＜論　　説＞

# 文系学生のためのVBAプログラミング教育についての考察

五月女　仁　子

## 1　はじめに

パソコンの普及や，若い世代への情報教育の実施に伴い，大学での情報教育にプログラミングを導入する大学も増えている。従来の大学教育で行われるプログラミングは，理系の学生を対象にしたもので，採り上げられるプログラミング言語もFortran, Pascal, Cなどが中心であった。これらのプログラミング言語は，エディタにプログラミングを入力→コンパイル→実行→実行結果を見るという形式である。しかし，最近ではWindows OSやインターネットの普及もあって，視覚的にとらえられるもの，インターフェースを意識したプログラミングを採り上げる傾向が強い。

筆者は，神奈川大学経済学部でプログラミング講義を担当して5年になる。2008年度から2009年度までの2年間はプログラミングについて全く初めての学生を対象としたが，それ以外は何らかのプログラミング言語を使った講義を受けたことがある学生を対象としている。プログラミング言語は，前期Excel VBA，後期Visual Basicを行っている。前期に講義するExcel VBAについて，学生が躓く箇所を指摘しながら講義を進めるうえでのポイントを考察する。

## 2　VBAについて

### (1) VBAとは

VBAとは，Visual Basic For Applicationの略で，Microsoft社が，同社のOffice各製品の機能を拡張するために開発したプログラミング言語である。本講義では，Office製品のうちExcelを対象とするVBAを扱った。VBAはExcelの他Accessが有名であるが，WordやPower Pointでも開発言語となっている。名称の前にVisual Basicとあることからわかるように，Visual Basicがベースである。Visual Basicは1991年にMicorosoft社がWindows上で動作するアプリケーションを作成する言語として発表したもので，初心者向けの言語であったがバージョン6を最後に初心者向けと言えないほど難しくなった。

### (2) Excel VBAの利点

VBAの利点は以下のとおりである。Excelを使って

- アプリケーション上で繰り返し行う操作の自動化が可能となる
- オリジナルの関数が作成できる
- オリジナルのダイアログボックスが作成できる
- オリジナルのアプリケーションが作成できる

Excelには関数が多数用意されているので，それでも十分であると思われがちである。例えば，毎月の社員の給与明細を作る場合，その月の働いた日数，そして働いた日数分の交通費はExcelの関数でもできるが，それを集計し，グラフに表示し，できた表とグラフを保存し，給与明細書として印刷するという一連の操作は，Excelの関数だけでは不可能で，VBAで行う必要がある。

### (3) Excel VBAを採り上げた理由

プログラミング言語としてVBAを採り上げた理由は以下のとおりである。

- Excelアプリケーションについての操作が1年生の講義で学習済みであること

Excelについての画面の操作，関数の使い方や種類に慣れているため，VBAはこれを一歩すすめて便利にするためのものという考え方が受け入れやすい。

- 対象が文系学生ということ

将来開発者になるということは前提とせず，将来仕事についた場合，アプリケーションを便利にするツールの作成が可能となることを目指す。

- VBAのコーディングの特徴

プロパティやメソッドの候補がピリオドを打つことによって，プルダウンでメニューに現れるので，全ての命令を入力する他のプログラミング言語よりも，入力ミスが少ない。文法上のエラーもプログラム入力中にエラーメッセージが出るので，どこが間違っているのかがわかりやすい。また，ユーザーフォームが作成できるので，インターフェースを作成するプログラミングの考え方の指導がしやすい。

## 3　プログラミングの実行環境

プログラミングの実行環境については，OSはWindows XPで，Excelは昨年よりMicrosoft Office Excel 2007を使用している。

## 4　プログラミングの講義内容

筆者担当は，昨年までプログラミングの基礎ということで，表1のような講義計画で行われていたが，学生が躓く箇所ではゆっくり講義をすすめるため，11回目位の講義計画の内容で終了していた。本年度はプログラミングの応用であったが，学生へのアンケートでは，復習をして欲しいとの要望が多かったため，1，2回目の内容は省き，変数，文法，配列，Subプロシー

ジャ，Function プロシージャ，インターフェースの作成とすすめた。インターフェースの作成が例年よりも，少し完成度が高いものができたものの，講義計画の 15 回目の内容までは入れなかった。

表 1

| 回数 | 内　　容 |
|---|---|
| 1 | マクロとは<br>VBA とは，VBA の特徴<br>記録マクロを使ったマクロの作成と修正 |
| 2 | VBE の説明<br>VBA を作ってみよう |
| 3 | Sub プロシージャの構成<br>フォント操作<br>変数 |
| 4 | セルの絶対参照と相対参照 |
| 5 | 条件分岐 1 If ステートメント |
| 6 | 条件分岐 If ステートメント応用<br>SelectCase ステートメント |
| 7 | 繰り返し処理（For ステートメント） |
| 8 | 繰り返し処理（Do ステートメント） |
| 9 | Sub プロシージャ |
| 10 | Function プロシージャ |
| 11 | 配列の基礎 |
| 12 | 配列の応用 |
| 13 | インターフェースの作成 1 |
| 14 | インターフェースの作成 2 |
| 15 | 印刷操作<br>Sheet 操作<br>Excel の関数の呼び出し |

