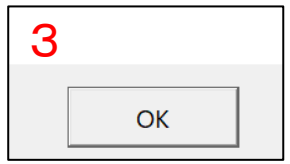
変数の宣言

Dim a As Integer　　整数（ Integer）の値が入る変数aを準備する

a=5は、aに5を代入するという意味

```
Sub Data()
        Dim a As Integer
        Dim b As Integer
        a=2
        b=3
        a=b   ←aにbの値を代入するので、最初入っていた2を3に置き換えとなる

        MsgBox a   ←メッセージボックスに　aを表示
End Sub
```



データの型	データの型名	入る値の範囲
バイト型	Byte（バイト）	0～255
整数型	Integer（インテジャー）	－32768～32767
長整数型	Long（ロング）	－2147483648～ 2147483647
単精度浮動 小数点型	Single（シングル）	$\pm 3.4 \times 10^{38}$～ $\pm 1.4 \times 10^{-45}$
倍精度浮動 小数点型	Double（ダブル）	$\pm 1.8 \times 10^{308}$～ $\pm 4.9 \times 10^{-324}$

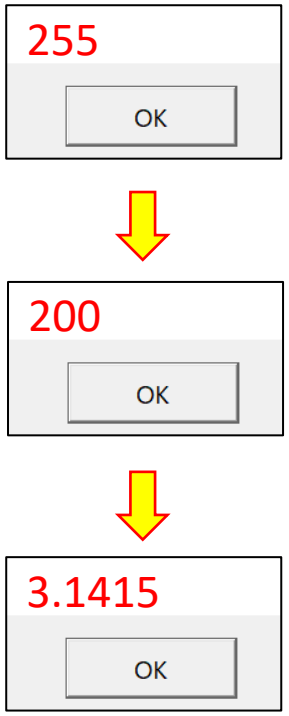
容量が2倍

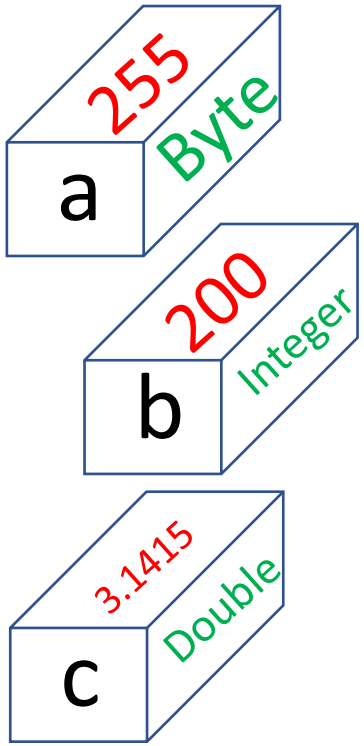
2^8

$\pm 2^{16}/2$

$\pm 2^{32}/2$

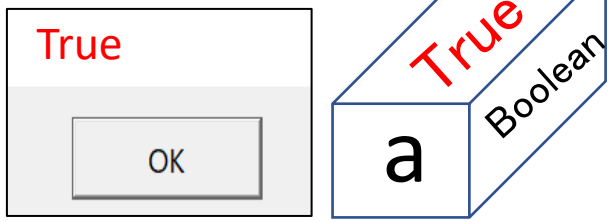
```
Sub NumData()
        Dim a As Byte
        Dim b As Integer
        Dim c As Double
        a=255
        b=200
        c=3.1415

        MsgBox a
        MsgBox b
        MsgBox c
End Sub
```



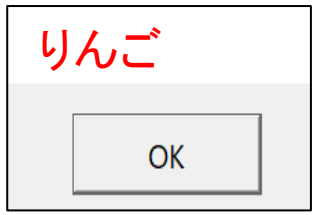


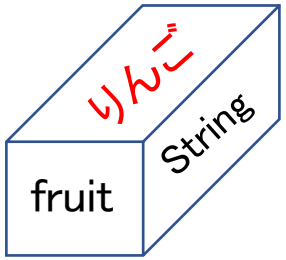
データの型	データの型名	入る値の範囲
論理型	Boolean（ブーリアン）	True, False
文字列型	String（ストリング）	
日付型	Date（デイト）	西暦100年1月1日～ 9999年12月31日
ｵﾌﾞｼﾞｪｸﾄ型	Object（オブジェクト）	

```
Sub BoolData()
    Dim a As Boolean

    a=True

    MsgBox a
End Sub
```




```
Sub StringSample()
    Dim fruit As String

    fruit = “りんご”

    MsgBox fruit
End Sub
```



