บทนำ

การใช้งาน MS-Excel มีหลายหลายทั่วไปตามหน่วยงานต่าง ๆ ทั้ภาคเอกชน และรัฐบาล ผู้ที่ใช้สูตรต่างๆของ Excel ได้มากก้จะเพิ่มประสิทธิภาพในงานของตนเองได้มาก ถ้าวิศวกรหรือพนักงานคนใดสามารถเขียนโปรแกรม visual basic for Application บน MS-Excel ได้ก็ยิ่งจะทำให้สามารถทำงานที่ซับซ้อนได้มากยิ่งขึ้น

ที่มาภาพ : https://www.bullfrag.com/

งานหลาย ๆ งานในโรงงานเป็นงานที่ต้องใช้สูตรต่าง ๆ ของ Excel และทำซ้ำ ๆ ทุกสัปดาห์ และเสียเวลามาก ถ้าวิศวกรหรือพนักงานคนใดสามารถเขียนโปรแกรมให้ทำงานแทนได้ จะทำให้งานที่ทำซ้ำ ๆ ดังกล่าวใช้เวลาน้อยลง เหลือเพียงแค่กรอกข้อมูล แล้วสั่งให้ โปรแกรมทำงานแทนเราได้ จะเป็น 100 รอบ 1,000 รอบ หรือ 10,000 รอบก็ไม่ไช่เรื่องใหญ่อะไรสำหรับโปรแกรมคอมพิวเตอร์

โรงงานแต่ละโรงงานบริษัทขนส่งแต่ละแห่งมีลักษณะการทำงานและปัญหาที่แตกต่างกัน ไม่มีโปรแกรมสำเร็จรูปใดสามารถตอบโจทย์และตอบความต้องการของโรงงานได้ทั้งหมด นอกเสียจากว่าพนักงานที่รู้จักงานนี้จะเขียนโปรแกรมเสียเองจะสามารถแก้ปัญหาดอย่างมีประสิทธิภาพ

สาเหตุที่ผู้แต่งใช้โปรแกรม VBA บน Excel ในการแก้ปัญหาต่าง ๆ ในโรงงานเนื่องจาก เหตุผลที่ 1 โรงงานส่วนใหญ่มี MS-Excel และใช้กันแพร่หลายอยู่แล้ว การใช้โปรแกรมที่บรรจุอยู่ใน Excel อยู่แล้วจึงไม่มีค่าใช้จ่ายเพิ่มเติม เหตุผลที่ 3 สูตรต่าง ๆ ของ Excel มีมากมาย เมื่อเขียนโปรแกรมบน Excel ทำให้สูตรทั้งหลายสามารถนำมาใช้ได้ในโปรแกรม เหตุผลที่ 4 สำหรับกรณีที่เขียนโปรแกรมให้พนักงานคนอื่นใช้ เนื่องจากพนักงานส่วนใหญ่ใช้ Excel เป็นอยู่แล้ว การที่เขียนโปรแกรมบน Excel แล้วนำไปให้ใช้งาน จะได้รับการต่อต้านน้อยกว่า เพราะความเคยชินใน Excel ที่มีอยู่เดิม

ผมสอนอยู่ที่มหาวิทยาลัยศิลปากร งานเขียนฉบับนี้เป็นผลงานศิลปของผม ที่รวมทั้งศาสตร์และศิลปะ เข้าไว้ด้วยกันดังนั้นเอกสารจึงมีศาตร์ที่ว่าด้วยวิชาความรู้และศิลปะอันประกอบด้วยศิลปะในการสร้างเอกสารด้วย power point ไม่ใช่ word และใช้ประโยชน์จาก power point ในการวางตำแหน่งของรูปและตารางให้ดูน่าอ่าน เนื่องจากมีภาพเยอะ ดังนั้นรูปภาพบางรูปภาพแสดงความสวยงามจะไม่มีการอ้างอิงรูปภาพเหล่านั้น แต่มีประกอบให้หนังสือน่าอ่าน

