」や「VBA.Math.」や「VBA.DateTime.」などは省略記述が可能です。 それは「グローバル」という仕組み（コード記述のみ専用？の特殊なオブジェクト？）があるため、省略できるようになっています。	(該当なしor未確認) ※WorksheetFunctionオブジェクトの関数全体の補足 WorksheetFunctionオブジェクトの関数は、Excelライブラリに属するメソッドです。 Excelライブラリに属するメソッドですので、使えるのはExcelだけで、WordやAccessでは使えません。 なお、WorksheetFunctionオブジェクトの関数の場合は、「グローバル」という仕組みで認められている省略は、「Application.」だけなので、コードの記述としては「WorksheetFunction.」を省略することができません。	Item,Value,ActiveCell, ActiveChart, ActiveSheet, ActiveWindow, ActiveWorkbook, Application, Cells, Charts, Columns, Rows, Selection, Sheets, Workbooks, Worksheets, Range, ActivePrinter, ActiveProtectedViewWindow, AutoFilter, BorderAround, Borders, BottomMargin, Characters, ChartObjects, Comment, CustomViews, Dialogs, Errors, Font, FormatConditions, Hyperlinks, Interior, Item, ListObjects, Name, Parent, Workbook, Worksheet, Workbook.PivotTables, QueryTables, RecentFiles, Shapes, Styles, Validation, Windows	Item, Next, Previous, Worksheet.PivotTables, PivotCaches, ChartGroups, ChartObjects, FullSeriesCollection, OLEObjects, SeriesCollection, Scenarios	存在できない。 ステートメント用のキーワードは、「式じゃないもの」の代表格です。	自由に作れる Function test03() As Worksheet Set test03=Activesheet End Function みたいな内容のものです。	存在できない	存在できない	存在できない	自由に作れる
(エ) 何らかの動き・生成・検索もしつつオブジェクト型の値を返すもの **※「＝式」となります。 変数の代入や引数、 条件式の作成、などに使えます。 なお、変数への直接代入時は、 「SET」が先頭に必要。 （省略不可能。）**	(該当なしor未確認)	CreateObject, GetObject	(該当なしor未確認)	(該当なしor未確認)	Offset, Resize, SpecialCells, CurrentRegion, End, Add, Open, Find, FindNext, FindPrevious, Copy, AddComment, ColumnDifferences, RowDifferences, Duplicate, GetWorkflowTasks, GetWorkflowTemplates, OpenXML, XmlDataQuery, XmlMapQuery	存在できない。 ステートメント用のキーワードは、「式じゃないもの」の代表格です。	自由に作れる	存在できない	存在できない	存在できない	自由に作れる
(オ) 何らかの動きをするだけで 何も返さないもの **※「＝式」とは「なれません」。 変数の代入や引数、 条件式の作成、なども 「使えません」。 無理に使おうとしてもエラーになるだけです。 「何らかの動作をするのみ」です。**	(該当なしor未確認)	(該当なしor未確認)	(該当なしor未確認)	存在できない。多分。 プロパティには必ず戻り値が存在します（多分どのオブジェクト志向の言語でも）。 なので、戻り値が無いプロパティは、恐らく この世にはありません。（読み書きに関わらず、左辺右辺に関わらず、引数にしようが、長い式の一部であろうが、何だろうが。） だから、このマス目にはプロパティは存在できません。 ちなみにですが、FunctionもSubも、PropertyLet/Set/Getも、組み込みのプロパティやメソッドも、VBA関数も、全部、「かっこ を付けるルールは同じ」くらいに思ってください。