はじめり

紙媒体なら試験時に参照可ということでMarkdownの練習がてら作成しているカンペです。

目次

VBAの始め方

はじめてVBAを作成するときにはExcelで設定を行う必要があります。

開発の有効化

1. Excelを起動する
2. 左下のオプションをクリック
3. リボンのユーザー設定を選択
4. リボンのユーザー設定内にある開発にチェックを入れる

Visual Basic Editorの表示

1. 開発タブをクリック
2. `Visual Basic`をクリック
3. 挿入から標準モジュールを選択

プロシージャの作成

```
Sub マクロ名 ()

End Sub
```

`Sub`と`End Sub`の間に処理を記述します。
実行するには開発タブ内のマクロを選択します。
Visual Basic Editor上で`F5`キーを入力することで実行することも可能です。

- マクロ名で使用できる文字について
  Visual Basic の名前付け規則 （閲覧日：2022/11/30）

セル関連

セルの選択（絶対参照）

- セルA1選択するだけ

```
Range("A1").Select
```

- セルA1からC3とD5を選択する

```
Range(A1:C3,D5).Select
```

Rangeでの選択は複数のセル選択が可能です。
次ページの文字入力や色変更にも使用できます。

セル選択（相対参照）

数値を用いてセルを選択します。Forによる繰り返しに向いています。基準はA1です。

```
Cells(上下,左右)
```

- セルA1を選択する

```
Cells(1,1).Select
```

- セルC5を選択する

```
Cells(5,3).Select
```

- セルF8から下に変数X、右に変数Y移動したセルを選択する

```
Cells(8+X,6+Y).Select
```

移動させる値がマイナスになれば反対方向に動きます。

セル内の変更する

- セルA1に数値を入力する

```
Range("A1")=100
```

```
Cells(1,1)=100
```

- セルB3に文字列テキストと入力する

```
Range("B3")="テキスト"
```

```
Cells(3,2)="テキスト"
```

- セルC4の色を変更する

```
Range("C4").Interior.ColorIndex=色コード
```

```
Cells(4,3).Interior.ColorIndex=色コード
```

- セルD5の文字の色を変更する

```
Range("D5").Font.ColorIndex=色コード
```

```
Cells(5,4).Font.ColorIndex=色コード
```

色コードについて

黒	赤	明るい緑	青	黄	緑
1	3	4	5	6	10

Rangeによるセルの複数選択に対応しています。

シート関連

初期シート以外を選択する際は事前にシートを追加しておく必要があります。
マクロは指定がない限りExcelで現在表示されるシート上での実行となります。

- シート内セルの全クリア

```
Cells.Delete
```

シートの選択（移動）

- `sheet2`へ移動する

```
Worksheets("sheet2").Select
```

- `sheet2`のセル`A1`を選択する

```
Worksheets("sheet2").Range("A1").Select
```

前述の色変更などが利用できます。

よくあるエラー（シート関連）

```
実行エラー'9':
インデックスが有効な範囲にありません。
```

→存在しないワークシートを選択しようとしていませんか？

計算

基本

計算結果を表示する場所の指定を忘れないようにしましょう。

- セルA1に計算結果を表示する

```
Range("A1")=1+2
```

```
Range("A1")=3-4
```

```
Range("A1")=5*6
```

```
Range("A1")=7/8
```

- セルA1にB2とC4の計算結果を表示する

```
Range("A1")=Range("B2")+Range("C4")
```

セル同士計算でも四則計算同様の演算子を利用します。

- その他の演算子

商（整数部）	商（余り）	べき乗
\	Mod	^

シートをまたいだ計算

- 現在のシートSheet1のA1にB1とSheet2のC1を計算した結果を表示する

```
Range("A1")=Range("B1")+Worksheets("sheet2").Range("C1")
```

変数

変数の型について

どのような内容を変数に代入するか**変数を使用する前**に指定する必要があります。

値の形	型	使用RAM	備考
0から255の正の整数	Byte	2B	
-32,768から32,767の整数	Integer	2B	
-2,147,483,648から2,147,483,647の整数	Long	4B	非推奨
±3.4×10^38の少数	Single	4B	
約10×10^7の文字	String	2B	
すべての型	Variant	16B	非推奨

