VBA 기본 및 활용

–성균관대학교 보험계리학과 특강–

중앙대학교 통계학과

성 병 찬

E-mail: bcseong@cau.ac.kr & Tel: 02-820-5216

□ 목차

□ 추천 서적 및 웹사이트

- John Walkenbach, Excel Power Programming with VBA
- Uno21.com
- 또는 수많은 VBA관련 웹사이트들

1. VBA의 개념

□ 매크로(macro)

- 어떤 단순한 반복적인 작업을 일괄처리할 수 있도록 하는 기능
- MS Office: VBA; 아래아한글

□ VBA란?

- Visual Basic for Application의 약자
- Visual Basic (VB) ≠ VBA
- MS Office 프로그램들에서 매크로 또는 그 이상을 작성하기 위한 언어
- 장점: 반복작업을 짧은 시간에 처리할 수 있으며, MS Office 프로그램들의 기능을 미세하게 조정하거나 확장할 수 있다.
  - 예: 엑셀에서의 데이터 분석도구, KESS (http://stat.snu.ac.kr/time/) 등
- 단점: 컴파일 언어(C, C++, Fortran)에 비해 속도가 느리고, 엑셀에서 다양한 관련함수들이 많지 않다는 단점이 있다.
  - 참고: DLL (dynamic linking library), 타 프로그램(예: matlab)과의 연결

□ 엑셀에서의 VBA

- 편집창(VBA Editor)에 들어가기 및 관련 메뉴 살펴보기
  - Alt + F11
  - [도구]-[매크로], -[추가기능], -[데이터 분석]
- 구성요소
  - 편집창
  - 도구모음들 (표준, 편집, 디버거(debugger) 등)
  - 직접실행창, 지역창
  - 메뉴 살펴보기: [삽입]-[모듈]
- 기타
  - 개체(object) = 속성(property) + 방법(method)
  - 더 많은 개체와 그것들의 속성, 방법에 익숙해져야 VBA 프로그래밍 실력 향상
    - ✓ 예: Range 개체의 속성(value)와 방법(clear)

```
Sub ChangeValue()
    Worksheets("sheet1").Range("A1").Value=123
End Sub
```



```
Sub ClearRange()
    Worksheets("sheet1").Range("A1").Clear
End Sub
```

  - 디버깅: 프로그램에 존재하는 버그들을 잡아내는 과정
  - Add-in (추가기능): xla → 주로 모듈 및 유저폼으로만 구성되어 있다.
  - xls=스프레드시트 + xla (모듈 및 유저폼 등)
  - 엑셀 도움말 이용하기: syntax (구문), 설명, 예제 살펴보기
    - ✓ 예1: rnd 함수

```
Function MyValue()
    a = Rnd
    b = Round(a, 4)
    MyValue = Int((6 * Rnd) + 1)      '1과 6 사이의 난수를 발생합니다.
End Function
```

    - ✓ 예2: MsgBox 함수

```
Sub MsgboxDemo()
```

```
        MsgBox "Click OK to continue"
    End Sub
    Sub GetAnswer()

        ans = MsgBox("Continue?", vbYesNo)
        Select Case ans
            Case vbYes
                MsgBox "예"
            Case vbNo
                MsgBox "아니오"
        End Select

    End Sub
```

✓ 예3: Visual Basic에서 사용할 수 있는 워크시트 함수 목록

2. 매크로 또는 모듈 작성하기

□ 기록, 편집, 주석(comment); 예,

- A1에 123 입력하기
- 123이면 msgbox로 OK! 출력하기
- 주석달기와 해제하기
- 기록하기를 통하여 개체들의 이름, 속성 및 방법을 살펴볼 수 있다.
  - 예: 매크로 기록기를 통하여 위의 예를 실행해 보자.

```
Sub write_a1()
    ActiveCell.FormulaR1C1 = "123"
    Range("A2").Select     'A2 값을 바꿔보자.
    ActiveCell.FormulaR1C1 = "123"
    ActiveCell.Offset(1, 0).Range("A1").Select
    '(1,0) 대신 (2,2)를 대입해보자.
End Sub
```

□ 실행하기
- F5키
- 스프레드시트 상에서 [도구]-[매크로]
- 단축키 및 할당하는 방법

□ 디버깅: 직접실행창, 지역창
- 직접실행창: 엑셀 및 VBA 관련 값 부여 및 알아내기
  - 예:

```
Range("a1").Value=123
Print 25*3
? 25*3
? activeworkbook.name
? Range(“a1:a10”).cells(2,1).value
```