## 5　講義のポイント

学生はどのようなところで躓くのか，初心者を対象とした去年までの 2 年間と経験がある学生を対象とした今年のような場合は，画面操作のような初めの段階で躓くかどうかの違いはあるが，それ以降の変数や文法を扱う点では，それほどの差は感じられなかった。学生が間違えやすい箇所を指摘しながら，プログラミングにおける講義ポイントとなる箇所を考察する。

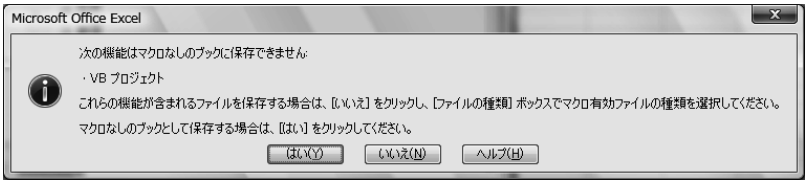
### (1) VBE について

VBA は Excel を立ち上げて，プログラミングを入力する VBE（Visual Basic Editor）を表示し，そこにプログラミングを作成する。全くの初心者の場合，この画面を出すことやこの画面の操作に慣れることに一苦労する。しかし今年は，そのようなことはなかった。

### (2) ファイルの保存と呼び出しについて

Microsoft Office のバージョンが 2007 となり，それまでのバージョンと違い，保存する際に「Excel マクロ有効ブック」を選択して保存するようになった。この点を忘れて図 1 のメッセージを出してしまう学生が見られた。ただ，初めの講義で指摘すれば，メッセージで気づいてやり直す学生がほとんどのためそれほど問題はなかった。

図 1




次に既存ファイルの呼び出しについてであるが，これもバージョンが 2007 になってから，ファイルを呼び出した際，図 2 のようなセキュリティ警告が出るようになった。このオプションボタンをクリックすると，図 3 の画面が表示され，「このコンテンツを有効にする」を選択し，OK ボタンをクリックする。

図 2

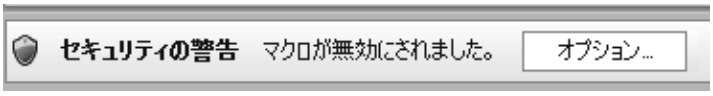
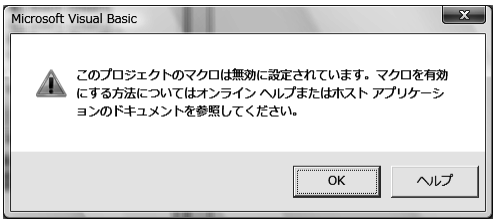



図 3


図 2 の［オプション］ボタンをクリックし忘れて，プログラムを実行しようとすると，図 4 のようなエラーメッセージが出るため，もう一度，ファイルの呼び出しからしなければならない。しかし，この点もメッセージが出ればやり直す学生がほとんどのため問題はなかった

図 4



### (3) 入力について

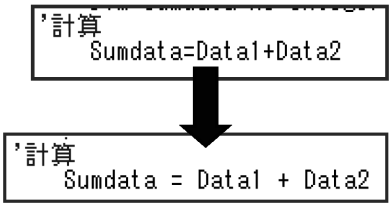



図 5

プログラミング作成時に「どこで空白を入れていいかがわからない」という質問を受ける。計算などでは，演算子と変数の間に自動的に空白があく（図 5）が，基本的には命令 1 つ，変数 1 つについてその前後に空白を入れる形で作成していく。この質問が出るのは，1 つ 1 つの命令を覚えていないためと考えられる。ただ渡されたものを入力するだけで済ませていると，この状態になる。