参考：変数の型（閲覧日：2022/12/06）

- 整数を代入する変数の作成

```
Dim 変数名 As Integer
```

- 少数に対応した変数の作成

```
Dim 変数名 As Single
```

- 文字列に対応した変数の作成

```
Dim 変数名 As String
```

    - 変数に使える文字について
      Visual Basic の名前付け規則（閲覧日：2022/11/30）

プロシージャ外で変数の宣言をすると、異なるプロシージャで同じ変数を利用できます。

```
Dim 変数名 As 型
変数名 = 値
Sub Hoge()
    '処理
End Sub
```

変数の使い方

- 数値100を代入する

```
変数名 = 100
```

- 文字列テキストを代入する

```
 変数名 = "テキスト"
```

- セルA1の値を代入する

```
変数名 = Range("A1")
```

- セルA1とB1の値を合計した結果を代入する

```
変数名 = Range("A1") + Range("B1")
```

- 変数の値をセルA1に表示する

```
Range("A1") = 変数名
```

よくあるエラー（変数）

- オーバーフロー

```
実行エラー'6':
オーバーフローしました。
```

→代入できる値の範囲を超えています。より大きな型で宣言し直してください。

- 不一致

```
実行エラー'13':
型が一致しません。
```

→数値の型に文字を代入しようとしていませんか？

条件分岐

基本構文

```
IF 条件式1 Then
    '条件式1に合致したときの処理
ElseIF 条件式2 Then
     '条件式1は合致しないが条件式2に合致したときの処理
Else
    'どの条件にも合致しないときの処理
End IF
```

- 記述時の注意
  - 処理を記述しないと **なにもしない**という処理になります。
  - インデントは視認性のためにTabキーにて挿入しています。
  - ElseIF部は **必須ではありません。** 必要に応じて消したり加えたりしてください。
  - プロシージャ作成のEnd Subは自動入力されますが、**End IFは自動入力されません。**

比較演算子について

条件Aと条件Bが存在するとき

演算子	利用例	意味
=	A = B	AとBは等しい
<	A < B	AはBより小さい
<=	A <= B	AはBと等しいか小さい
>	A > B	AはBよりも大きい
>=	A >= B	AはBと等しいか大きい
AND	A AND B	AとB両方の条件が合致している
OR	A OR B	AとBどちらかの条件に合致している
NOT	NOT A	条件Aに合致しないとき

利用できる算術演算子は計算と同じです。

よくあるエラー（条件分岐）

```
コンパイルエラー:
修正候補:Then または GoTo
```

→条件式末尾のThenが抜けていませんか？

条件式の書き方

条件には数値や文字列、`Range`、`Cells`、変数が利用できます。

- セル`A1`が数値`100`と等しいかを判断

```
IF Range("A1") = 100 Then
```

- セル`A1`の値が`90`以上か判断する

```
IF 90 <= Range("A1") Then
```

- セル`A1`の値が`90`以上`100`未満かを判断する
  悪い例

```
IF 90 <= Range("A1") < 100 Then
```

  正しい式

```
IF 90 <= Range("A1") And Range("A1") < 100 Then
```

- セル`A1`の値が**偶数ではない**ことの判断（NOTの利用）

```
IF NOT Range("A1") Mod 2 = 0
```

Forによる繰り返し

特定の操作を任意の回数繰り返すことができます。

基本構文

数値に対応した変数を作成する必要があります。

```
Dim i As Byte
For i = A to B Step C
    '繰り返す操作
Next i
```

i	A	B	C
利用する変数	iの開始値	iの終了値	1回のループで増えるiの数

セル選択にはセル選択（相対参照）が便利です。

- セルA1からA100まで下に１ずつセルを選択する

```
Dim i As Byte
For i = 0 To 99
    Cells(1+i,1).Select
Next i
```

- （応用）A1からA10に入力されている数が偶数か奇数かを判断する

```
Dim i As Byte
    For i = 0 To 9 Step 1
        IF Cells(1+i,1) Mod 2 = 0 Then
            '偶数のときの処理
        Else
            '奇数のときの処理
        End IF
    Next i
```