- 지역창: 모듈의 변수값 알아내기
  - 또는, 커서를 해당 변수 위에 둔다.
- 예: 중단점 활용한 디버깅

```
Sub mySum()
```



```
    Dim sum As Double
    For i = 1 To 100
        sum = sum + i
    Next i
    MsgBox sum, Title:="1부터100까지 더하면?"
End Sub
```

□ 연산자(oprators)
- 대입연산자
- 사칙연산(+, –, x, /)
- 논리연산(and, or, not)
- 기타: mod
- 예:

```
Sub Operators()

    x = 1
    x = x + 1
    x = x ^ 2
```

```
    y = (x * 2) / (x * 4)
    z = 10 Mod 3

    FileOpen = True
    FileOpen = Not FileOpen
    Range("A1").Value = 2009

End Sub
```

□ 사용자정의함수(user-defined function, module, subroutine)의 종류

- Sub vs Function
- 반환값 유무의 차이
- 사용자정의함수 이름 규칙
  - 예: 숫자 및 특수기호로 시작할 수 없다; 공백이 있을 수 없다.
- 사용자정의함수의 구조
  - 함수이름, 입력값, 본문, 반환값(출력값)
  - 예:

```
Sub test()
```



```
    Sum=range("a1").value+1
    Msgbox "The answer is " & Sum
End Sub

Function AddTwo(arg1, agr2)
    AddTwo=arg1+arg2
End Function

Function func_mySum(endp As Double) As Double
    Dim sum As Double: Dim i As Double
    For i = 1 To endp
            sum = sum + i
    Next i
    func_mySum = sum
End Function
```

- 들여쓰기(indentation)

□ 모듈 부르기

- Sub의 실행: [도구]-[매크로]
- Function의 실행: [삽입]-[함수]
- 모듈에서 특정 sub 또는 function 부르기
- 예:

```
Sub test2()
    a = func_mySum(5)     'function call
    MsgBox "1부터 5까지 더하면" & a
    sub_mySum     'sub call
End Sub
```


3. 프로그래밍을 위한 주요 구문 및 요소들

□ 변수 종류 및 선언하기

- 변수종류(data type): Boolean, Integer, single, long, float, double, string, variant
- Variant: 변수를 특정 타입으로 정하기 어려운 경우
- Dim 변수이름 as 변수종류
- 주로 함수의 맨위에서 선언
- 변수를 선언하면 프로그래밍이 좀더 편하거나 쉬워질 수 있다. 변수를 선언하지 않고 사용하면 메모리 낭비 및 디버깅 어려울 수 있다.
- Option explicit
- 예제:

```
Sub test3()

    Dim v1 As Integer
    Dim v2 As Long
    Dim v3 As Double
    Dim v4 As String
```

```
v1 = 10
v1 = 10.2
v2 = 10.3
v3 = 10.3456
v4 = "I am a string type."
conv = Int(v3)      'conv is a variant type.

Dim rn As Range

Set rn = Range("b2")
Value = rn.Value

End Sub
```

- 참고: 개체 변수
  - 예:

```
Sub RangeV()
    Dim r As Range
    Dim s As Worksheet

    Set s = ActiveSheet
    Set r = ActiveCell
    '개체변수를 할당할 때는 Set 을 사용!
    MsgBox s.Name
    MsgBox r.Address
End Sub
```


□ 배열 (array)

- 통계학 관련 함수 작성할 때는 주로 두개 이상의 관측치를 다루기 때문에 배열을 사용하는 것이 편리
- 배열선언 및 사용방법
- 예제:

```
Sub myArray()

    Dim a(0 To 99) As Double
    Dim b(1 To 100) As Double
    Dim c(1 To 10, 1 To 10) As Integer
```

```
        For i = 1 To 100
            a(i - 1) = i
        Next i

        MsgBox Application.sum(a)
        Debug.Print Application.Max(a)
        Debug.Print Application.Min(a)
        Debug.Print Application.Average(a)

    End Sub

    Sub myArray2()
        myend = 5
        subArray myend
    End Sub

    Sub subArray(endp)

        Dim a() As Double     'dynamic array
        ReDim a(0 To endp)    '끝이 구체적으로 정해졌을 때 비로소 선언

        For i = 1 To UBound(a)
            a(i - 1) = i
```



```
        Next i

        MsgBox Application.sum(a)

    End Sub
```