### (4) メソッドとプロパティに関して

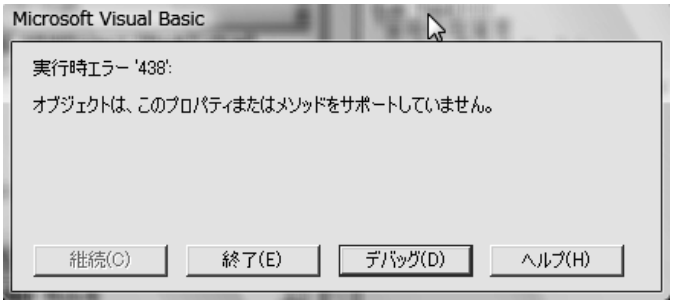



図 6

オブジェクトで使えないメソッドやプロパティを使うと，図 6 のようなエラーが出る。VBA の場合，オブジェクト名の後にピリオドを入力すると，そのオブジェクトで使用できるメソッドやプロパティのメニューがプルダウンで出る（図 7）。メニューから該当するものをダブルクリックすればよいので，メソッドやプロパティの単なる入力ミスは少ない。これは，その前に入力したオブジェクトの入力ミスの場合が多い。オブジェクトを間違っていて，ピリオドを入力しても表示されるメニューの中に該当するメソッドやプロパティがないにもかかわらず，そのまま入力してしまう。この点，ピリオドを入力してメニューに該当するものがなければオブジェクトをよく確めるように注意した。

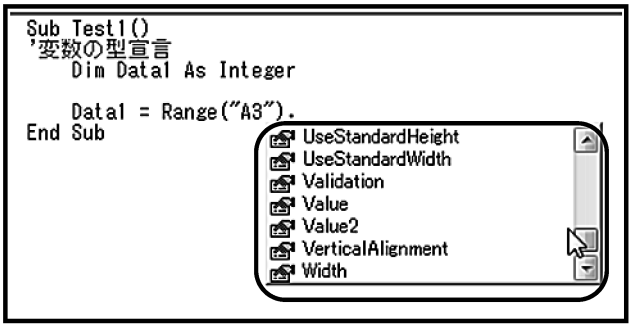



図 7

ただオブジェクトを Selection とすると，プルダウンのメニューが出ないので，この場合は入力ミスがみられる。

### (5) 予約語について

Dim や As, Range などの予約後の入力ミスがある。これは，このような予約語を小文字で入力

することを指導することによって改善された。小文字で入力し，スペル，用法があっていれば，カーソルを違う行に移動すると最初の文字が大文字に代わる（図8）ので，間違いに気づきやすい。

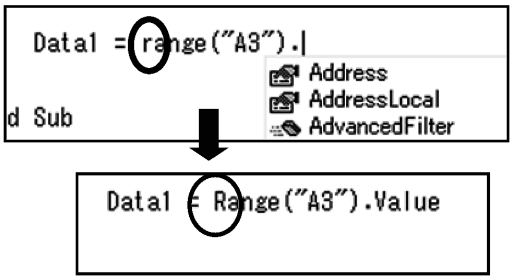



図8

### (6) 変数名の間違い

変数名を間違えても，エラーメッセージは出ない。この点，変数名を宣言する際に，最初の文字を大文字にするように指導した（図9）。例えば，data1という変数の場合，初めの1文字を大文字にして，Data1と宣言する。そして，変数Data1を入力する際，わざと小文字でdata 1と入力するようにして，Enterキーを押すと，変数名が正しければ最初の1文字目が大文字に代わり確かめられる（図9）。

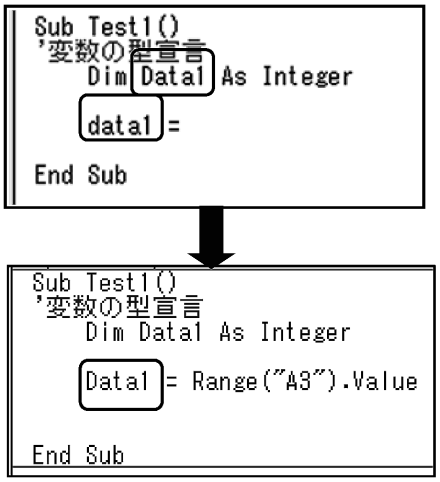



図9

また，「Option Explicit」を宣言するようにも指導した。これを宣言すると，変数を間違えた場合，実行時にエラーメッセージが出るようになる（図10）。

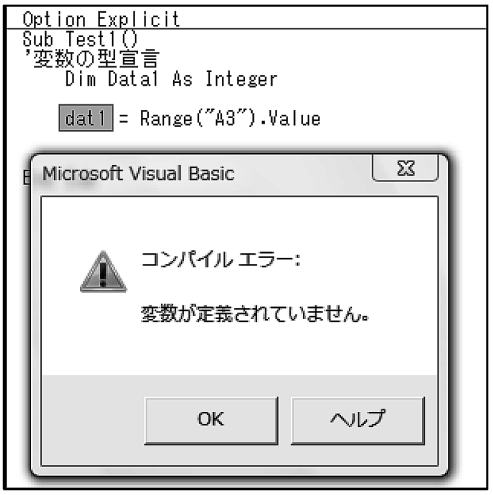



図10

### (7) セルの指定について

セルの指定方法として，Range（“A4”）というように直接セルA4を指定する方法もあるが，

繰り返し処理や，配列を想定すると Offset や Cells でセルを表す方法を知っておくほうが便利である。これについてのミスは以下のとおりである。

### ① Offset に関するミス

Offset は，カレントセルとなっているところから相対的に行と列をどのくらいずらすかを指定する。行の場合は右側がプラスで，左側がマイナスになり，列の場合は下側がプラスで，上側がマイナスになる。図 11 では，A はセル D6，B はセル A3 に表示される。

```
Range("C4").Select
Selection.Offset(2, 1).Value = "A"
Selection.Offset(-1, -2).Value = "B"
```