บทที่ 1 หลักการเบื้องต้น

1 object. Property

คอมพิวเตอร์จะมองทุกอย่างเป็นวัตถุ ดังนั้นผู้เขียนจะใช้คำสั่งว่า “วัตถุ จุด คุณสมบัติ” ตัวอย่างเช่น

ตัวอย่างที่ 1.1

```
Car. Color=red
Car. Wheels = 4
```

ในตัวอย่างที่ 1.1 เป็นการกำหนดให้รถมีสีแดง และกำหนดให้รถยนต์มี 4 ล้อ

2.object. method

วัตถุจะสามารถดำเนินการต่าง ๆ ได้ ดังนั้นนักเขียนโปรแกรมต้องสามารถสั่งให้วัตถุนำเดินการต่าง ๆ ได้ เช่น

ตัวอย่างที่ 1.2

```
Excel. Open
Excel. Close
Car. Park
Car. Drive
```

ในตัวอย่างที่ 1.2 เป้นการเขียนโปรแกรมเพื่อ ให้โปรแกรม excel เปิดขึ้นมา ให้โปรแกรม excel ปิดลงไป กำหนดให้รถยนต์จอดอยู่กับที่ และกำหนดให้รถยนต์ขับเคลื่อนตามลำดับ อย่างไรก็นี้คำสั่งทั้งสี่นี้เป็นเพียงตัวอย่างที่ให้เห็นภาพการทำงานของโปรแกรมเท่านั้น

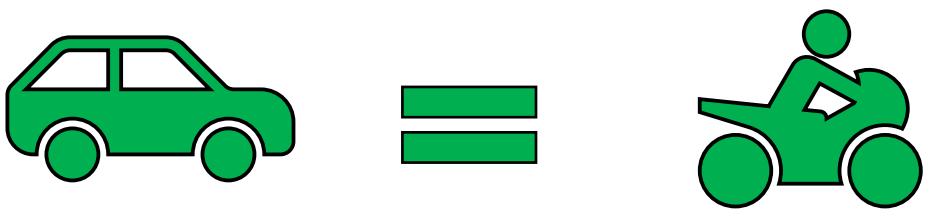
ซ้ายรับค่า ขวาให้ค่า

ในโปรแกรมคอมพิวเตอร์ผู้อ่านพึงระลึกไว้เสมอว่า ด้านซ้ายรับค่า ด้านขวาเป็นฝั่งให้ค่า เช่น

ตัวอย่างที่ 1.3

Car. Wheels=Motorcycle. Wheels

ตัวอย่างที่ 1.3 รถยนต์จะมีจำนวนล้อเท่ากับ มอเตอร์ไซด์เนื่องจากกฎที่ว่า ซ้ายรับค่าขวาให้ค่า



ตัวอย่างที่ 1.4

A=3

B=2

A=B

ตัวอย่างที่ 1.4 คอมพิวเตอร์จะอ่านจากบนลงล่างเสมอ ตอนแรก A จะมีค่าเป็น 3 ขั้นตอนถัดมา B จะมีค่าเป็น 2 และ ขั้นตอนถัดมา จะนำค่าจาก B ไปใส่ให้ A ดังนั้น A จึงมีค่าเป็น 2 เช่นเดียวกัน

เรื่อง error หรือ debug โปรแกรม

โปรแกรมมักผิดพลาดได้หลายสาเหตุ ไม่ต้องกังวลเพราะเป็นเรื่องทั่วไปที่เกิดขึ้นได้ ตราบใดที่ยังพิมพ์ Ms-word ผิดได้ จะนับประสาอะไรที่โปรแกรมจะไม่ผิด ดังนั้นจึงควรมองเป็นธรรมชาติของการทำงาน เพียงแต่ผู้อ่อนต้องแก้ไขให้มันถูกต้องเท่านั้น วิธีการเช็คข้อผิดพลาดให้ใช้หลักการดังนี้