□ 조건문

- 프로그램에서 어떤 판단을 해야하는 시점에서 사용
- 종류: if ~ then ~ end if, select case
- 구문

```
If condition Then
    [statements]
[ElseIf condition-n Then
    [elseifstatements]] …
[Else
    [elsestatements]] …
End If
```

```
Select Case testexpression
[Case expressionlist-n
    [statements-n]] …
[Case Else
    [elsestatements]]
End Select
```

- 예제

```
Function func_mySum(endp As Double) As Double
    Dim sum As Double: Dim i As Double
    For i = 1 To endp
        If i > 50 Then
            sum = sum + i
        ElseIf i > 20 Then
            sum = sum + 0.5 * i
        Else
            sum = sum + 0.1 * i
        End If
    Next i
    func_mySum = sum
End Function

Function Bonus(performance, salary)
```



```
    Select Case performance
        Case 1
            Bonus = salary * 0.1
        Case 2, 3
            Bonus = salary * 0.09
        Case 4 To 6
            Bonus = salary * 0.07
        Case Is > 8
            Bonus = 100
        Case Else
            Bonus = 0
    End Select
End Function
```

□ 반복문 (Loop)

- 특정한 작업을 계속 반복해야 하는 상황에서 사용
- 종류: For ~ next, do loop
- 구문

For counter=*start* To *end* [Step stepval]

[*statements*]
[Exit For]
[*statements*]
Next [counter]

Do [While condition]
[*statements*]
[Exit Do]
[*statements*]
Loop

- 예:

```
Sub SumOdd()
    sum = 0
    For Count = 1 To 100 Step 2
        sum = sum + Count
    Next Count
    MsgBox sum
```



```
End Sub

Sub sumeven()
    sum = 0: Count = 1
    Do While Count <= 100
        Count = Count + 2
        sum = sum + Count
    Loop
    MsgBox sum
End Sub
```

□ 행렬연산

- 통계학 관련된 함수를 작성할 때 유용
- 단점: 덧셈, 뺄셈이 불편; 연산자가 없다.
- 종류: Application.MMult(a,b), . MInverse(a), . Transpose(a)
- 예제:

```
Sub Matrix_Al()
```

```
    Dim A

    Set A = Range(Cells(1, 1), Cells(10, 10))

    b = Application.MMult(A, A) 'Calculate A^2
    c = Application.MMult(Application.MMult(A, A), b) 'Calculate A^3
    D = Application.MInverse(Application.MMult(Application.Transpose(A), A)) 'Calculate (A'*A)^(-1)

    MsgBox D(2, 2) 'Find (2,2) elts of (A'*A)^(-1)

End Sub
```

□ 자주 사용되는 VBA 함수들

- 수학통계함수
  - exp(), log(), max(), min(), sqr()
  - Application.
    - ✓ .average(), .fact(), .max(), .min(), .sum()

    - ✓ .normsdist()
  - Rnd

- 기타함수
  - selection
  - inputbox, msgbox

- 예제:

```
Sub Factorial()
    'calculate the factorial of a number
    'On Error Resume Next
    num = InputBox("Enter integer number", "Calculate Factorial")
    fac = Application.Fact(num)
    MsgBox "Factorial is " & fac
End Sub

 Sub mySelection()
```

```
    s = Application.sum(Selection)
    MsgBox "The sum of selection is " & s
End Sub
```

□ 셀을 참조하는 방법

- 상대참조, 절대참조
  - 워크시트함수에서의 개념이 프로그램에서도 사용
- 상대참조로 매크로 기록하기
- 예:

```
Sub 상대참조로기록()
    ActiveCell.Offset(3, 4).Range("A1").Select
    ActiveCell.FormulaR1C1 = "123"
    ActiveCell.Offset(1, 0).Range("A1").Select
End Sub
Sub myMonth()
    ActiveCell.Offset(0,0)=”Jan”
    ActiveCell.Offset(0,1)=”Feb”
```



```
    ActiveCell.Offset(0,2)=”Mar”
End Sub
```

- Range 방법, Cells 방법
- 예: 디버그 모드를 이용하여 워크시트의 변화를 살펴보자.

```
Sub methods()

    Range("c1:c10 a6:e6") = 3
    Range("A1,A3,A5,A7,A9") = 4
    ActiveCell.Range("b2") = 5
    Worksheets("sheet4").Cells(1, 1) = 1
    ActiveSheet.Cells(3, 4) = 7
    ActiveCell.Cells(2, 1) = 5

End Sub
```