図 11

行と列を逆にとらえているミスが最も多く，次に列についてプラスとマイナスを逆にとらえているミスが多い。また，Offset でずらしたセルをカレントセルになっていると勘違いするミスも多い。

Offset の Value プロパティでは値を表示したり，値を取得することはあっても，そのセル自体はカレントセルになるわけではない。そのため図 11 でセル D6 に A と表示されたとしてもカレントセルはセル C4 である。次の Offset（−1, −2）はこのカレントセルであるセル C4 から上に 1 行，左に 2 列ずれたセル A 3 となる。

Offset に関するミスは，指定された行や列が存在しない場合はエラーメッセージが表示されるが，それ以外は表示がないため結果が表示され，違うことに気づいてから修正する形になる。

繰り返し処理とともに，Offset を使用する場合，どこにカレントセルがあるのかがわからなくなる学生が多く，Offset を指定できない学生が多かった。どこにカレントセルがあるかを意識して繰り返し処理をするように指導した。

### ② Cells に関するミス

Cells は行と列を直接指定する絶対表示の形式であるが，「Range（“C 4”）」を Cells で表すと「Cells（4, 2）」と表すように Range とは行と列が逆になる。そのため Cells についても行と列を反対にとらえてしまうミスが多い。

### (8) 文法について（条件分岐 If ステートメント）

If ステートメントの構文は右の構文 1 のとおりである。

構文 1

**If ステートメント**

```
If　条件 1　Then
　　条件 1 が真の場合の処理
ElseIf　条件 2　Then
　　条件 1 は偽で、条件 2 が真の場合の処理
・・・
・・・
Else
　　条件がすべて偽の場合の処理
EndIf
```

学生が多く間違える箇所を以下の 2 つの例をあげて説明する。正解は右図に示す。

（ア）点数（変数名：Ten）が 80 点以上の場合「A」，80 点未満 70 点以上の場合「B」，70 点未満の場合「C」と判定（変数名：Hantei）する（図 12）。

図 12

```
If Ten >= 80 Then
    Hantei = "A"
ElseIf Ten >= 70 Then
    Hantei = "B"
Else
    Hantei = "C"
End If
```

（イ）点数が 80 点以上の場合「合格」と判定する（図 13）。

図 13

```
If Ten >= 80 Then
    Hantei = "合格"
End If
```

**①比較演算子のミス**

（ア），（イ）に共通に考えられるミスとしては，比較演算子のミスが考えられるが，本講義では，1 年生で Excel の指導を受けてきているため，このミスは少ない。

**②入力忘れ**

Then, ElseIf, Else, EndIf の構文の入力を忘れてしまうミスは多い。例（ア），（イ）に共通にみられるミスであるが，プログラム作成時に，図 14 のようなエラーメッセージが表示されるため，学生は自分で気づく場合が多い。


図 14

**③ 2 つ目以降の条件作成時のミス**

例（ア）の場合のミスとして，2 つ目以降の条件作成時のものが非常に多い。条件 2 の「80 点未満 70 点以上の場合 B」の条件を「ElseIf 80＞Ten＞＝70 Then」とする。これはエラーメッセージが表示されないが，正しい結果は出ない。また，正しい結果とはなるが「ElseIf 80＞Ten And Ten＞＝70 Then」とする学生も多い。これらは If の構造がわかっていないためと思われる。

筆者は，If ステートメントを説明する場合，フローチャート（図 15）を使って説明するようにしている。フローチャートの記号の意味は説明に必要な記号だけわかってもらえればよいので，条件がひし形で，出力が左の先のとがった楕円であることを説明し，(a)，(b)，(c)，(d) には，どのような人が来るのか，条件はどう置けばよいかを解説する。

ここでは，「(b) に来る人はどのような人か」それは「全員が 80 点未満の人たち」なので，条件 2 はその中で 70 点以上という条件式を作ることになると解説する。