1 ถ้า error บรรทัดใด แปลว่า บรรทัดนั้นผิด หรือ บรรทัดก่อนหน้านั้นผิด

2 ใช้ break point เพื่อตรวจสอบค่าต่าง ๆ ในตัวแปรที่คาดว่าจะผิด

3 ถ้าโปรแกรมซับซ้อนอาจใช้แผนภูมิก้างปลาช่วยในการวิเคราะห์ว่าผิดจากอะไรได้บ้าง แล้วตรวจสอบทีละประเด็น

4 เขียนโปรแกรมบ่อย ๆ อาจเจอ error บ่อย ๆ เวลาผ่านไป ความชำนาญจะช่วยให้ผู้อ่านสามารถแก้ error ได้รวดเร็วขึ้น และเขียนโดยมีความผิดพลาดลดลง

5 เขียนไปเช็คไปทีละจุด อย่าเขียนทีเดียวแล้วค่อยตรวจสอบ เพราะจะหา error ไม่เจอ

ผู้ช่วยศาสตราจารย์ ดร.คเณศ พันธุ์สวาสดิ์

โครงสร้างโดยรวม

หลายครั้งที่นักเขียนโปรแกรมใหม่ มือเผลอไปโดนคีย์บอร์ด ทำให้ข้อมูลบางอย่างลบไป แล้วไม่สามารถแก้ไขได้ในช่วงแรก ดังนั้นผู้เขียนจึงแนะนำให้รู้จักโครงสร้างโดยรวมเสียก่อน (ดังภาพที่ 1.1) เพื่อว่าเมื่อใดที่พลาดไปลบทำให้โครงสร้างโปรแกรมเสียไปจะได้แก้ไขกลับมาได้

```
Public sub SubName1
        คำสั่ง
End sub
Public sub SubName2
        คำสั่ง
End sub
Public function FuncName1
        Return ค่า
End function
```