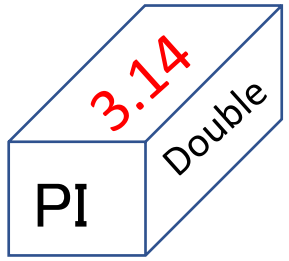


定数

```
Const PI As Double=3.14
```

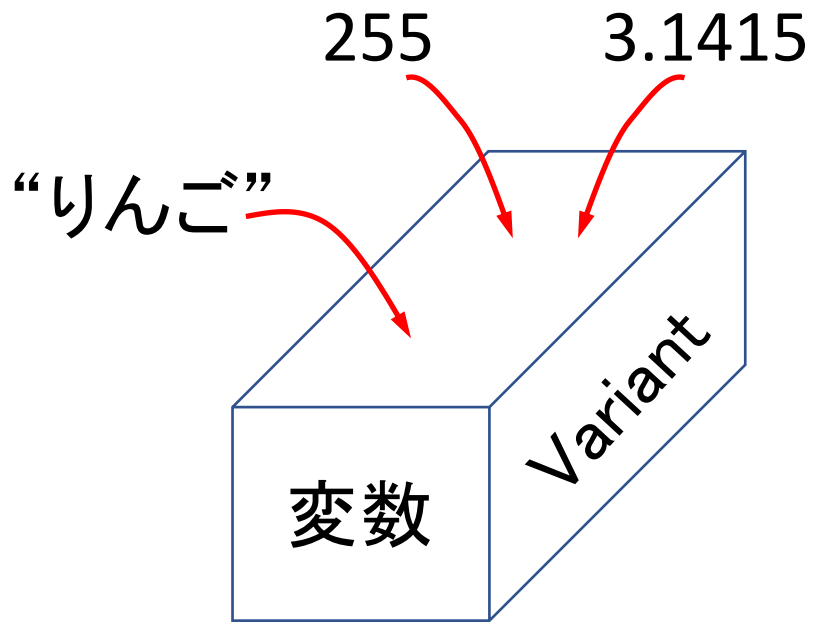
定数名　データ型　値

一度上記のように定義すると
PI=3.14は代入不可

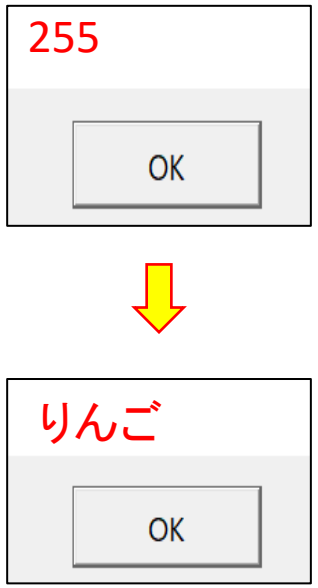


データの型	データの型名
バリアント型	Variant（バリアント）

宣言していない変数を使用したり、データ型を省略して変数を宣言できる




```
Sub VarData()
    Dim a

    a=255
    b=”りんご“

    MsgBox a
    MsgBox b
End Sub
```