最初のうちはそれでOKです。 『プロパティは戻り値を使う（取得が設定に）に決まってるから、単にかっこ使わないことがありえない「って だけ」。どういのこじつけでもいいので。 実際、PropertyLet/Set/Getでプロパティを作ると、そんな感覚になるので。 **※余談ですが、**Excel.Application.Workbooks.Item(xxx).Worksheets.Item(xxx).Range といった「オブジェクト式」の**「ドット」は、「フォルダツリー」の「¥」と「同じような役割」**と考えるとオブジェクトモデルの一覧図や、グローバル、などの理解がしやすくなります。 Excel¥Application¥Workbooks¥Item(xxx)¥Worksheets¥Item(xxx)¥Range みたいな。cmdやexe が、「基本は、それが 在るカレントにまで行かないと実行できない」というのと同じです。プロパティやメソッドの実行も。	Activate, Select, Calculate, Clear, ClearContents, ClearFormats, Close, Delete, PrintOut, PrintPreview, Save, SaveAs, Unprotect, Protect, Quit, Refresh, Replace, Sort, AutoFill, CheckSpelling, Copy, Cut, Paste, Merge, UnMerge, Insert, ClearComments, RefreshAll, Undo, Redo, ScrollIntoView, Export, Import	AppActivate, Beep, ChDir, ChDrive, Close, Date, DeleteSetting, DoEvents, End, Erase, Error, Exit, FileCopy, Get, Kill, Let, Line Input #, Load, Lock, LSet, MkDir, Name, On Error, Open, Print #, Put, Randomize, ReDim, Reset, Resume, RmDir, RSet, SaveSetting, Seek, SendKeys, Set, SetAttr, Stop, Time, Unload, Unlock, Width #, Write # ※このマス目に書いてあるのは「ステートメント」ではなく、「ステートメントの性格？を決める」。「キーワード（＝予約語）」です。**ステートメントは「文 のことです。**このマス目に書いてある単語たちは当然「文ではありません」。その文を性格づける「命令？みたいな？でも違うもの？みたいな単語」です。（でも基本、制御ロボットなくてコマンドプロンプトのコマンドのようなものや、関数のような引数を持つもの、ハードウェアに働くもの、まであります。FileCopy, SendKeys, Beep、などなど。だから余計に調べにくい。ステートメント用のキーワードはちょっと動きが同じ分類しづらくて、ある意味「特殊」です。また、演算子と同じく、オブジェクトに紐付けない命令、みたいな側面を持っています。 **ステートメントは「文全体で１つの命令」？みたいな感じかも。その他の単語は単語自体が１つの命令・・・、みたいな感じですが。なので、このマス目の単語自体を「ステートメント」と間違って理解しないでください。（他にはそう誤解させる説明だらけですが・・・。）この単語たちはあくまでも「ステートメント」用の「キーワード」です。** 本来はこの表に書くべきものではないかもしれませんが、でも「あえて」、これらの「ステートメント用のキーワード」を「命令単語の一種」と捉えてしまいます。それは単に「できるだけ例外を減らし、シンプルに考えたいから」です。「最初から在る組込の単語は、（定数以外は）「もうすべてが」「命令だ」、くらいにシンプルに考えることができるからです。難しめはは便宜的に一時的にそのように捉えてください。 もちろん、これらの単語は「何も返しません。そういう「動きベース」で、モノを見てください。	自由に作れる	自由に作れる ※プリミティブ型の値を設定値にできる ※値を設定する、という動きをするだけなので、戻り値は無し。 プロパティの値は、プリミティブな型の値もオブジェクト型の値も、PropertyGetで返してもらいます。	自由に作れる ※オブジェクトを値として設定値にできる ※同左。値を設定する、という動きをするだけなので、戻り値は無し。 プロパティの値は、プリミティブな型の値もオブジェクト型の値も、PropertyGetで返してもらいます。	存在できない