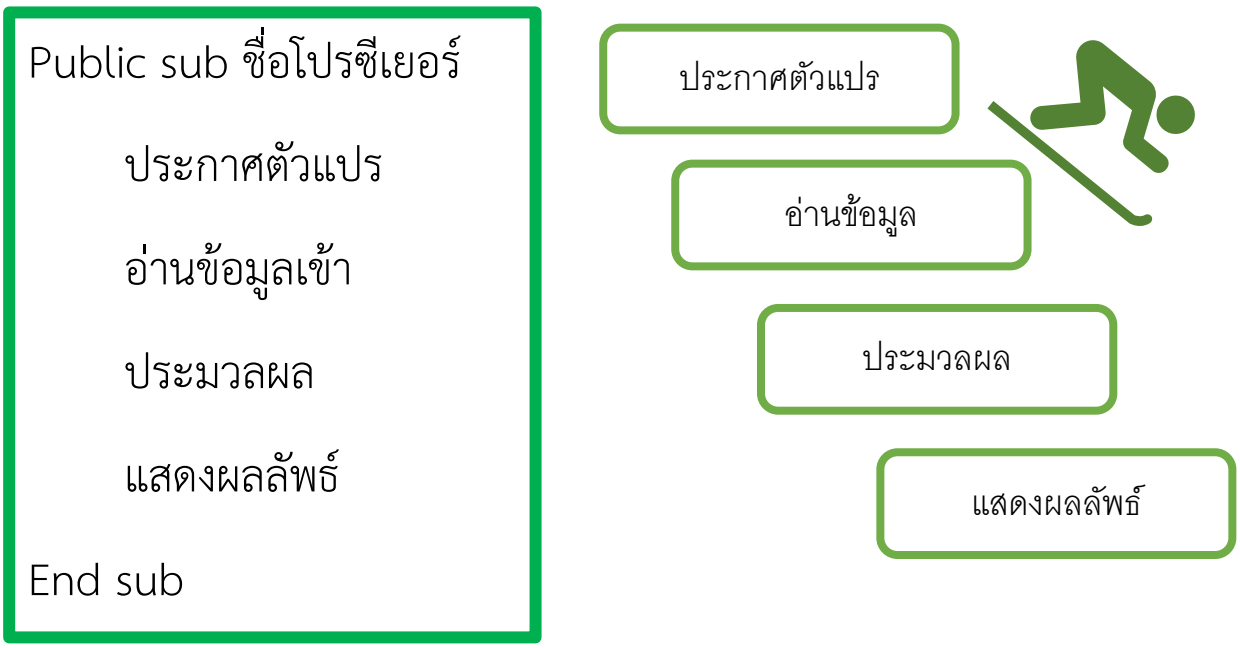
ภาพที่ 1.1 โครงสร้างโปรแกรม

สังเกตดูจะพบว่า โครงสร้างนอกสุดจะเป็น class หรือ module ภายในจะประกอบด้วย sub หลาย ๆ sub หรือ ฟังก์ชัน หลาย ๆ ฟังก์ชัน ก็ได้ ต้องมีเปิด และปิดเสมอ โดย public sub จะปิดด้วย end sub และ public function จะปิดด้วย end function และนอกจากนี้ฟังก์ชันต้องมีการคืนค่ากลับไปยังตัวแปรด้วย (จะกล่าวต่อไปในภายหลัง)



ลำดับการเขียนโปรแกรม

ในการเขียนโปรแกรมให้เป็นระเบียบ ผู้อ่าน อาจเขียนโปรแกรมตามลำดับดังภาพที่ 1.2 เริ่มต้นจากการประกาศตัวแปรที่ต้องใช้ อ่านข้อมูลมาเก็บไว้ในตัวแปร เขียนการคำนวณหรือการประมวลผล และปิดท้ายด้วยการแสดงผลลัพธ์ ในบางครั้งผู้เขียนโปรแกรมอาจนึกขั้นตอนทั้งหมดในการเขียนไม่ได้ในตอนแรกเริ่ม ผู้แต่งแนะนำว่า เขียนส่วนใดส่วนหนึ่งก่อนก็ได้ แล้วค่อยๆ เติมส่วนที่เหลือจนโปรแกรมสมบูรณ์



ภาพที่ 1.2 ลำดับการเขียนโปรแกรม

Note พึงคิดไว้เสมอว่า การเขียนโปรแกรม “ต้องคิดแบบที่คอมพิวเตอร์คิด”

ตัวแปร

ในการเขียนโปรแกรม ผู้อ่านควรคิดว่าในการทำงานดังกล่าวต้องใช้ตัวแปรกี่ตัว อะไรบ้าง และตัวแปรควรเป็นชนิดใด โดยชนิดของตัวแปรจะกล่าวโดยละเอียดในภายหลัง

การประกาศตัวแปร

ชนิดของตัวแปรที่นิยมใช้ประกอบด้วย

Integer ใช้เก็บจำนวนเต็ม

long ใช้เก็บจำนวนเต็มแต่เก็บได้มากกว่า integer

string ใช้เก็บตัวอักษร หรือข้อความ

single ใช้เก็บทศนิยม

double ใช้เก็บทศนิยมที่มากกว่า single

Boolean ใช้เก็บ จริง/เท็จ

ชนิดของข้อมูลเหล่านี้เป็นที่นิยมใช้ในการเขียนโปรแกรม แต่ความจริงแล้วตัวแปรยังมีอีกมาก แต่ผู้เขียนขอยกตัวอย่างไว้เพียงเท่านี้เพื่อจะได้ไม่สับสน ถ้าต้องการรายละเอียดทั้งหมดผู้อ่านอาจค้นคว้าได้จากหนังสือคอมพิวเตอร์

วิธีการประกาศตัวแปรในแต่ละ sub หรือ ฟังก์ชันทำได้ดังนี้

```
Dim A as integer
Dim B as Long
Dim C as Single
```