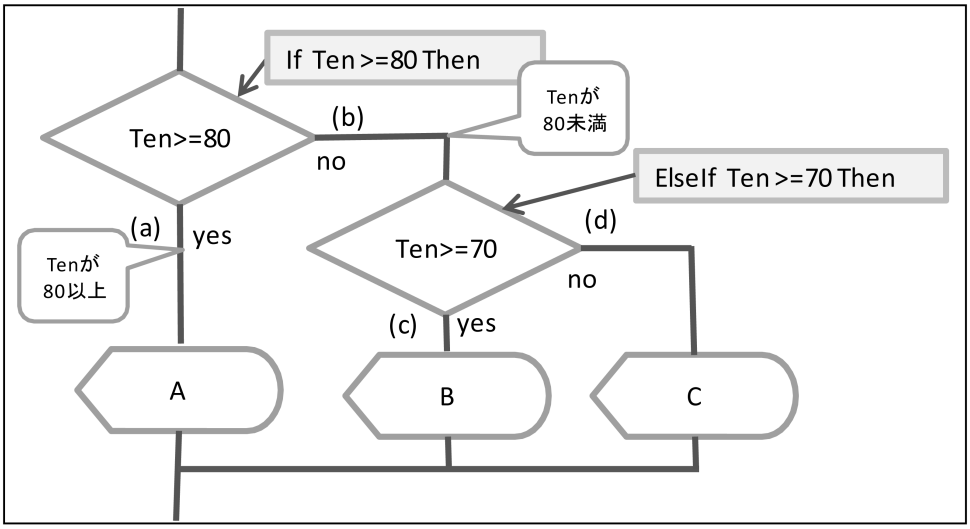



図 15


図 16

**④最後の Else のミス**

例（ア）の場合のミスとして，「Else Ten＜70」，「Else Ten＜70 Then」とする学生が多い。これは，プログラム作成中に，図 16 のようなエラーメッセージが出るが，何が違っているのか気づきにくい。図 17 のフローチャートを使い，「(c) に来る人はどのような人か」，それは「80 点未満ではあるが 70 点以上とれた人」であり，「(d) に来る人はどのような人か」それは「70 点未満の人である」と解説し，Else の後の条件を置かないことを指導する。

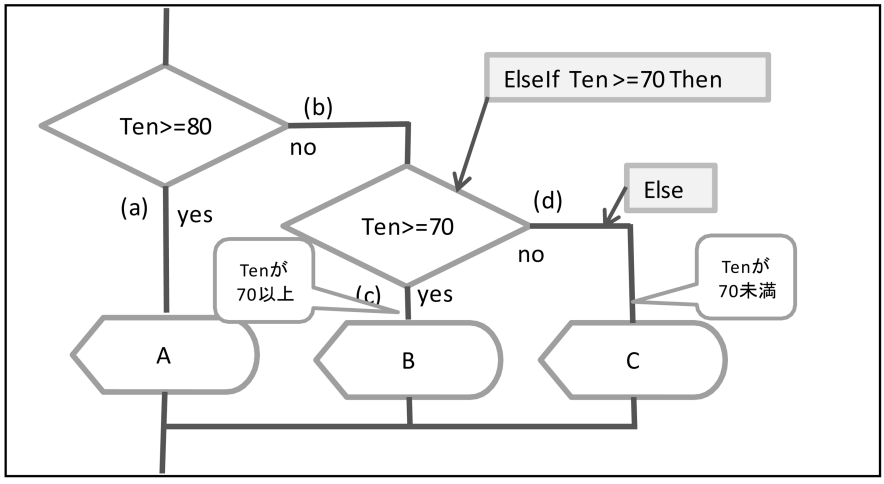



図 17

例（イ）の場合のミスとして，Else を書かなくてもよいのに書いてしまって，EndIf を書き忘れている学生が多い。これについては，事前に例題を示し，それ以外がない場合は Else は書く必要がなく，直接 EndIf を書くことができることを示す。

### (9) 文法について（繰り返し処理 For ステートメント）

Excel の VBA であることから，表を扱うことが多い。そのため，For ステートメントは，表を想定して 2 重ループまで取り組んでいる。

**For ステートメント**
For カウンタ=初期値 To 終了値 Step 増分値
　　・・・・
Next　カウンタ

**構文 2**

#### ①増分値のミス

増分値が 1 である場合は「Step 1」を省略できる。はじめに習う例題として，省略されるケース（例えば 1 から 10 までの和を求めるもの）を扱っているため，Step を使うことに慣れず，増分値が 2 や 10, または −1 とするようなケースの場合，増分値に数がおけないで困っている学生が見られた。何題か Step 増分値を利用する課題を挑戦し練習した。