●プロパティ名やメソッド名の正しい書き方の例（オブジェクト名の部分を省略しない正しい書き方。上の表はマス目が狭いので仕方なく省略してあります。）
※↓この書き方はVBAのヘルプと全く同じ書き方」です。オブジェクトブラウザも似ています。逆にこの書きかを知らないと、「同じ単語でも紐付くオブジェクトが異なるとまるで動きが違う命令になる」、ってことがわからないままなので、VBA的にはバカにされます。ご注意下さい。
プロパティやメソッドのヘルプページは、「必ず」、「この書き方で検索」します。でないと、「同名だけど全ったく違う動きのプロパティやメソッド」のヘルプページが出てきてしまい、問題の解決がまったくできなくなります。
なお、この書き方は、もしAIに聞くなら・・・、
『ExcelVBAにて、××××プロパティの正式なメンバー名の表記法（シンプルな修飾名）を教えて。ヘルプでMS公式リファレンスサイトに書かれている「Range.Value」みたいなやつです。複数あったら複数教えて。』
・・・と聞けばいいです。「××××プロパティ」の部分は「××××メソッド」と変えてもOKです。（今この質問文の意味が完全にわからなくてもコピペして聞いてみてください。）
同名のモノが複数ある場合は、色々と出てきますので、その説明を読むと、自分がその中のうち、どれを使えばいいかがわかると思います。
VBA関数列の（イ）の場合：（VBA関数 の列は、検索時には「VBA.Math」や「VBA.String」が不要なことが多いかも。ただし、付けなかったときに検索結果に注意だけはしてください。）
☞ VBA.Math.Abs、VBA.Strings.Left、VBA.Strings.Mid、VBA.Strings.Replace、VBA.DateTime.DateDiff、VBA.Strings.Len、VBA.Information.UBound、VBA.Strings.Split　など。
「WorksheetFunctionオブジェクトの関数」の（イ）の列の場合：
☞ WorksheetFunction.VLookup、WorksheetFunction.XLookup、WorksheetFunction.Match　など。
プロパティ列の（ア）の場合：
☞ Range.Value, Range.Text, Worksheet.Name, Collection.Count, Range.Row, Range.Column　など。
プロパティ・メソッド列の（ウ）の場合：
☞ Application.ActiveCell, Application.ActiveSheet, Worksheet.Range, Worksheet.Cells, Worksheet.Rows, Collection.Item(プロパティ版)　など。
メソッド列の（エ）の場合：
☞ Collection.Item(メソッド版), Range.Offset, Range.Resize, Range.SpecialCells, Range.End, Workbooks.Add, Workbooks.Open, Range.Find, Range.Range　など。

●アとイ、あるいは、ウとエの区分けの考え方の事例（「してから値を返す」を付加して理解してください。ただし、AIに聞いたので完璧ではないかもしれません。）
※アやウは、「～してから返す」ではなく、「既定値か現状」の「プリミティブな型の値orオブジェクト型の値」を単純に返すのみです。
★以下、（イ）の場合
Abs → 計算してから値を返す。
Asc → 変換してから値を返す。
Chr → 変換してから値を返す。
DateDiff → 差分計算
DatePart → 一部分を取り出す
Format → 書式変換
InStr → 検索
IsDate → 判定
Left → 切り出し
Len → 長さを調べる
Replace → 置換
Split → 分割
TypeName → 型を調べる
UBound → 上限を調べる
VarType → 型を調べる
★以下、（エ）の場合
CreateObject → 生成してから値を返す。
GetObject → 取得処理してから値を返す。
Offset → 位置を計算して別Rangeを求めてから値を返す。
Resize → 範囲を計算して別Rangeを求めてから値を返す。
SpecialCells → 条件検索してから値を返す。
CurrentRegion → 連続領域を特定
End → 終端方向をたどる
Add → 新規生成
Open → 開く
Find → 検索
FindNext → 次を検索
Copy（コピー処理後にオブジェクトを返す場合）
→コピーしつつ、そのコピーしたオブジェクトを値として返す
（「才」のCopyは何も返さないケースの場合。
紐付くオブジェクトによってかわる。
ちょっと理解が難しいので分かる人に聞いてください。）