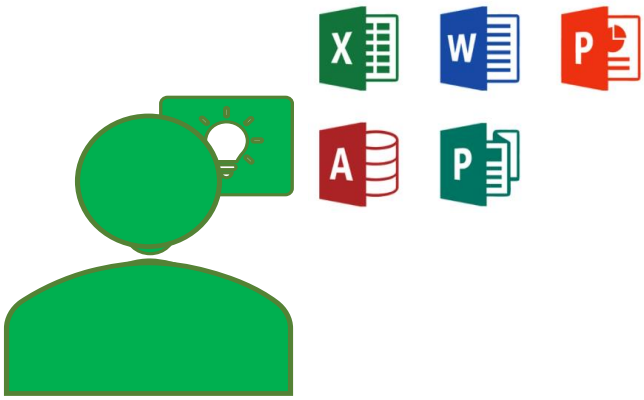
การประกาศตัวแปรลักษณะนี้ ตัวแปรจะมองเห็นเฉพาะใน sub หรือ function เท่านั้น ถ้าต้องการประกาศตัวแปรให้ sub อื่น หรือ function อื่นสามารถเรียกใช้ได้ด้วย จะประกาศว่า

```
Private R as integer
Public X as integer
```

การประกาศ private จะสามารถเรียกใช้ตัวแปรใน class หรือ module เดียวกันได้ทั้งหมด ส่วนการประกาศว่า public จะสามารถเรียกใช้ได้ทั้ง program รายละเอียดและตัวอย่างผู้เขียนจะละไว้เพื่อไม่ให้เกิดความสับสน และจะนำตัวอย่างมากล่าวในภายหลังหากจำเป็นใช้งาน

การอ่านค่า

ในการรันโปรแกรม ผู้อ่านต้องจินตนาการไปว่า ตัวแปรแต่ละตัวต้องรับค่ามาจากแหล่งใด อาทิเช่น แป้นพิมพ์ หรือจากการอ่านค่าใน excel access หรือจะเป็นการรับค่าผ่านโปรแกรมโดยตรง



ตัวอย่าง การอ่านค่าจาก excel

```
a = Sheets(1).Cells(1, 2).Value

a = Sheets(1).Range("B1").Value
```

ตัวอย่าง การอ่านค่าจาก textbox

```
A=textbox1.text
```

Operator ต่าง ๆ

ตัวดำเนินการสำหรับคำนวณ

ในการเขียนโปรแกรม เครื่องหมายบวกลบคูณหาร ต่าง ๆ จะมีลำดับการประมวลผลต่างกันผู้อ่านควรทำความเข้าใจให้ดีเพราะจะส่งผลต่อคำตอบให้ถูกหรือผิดพลาดได้ง่าย โดยลำดับของ operator มีลำดับดังตารางที่ 1.1

ตารางที่ 1.1 ลำดับการทำงานของตัวดำเนินการ

ลำดับ	operator	คำอธิบาย
1	()	วงเล็บ
2	^	ยกกำลัง
3	-	เลขติดลบ
4	/	หาร
4	*	คูณ
4	\	หารไม่เก็บเศษ
4	mod	เศษการหาร
5	+	บวก
5	-	ลบ

แบบฝึกหัด

ข้อ 1.1 จงหาค่าต่อไปนี้

5+5x-5 = ?