#### ②二重ループのミス

外ループを行，内ループを列として二重ループを取りあげたが，図 18 左の表のケースで考えると行が 10 回のループ，列は 5 回のループとなり，正解は図 18 右になるが，以下にあげるミスが見られた。

| | A | B | C | D | E | F | G |
|---|---|---|---|---|---|---|---|
| 1 | 商品名 | 月 | 火 | 水 | 木 | 金 | 合計 |
| 2 | 商品名1 | 92 | 52 | 63 | 85 | 73 | 292 |
| 3 | 商品名2 | 94 | 70 | 51 | 81 | 51 | 296 |
| 4 | 商品名3 | 90 | 98 | 99 | 97 | 69 | 384 |
| 5 | 商品名4 | 62 | 63 | 85 | 91 | 96 | 301 |
| 6 | 商品名5 | 74 | 74 | 72 | 79 | 96 | 299 |
| 7 | 商品名6 | 83 | 57 | 71 | 55 | 78 | 266 |
| 8 | 商品名7 | 65 | 71 | 72 | 99 | 66 | 307 |
| 9 | 商品名8 | 98 | 78 | 98 | 86 | 94 | 360 |
| 10 | 商品名9 | 59 | 69 | 72 | 96 | 84 | 296 |
| 11 | 商品名10 | 56 | 62 | 83 | 63 | 99 | 264 |
| 12 | | | | | | | |

```
For i = 2 To 11
    Sum = 0
    For j = 2 To 5
        Kosu = Cells(i, j).Value
        Sum = Sum + Kosu
    Next j
    Cells(i, j + 1).Value = Sum
Next i
```

**図 18**

(a)　外ループと内ループのカウンタを同じにしてしまっている（図 19 左）

これは，図 19 右のようなエラーメッセージが出るので，対処されやすい。

```
For i = 2 To 11
    ・・・
    For i = 2 To 5
        ・・・
    Next i
    ・・・
Next i
```

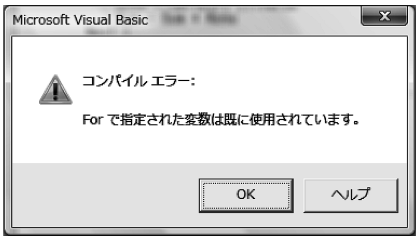



図 19

(b)　For～Next の対応があっていない（図 20 左）

これは，右のようなエラーメッセージが表示されるが，Next 忘れでも同じメッセージが表示されるので気づきにくい。ここでは，外側の For～Next と内側の For～Next でインデントを変え対応がわかるように指導した。

```
For i = 2 To 11
    ・・・
    For j = 2 To 5
        ・・・
    Next i
    ・・・
Next j
```


図 20

(c)　Next を忘れる（図 21）

これは上記と同じエラーメッセージがあるのでわかりにくいが，For の数と Next の数の違いを指摘することで対処される。

```
For i = 2 To 11
    ・・・
    For j = 2 To 5
        ・・・
Next i
```


図 21

(d)　行 10 回と列 5 回のループにしていない

これはエラーメッセージとして表示されず，結果が違うことで対処しなければならない。

(e)　For i＝1 To 10，For j＝1 To 5 とした場合，対応する Cells(i, j) を Cells(i＋1，j＋1) とずらしていない

これもエラーメッセージとして表示されず，結果が違うことで学生から手があがる箇所である。

**③二重ループを使った合計計算のミス**

図18左表のG列は，各商品の合計が求められている。図18右のプログラムでは，変数Sumに合計を求めているが，プログラム2行目「Sum＝0」が重要である。この「Sum＝0」と置けない学生が多い。VBAの場合，宣言をした段階で変数は0とされているが，このように変数を使いまわして計算する場合，「Sum＝0」としなければ，この場合は累計になってしまうことを説明した。

**（10）文法について（繰り返し処理 Do ステートメント）**

Do ステートメントについては，2つの条件を置くケースを講義した。While の場合（構文3）は条件が成立している間繰り返し，Until の場合（構文4）は条件が成立したら繰返しを終了するという違いがある。For ステートメントはデータ数がわかっている場合に使うことが想定されるが，Do ステートメントはデータ数がわからないことを想定して使う場合が多い。実際に仕事でデータを扱う場合，正確なデータ数がわかっていることは少ないので，Do ステートメントの出番は多い。講義では，IsEmpty 関数も併せて講義した。IsEmpty 関数は該当セルが空白かどうかを判定する関数である。実際のデータは，キーとなる学籍番号や，商品コードが入るセルは途中に空白があるということはないので，そのセルに IsEmpty 関数で使い，空白が判定されたらそこでデータが終わりと考えて条件を作った。

```
Do ステートメント 1
Do　While　条件
　　・・・・
Loop
```

構文 3

```
Do ステートメント 2
Do　Until　条件
　　・・・・
Loop
```

構文 4

ここでのミスは以下のとおりである。

**①　Until の条件と While の条件を反対にしているミス**

条件の設定を間違え，無限ループに入ってしまう学生が多い。プログラムを実行前に必ず保存するように指導し，無限ループに入ってもまたやり直しができるようにする。

**②　条件が成立する（成立しなくなる）ような処理をループの中で行っていないミス(図22)**

これは一度もループに入れないか，無限ループになってしまうミスで，図22の例では網かけ部分を作成しないケースである。プログラムを実行する前に必ず保存するとともに，For ステートメントと違い自動的にカウントされることはないため，Do ステートメントの中で何かしら条件を成立する，または成立しなくするようなプログラムが必要であることを指摘する。図22では，条件としてiを使っているので条件を

```
Sub Test8( )
    Dim i As Integer
    i = 0
    Do While i < 10
        i = i + 1
        Cells(10, i).Value = i
    Loop
```