よくあるエラー（繰り返しと条件分岐の組み合わせ）

```
コンパイルエラー:
Nextに対応するForがありません。
```

→End IFを忘れていませんか？

関数のネスト

Forループ内でForループを行います。
内側のループが優先して行われます。
Forループの数だけ数値に対応した変数が必要です。

- A1からE10に入力されている数値が偶数ならセルの色を青にし、奇数なら黄色にする

```
Dim i As Byte
Dim j As Byte

For j = 0 To 4 Step 1
    For i = 0 To 9 Step 1
        If Cells(1+i,1+j) Mod 2 = 0 Then
            Cells(1+i,1+j).Interior.ColorIndex = 6
        Else
            Cells(1+i,1+j).Interior.ColorIndex = 5
        End If
    Next j
Next i
```

i	j
上下（1から10）	左右（AからE）

  - カラーコードについては4ページを参照

繰り返し合計

繰り返し処理の中で計算を行います。

- セルA1からC5のセルで奇数のセルを合計する

```
Dim i As Byte
Dim j As Byte
Dim Goukei As Integer
Goukei = 0
For j = 0 To 2 Step 1
    For i = 0 To 4 Step 1
        If Cells(1+i,1+j) Mod 2 = 0 Then
        Else
            Goukei = Goukei + Cells(1+i,1+j)
        End If
    Next i
Next j
```

  - 合計を代入する変数をループ前に**初期化**する必要があります。（ここでは0）
  - Cells(1+i,1+j)が偶数の場合はなにもしないため、IF部は空白にしています。

その他便利機能

オートフィル

1. 数値のオートフィル
   連続したセルに数値が入力されている必要があります

- A1に100、A2に200と手で入力し、A10まで100・・200・・300と連続した数値を表示する

```
Range("A1:A2").AutoFill Destination:=Range("A1:A10")
```

2. 文字列のオートフィル

- A1に月と入力しA1からA7に一週間分の曜日を表示する

```
Range("A1").AutoFill Destination:=Range("A1:A7")
```

手入力やRange・Cellsによる方法でセルに値を入力してもOK

メッセージボックス

- メッセージボックスで文字列テキストを表示する

```
MsgBox "テキスト"
```

  数値を表示させる際でも""で囲む必要があります。

- メッセージボックスでセルA1の内容を表示する

```
MsgBox Range("A1")
```

- メッセージボックスで変数TEXTの内容を表示する

```
MsgBox TEXT
```

- （応用）メッセージボックスにA1の値は（A1の値）ですと表示する

```
MsgBox "A1の値は" & Range("A1") & "です"
```

  &を挟むことで結合できます。

- （応用）セルA1の内容を表示するメッセージボックスのタイトルA1の中身は？を設定する

```
MsgBox Range("A1"), vbOKOnly, "A1の中身は？"
```

    - メッセージボックスについて
      第23回.メッセージボックス(MsgBox関数)（閲覧日：2022/12/01）

罫線

```
罫線を引く範囲.BorderAround 線のスタイル定数, 線の太さ定数, 色コード
```

- 枠線を引く範囲には3ページのRangeやCellsが利用できます。
- 色コードには4ページのColorIndexが利用できます。
- 線のスタイルについて

罫線の種類	定数
線なし	xlLineStyleNone
一重線	xlContinuous
二重線	xlDouble
破線	xlDash
一点鎖線	xlDashDot
二点鎖線	xlDashDotDot
点線	xlDot
斜破線	xlSlantDashDot

- 線の太さについて

太さ	定数
極細線	xlHairline
細線	xlThin
中太線	xlMedium
太線	xlThick

- セルC1の周りに赤色の細い破線を引く

```
Range("C1").BorderAround xlDash, xlThin, 3
```

- セルC1からG5の周りに青色の太い一重線を引く

```
Range("C1:G5").BorderAround xlContinuous, xlThick, 5
```

参考

FOM出版　よくわかるMicrosoft Excel 2019/2016/2013 マクロ/VBA

Visual Studio Code拡張 Markdown All in One（閲覧日：2022/11/30）

Qiita マークダウン記法 一覧表・チートシート（閲覧日：2022/11/30）

Visual Basic の名前付け規則（閲覧日：2022/11/30）

変数の型（閲覧日：2022/12/06）

エクセルの真髄 第23回.メッセージボックス(MsgBox関数)（閲覧日：2022/12/01）

VScodeのMarkdownからPDF変換時に改ページを挿入（閲覧日2022/12/01）

このノートについて

- 作成： **matsukz**
- レポジトリURL： https://github.com/matsukz/3-1ClassNote
- 最終更新日： **2023年01月18日**
- 皆伝レベル： **1**
- 初音ミクのキャラクターランク： **63**
- 彼女： **なし**