ข้อ 1.2 จงหาค่าต่อไปนี้

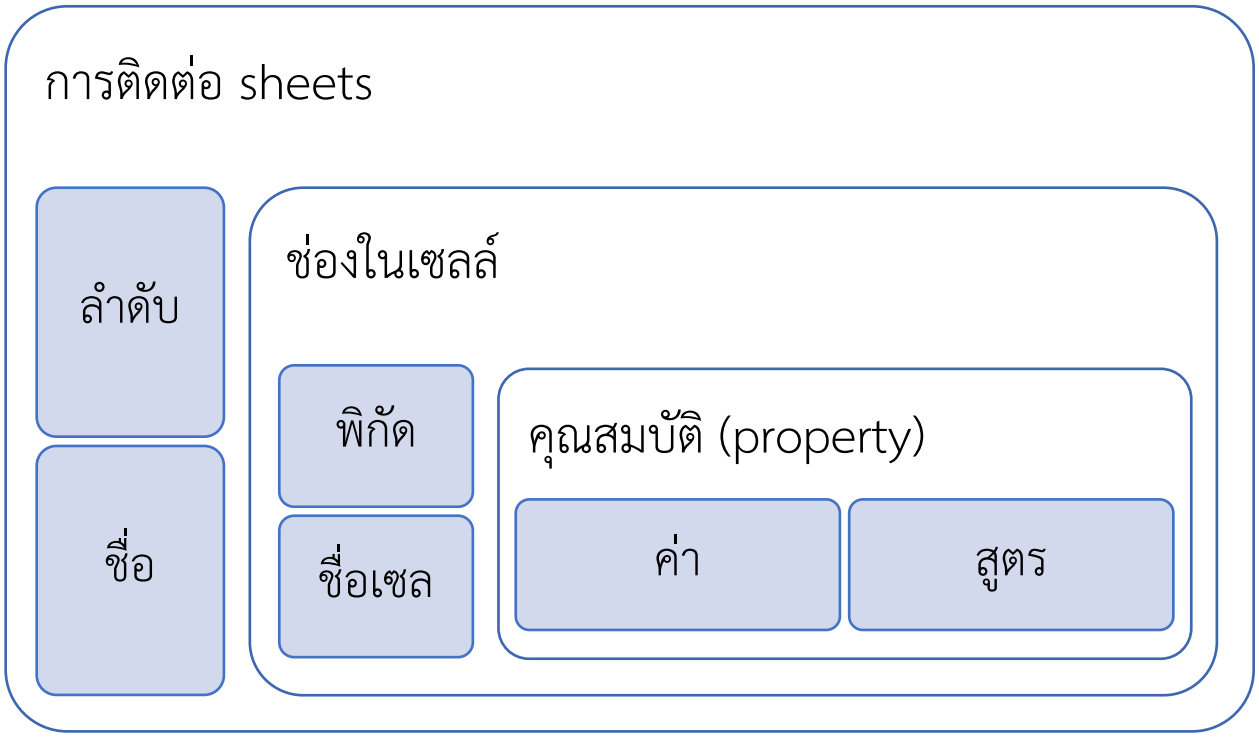
(5 mod 3) ^2 =?

ข้อ 1.3 จงหาค่าต่อไปนี้

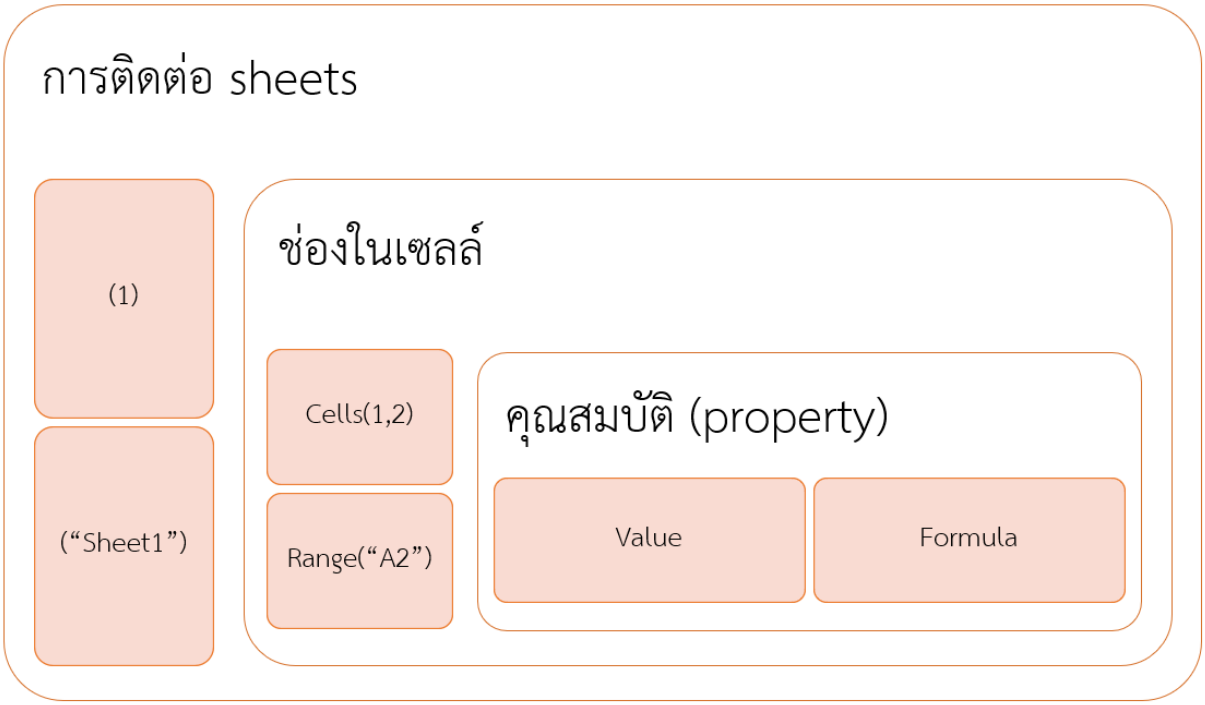
4+4*2^2=?