この部分がない

図 22

成立しなくなるようにiを変化させるプログラム「i=i+1」が必要である。

### (11) 配列について

配列の場合，厳密には違うが，変数の説明とあわせて，「同じ名前の箱をメモリの中に作り，同じ名前だと区別がつかないので，番号（添え字）をふる」と説明している。その考え方をとらえられるかが，まず第一歩となる。講義では1次元配列を中心に説明している。

#### ①宣言についてのミス

そのままの宣言では，添え字は0から始まるので，もし100個の配列（配列名：Data 1，整数型）を作成する場合は「Dim Data 1(99) As Integer」となるが，「Dim Data 1(100) As Integer」とする学生が多い。ただ多めに配列が用意される分には，エラーにはならない。このような設定をする学生の多くは，添え字を1からはじまると考えている場合が多いので最初のデータ Data 1(0)を計算していないミスが見られ（図23）る。

```
'計算
    Sum = 0
    For i = 0 To 99
        Sum = Sum + Data1(i)
    Next i
```

⇩

```
'計算
    Sum = 0
    For i = 1 To 100
        Sum = Sum + Data1(i)
    Next i
```

図 23

#### ②添え字についてのミス

配列を使って計算をする場合，配列の表記は図23のように「Data 1(i)」となるのであるが，それが表記できず「Data 1」としてしまうミスが多い。

### (12) Sub プロシージャと Function プロシージャについて

Sub プロシージャは下記のような構文で，呼び出す側ではその呼び出したい Sub プロシージャ名を表記し，渡したい実引数を右側に並べる。呼び出される Sub プロシージャでは（　）の中に，仮引数を入れて作成する。

**構文 5**

```
Sub プロシージャ
呼び出す側
Sub Main( )
   ・・・
  Sub プロシージャ名  実引数1，実引数2，・・・
   ・・・
End  Sub
呼び出される側
Sub  Sub プロシージャ名(仮引数1   As型，仮引数2   As型，・・・)
   ・・・
End  Sub
```

Function プロシージャは，Sub プロシージャと似ているが，それ自体が値を持ってくるため，構文 6 の y のようにどこかに代入される形をとる。また Function プロシージャの中で「Function プロシージャ名＝」という形で値が代入される。

**構文 6**

**Function プロシージャ**
**呼び出す側**
```
Sub　Main( )
　・・・
　Dim　y　As　型
　・・・
　y = Function プロシージャ名（実引数 1，実引数 2，・・・）
　・・・
End　Sub
```
**呼び出される側**
```
Function　Function プロシージャ名（仮引数 1　As型，仮引数 2　As型，・・・）As型
　・・・
　Function プロシージャ名=・・・
　・・・
End　Function
```

**①仮引数と実引数の対応ミス**

Sub プロシージャでも Function プロシージャでも共通のミスであるが，実引数と仮引数の数が合わない，または引数の型が合っていないミスがある。

図 24 は Sub プロシージャでの例であるが，引数の渡し方についてこのような図を使い説明している。呼び出す側の Main プロシージャを実行すると，Test の所で Test プロシージャを探しに行き，見つけるとその中に入る（図 24 ①実線）。その際，右側の実引数 a, b, c を順番に仮引数