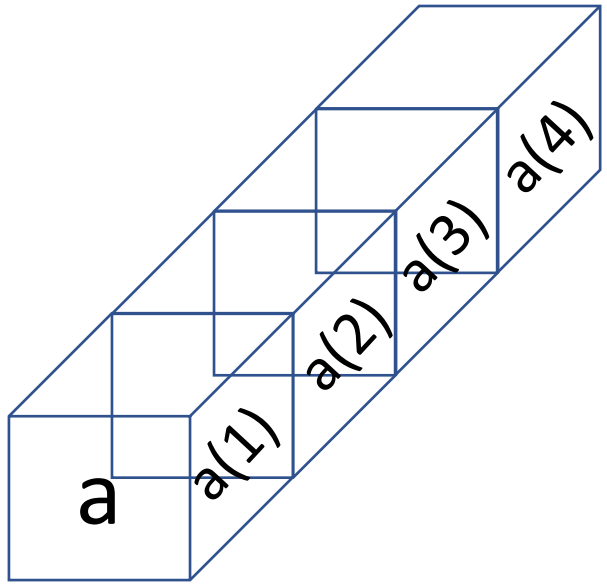
デメリット

①変数の中身がわかり難い
②宣言していない変数を間違える
③メモリを多く使うため、実行速度が遅くなる

配列

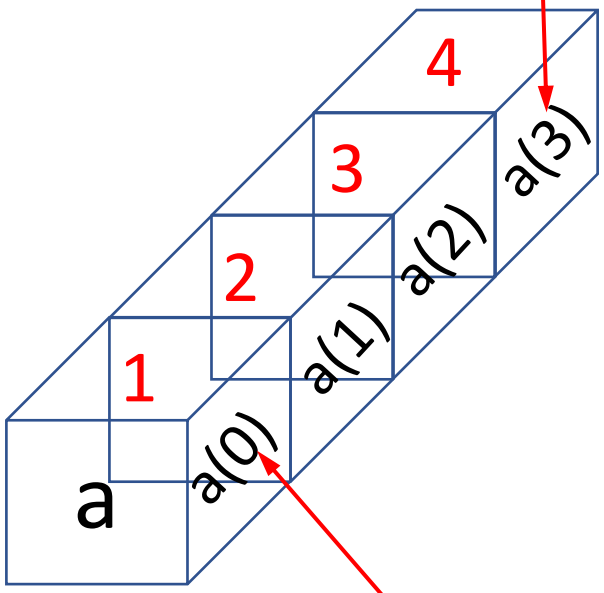
Dim a(1 to 4) As Integer

配列名　　インデックス番号の範囲　　データ型



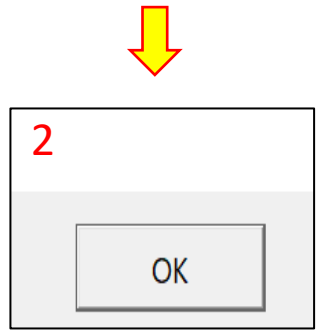
Dim a(3) As Integer

インデックス番号の最大値

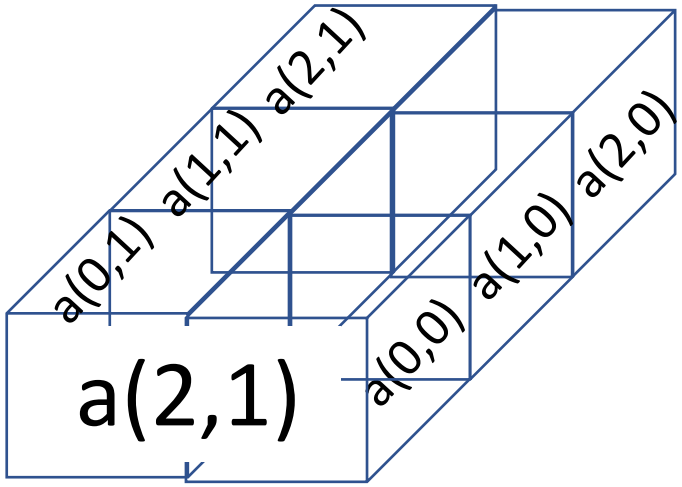


指定なければ、インデックス番号は0から

```
Dim a(3) As Integer
Dim n As Integer
n=1

a(0)=1
a(1)=2
a(2)=3
a(0)=4

MsgBox a(n)
End Sub
```



2次元配列



Dim a(0 to 2, 0 to 1) As String

配列名

データ型

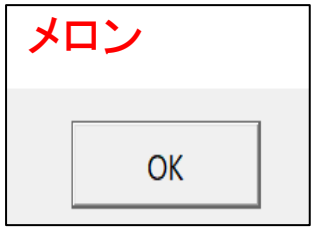
行方向インデックス番号の範囲

列方向インデックス番号の範囲

Dim a(2 , 1) As String

```
Sub TDArray()
    Dim a(2, 1) As String
    a(0,0)=”りんご”
    a(1,0)=”みかん”
    a(2,0)=”いちご”
    a(0,1)=”バナナ”
    a(1,1)=”メロン”
    a(2,1)=”スイカ
    Range(“A1:B3”)＝a
    MsgBox Range (“B2”).Value
End Sub
```