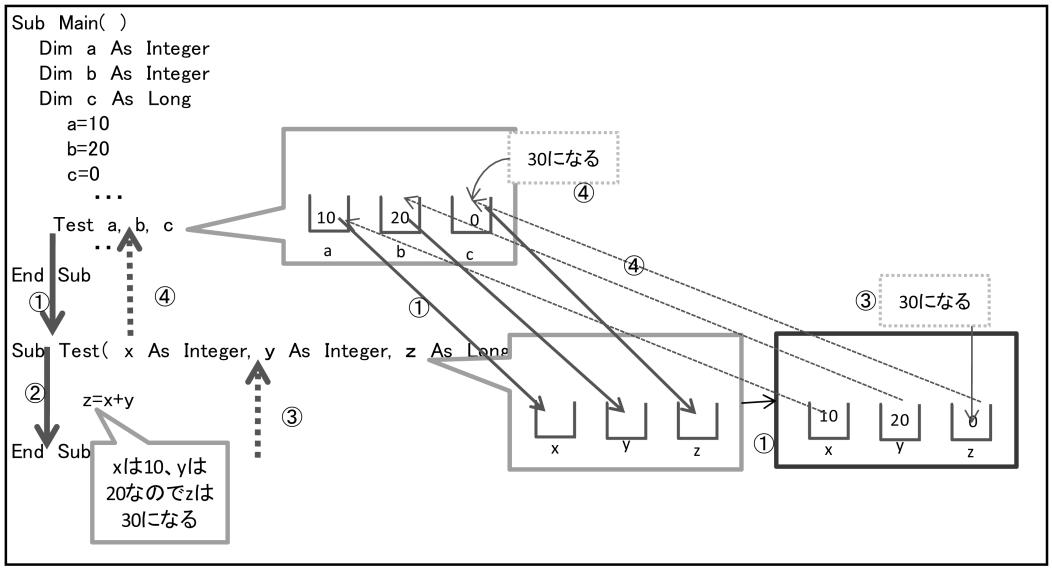



図 24

である変数の箱 x, y, z に落とす。変数の箱に落ちると，以後，変数は x, y, z となって Test プロシージャの中で使われる（②実線）。Test プロシージャの実行終了後は，変数 x, y, z は処理後の値が入っている。この場合 z は 30 となる（図 24 の③破線）。次に呼び出した側に戻り（図 24 の④破線），実引数の変数の箱 a, b, c に値が戻ってくることになる。このため，引数の数と引数の型は重要で，渡す側と渡される側が違えば，行って戻ることができないことを指摘する。

**②配列を引数にした場合のミス**

配列を引数にした場合，実引数でのミスは少ないが，仮引数でカッコ（　）のつけ忘れが非常に多かった。ここでは，配列を忘れてしまっている場合が多いので，配列の説明から繰り返す形になる。

**③実引数と変数について**

プロシージャに実引数として渡しておきながら，プロシージャの中で変数として宣言しているケースが見られた。これは，図 24 の説明とともに引数とはどのようなものかを指摘する。

**④ Function プロシージャの代入ミス**

Function プロシージャはそれ自体が値を持って帰ってくる。そのため，どこかに代入できるようにしなければならない。代入先がないミスが多い。Sub プロシージャと Function プロシージャが一緒になってしまっている学生が多いので再度構文に戻って解説する。

**⑤ Function プロシージャ内で値の代入がない**

Function プロシージャの中では，どこかに「Function プロシージャ＝」という形で値を代入する箇所がなければならないが，それを忘れてしまうミスがある。

**（13）インターフェース作成**

インターフェース作成は，ユーザーフォームを作りその中にオブジェクトを入れる→オブジェクトのプロパティの設定→プログラムの入力→実行→バグ取りの順になる。

**①ユーザーフォームの作成について**

何度かユーザーフォームを出すうちに慣れてくるが，まずユーザーフォームを VBE に出すことに躓く学生が多い。

次にツールボックスからオブジェクトをフォーム上に配置していく段階で，何度もツールボックスをクリックしてしまい，オブジェクト名が違ってしまうミスや違うツールを選んでしまうミスが多い。オブジェクトには，名前がついていることを理解させ，この名前を使ってプログラム

を作成することを指摘する。

**②プログラムの入力について**

いったんフォームに戻って，ユーザーが実行する動作をもとにそこにプログラムを埋めこんでいく考え方を指導する。例えば，「CommandButton 1 をクリックしたら，文字の色が変わる」場合，この CommandButton 1 の中にプログラムを入れるつもりで，CommandButton 1 をダブルクリックして，プロシージャを表示し，表示されると既に入力されている PrivateSubCommand-Button 1__Click と End Sub の間に文字の色を変えるプログラムを入れる（図 25）。もしイベントとなる Click が違えば，右上のイベント一覧が出る▼をクリックしてイベントを選び，プログラミングを入れることなる（図 25）。ここでは，違うオブジェクトまたは違うイベントにプログラムを入力してしまい，プログラムを実行して CmmadButton 1 をクリックしたとしても文字の色が変わらないというミスが多い。


**図 25**

## 6　おわりに

今回，プログラミングについて，学生の躓く点と講義上の留意点について述べた。VBA は，情報教育として Excel を学習済みの学生にとって，受け入れやすいプログラミング言語である。またどこが間違っているのかをエラーメッセージとして表示してくれるのでプログラミング初心者でもわかりやすい。筆者は，講義を受講した学生が，少しでもプログラミングに関心を持ち，更に違うプログラミングに挑戦しようと思ってもらえばと考える。就職をした場合，職場で日常の仕事が楽になるようなツールを作成できるような人材を育成できるよう指導したい。

**参考資料**

太田信宏　「Java プログラミング教育に関する一考察」文教大学女子短期大学部研究紀要 52 集 1–16, 2009
桑原　悟　「プログラミング教育環境の講座王に関する一考察」新潟国際情報大学　情報文化学部紀要
国分道雄　「文系学生へのプログラミング教育」聖学院大学論叢第 20 巻第 2 号