	A	B
1	りんご	バナナ
2	みかん	メロン
3	いちご	スイカ



配列の要素数変更

ReDim

配列未設定

Dim a() As String

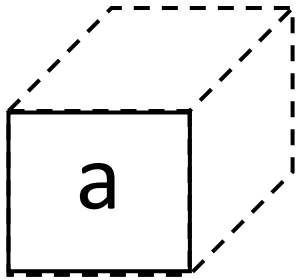
ReDim a(1) ←ｲﾝﾃﾞｯｸｽ番号の最大値が1

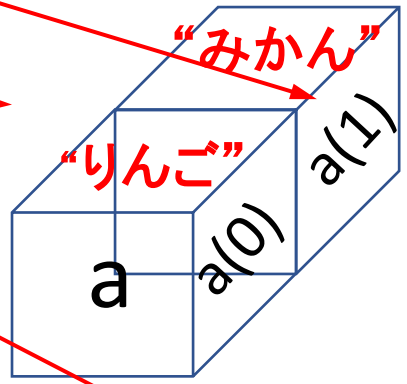
a(0)=“りんご”
a(1)=”みかん“

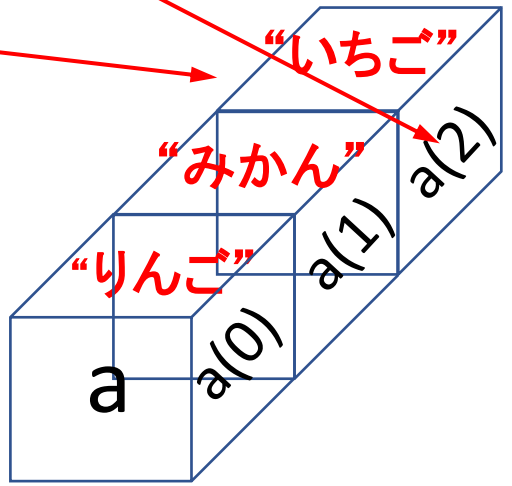
ReDim Preserve a(2) ←ｲﾝﾃﾞｯｸｽ番号の最大値が2

a(2)=“いちご”

元のデータを保持したまま







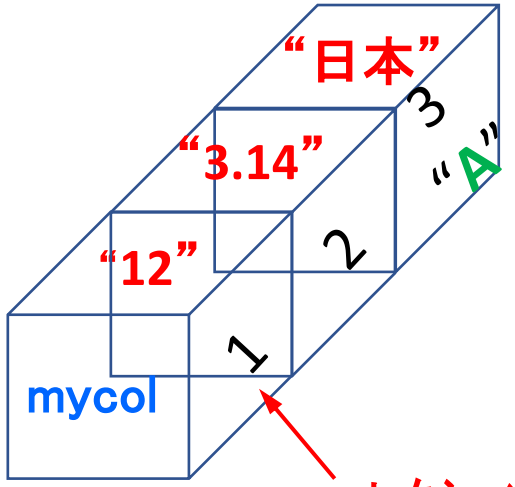
コレクション　関連データを1つにまとめて管理する配列

mycol

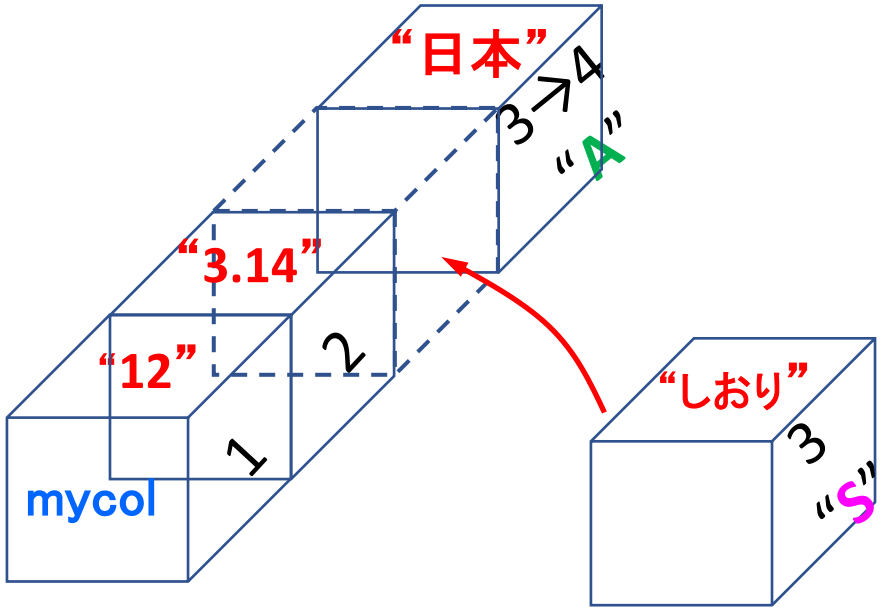
```
Dim mycol As New Collection

mycol.Add Item:=12
mycol.Add Item:=3.14
mycol.Add Item:=”日本“　, Key:=“A”

mycol.Add Item:=”しおり“　, Key:=“S” , before :=“A”
        または
mycol.Add Item:=”しおり“　, Key:=“S” , after :=“2”
```

目印

追加はAdd



コレクションでは1から始まる



コレクション要素の取得

mycol.Item(7)　または　mycol.Item(“A”)

省略してmycol(7)　または　mycol(“A”)でも可

コレクション要素の削除

mycol.Remove(7)　または　mycol. Remove(“A”)

コレクション要素数を求める

```
Dim mycol As New Collection

mycol.Add Item:=3
mycol.Add Item:=5

Dim n As Integer= mycol.Count
```