การติดต่อกับ excel

การติดต่อกับ excel ทำได้ 2 รูปแบบ รูปแบบแรกเป็นการติดต่อด้วยตัวเลขเช่นอาศัยลำดับของชีทส์ (นับจากซ้ายไปขวา ซ้ายเป็นลำดับที่ 1) และอาศัยลำดับ อ้างแถวและสดมภ์ รูปแบบที่สอง อ้างด้วยชื่อของ ชีทส์และเซลล์ ภาพที่ 1.3 แสดงวิธีอ้าง และภาพที่ 1.4 แสดงตัวอย่างการอ้างถึง



ภาพที่ 1.3 ลำดับการอ้างถึงเซล



ภาพที่ 1.4 ตัวอย่างการอ้างถึงเซล

ตัวอย่างการติดต่อ excel ด้วยชื่อ

คำสั่ง sheets ตามด้วยชื่อของ sheet การใช้คำสั่ง Range จะตามด้วยชื่อของ เซล เช่น A1 A2 และสุดท้ายใช้ property value ดังแสดงในซับ excelconnect1

```
Public Sub excelconnect1()
    Sheets("sheet1").Range("A1").Value = "fried rice"
    Sheets("sheet1").Range("A2").Value = "fried chicken"
    Sheets("sheet1").Range("A3").Value = "pizza"
End Sub
```

ตัวอย่างการติดต่อ excel ด้วยลำดับตัวเลข

Sheets ตามด้วยลำดับนับจากซ้ายไปขวาคำสั่งเซลล์ตามด้วยแถว และหลักแล้วตามด้วยคุณสมบัติ value

```
Public Sub excelconnect2()
    Sheets(1).Cells(1, 2).Value = 80
    Sheets(1).Cells(2, 2).Value = 45
    Sheets(1).Cells(3, 2).Value = 300
End Sub
```

ผลลัพธ์ที่ได้จากการรัน excelconnect และ excelconnect2 แสดงดังภาพที่ 1.5

	A	B	C
1	fried rice	80	
2	fried chicken	45	
3	pizza	300	
4			

ภาพที่ 1.5 ผลจากการเขียนโคร์ด

การเขียนแบบเงื่อนไข

ในการเขียนโปรแกรม บางคำสั่งจะให้ดำเนินการทุกครั้งที่รันโปรแกรม แต่ในบางครั้งผู้อ่านอาจต้องการให้รันเฉพาะบางกรณี ซึ่งในโปรแกรม VB ได้เตรียมไว้ 2 ชุดคำสั่งคือ if then else และ select case ชุดคำสั่งทั้งสองใช้ในการตรวจสอบ จะใช้ร่วมกับตัวดำเนินการดังตารางที่ 1.2

ตารางที่ 1.2 ตัวดำเนินการเปรียบเทียบ

สัญลักษณ์	ความหมาย
=	เท่ากับ
<>	ไม่เท่ากับ
>=	มากกว่าเท่ากับ
<=	น้อยกว่าเท่ากับ
>	มากกว่า
<	น้อยกว่า

If then else

ตัวอย่าง เช่น ถ้าฝนตก แล้ว เราจะกางร่ม เขียนได้ว่า

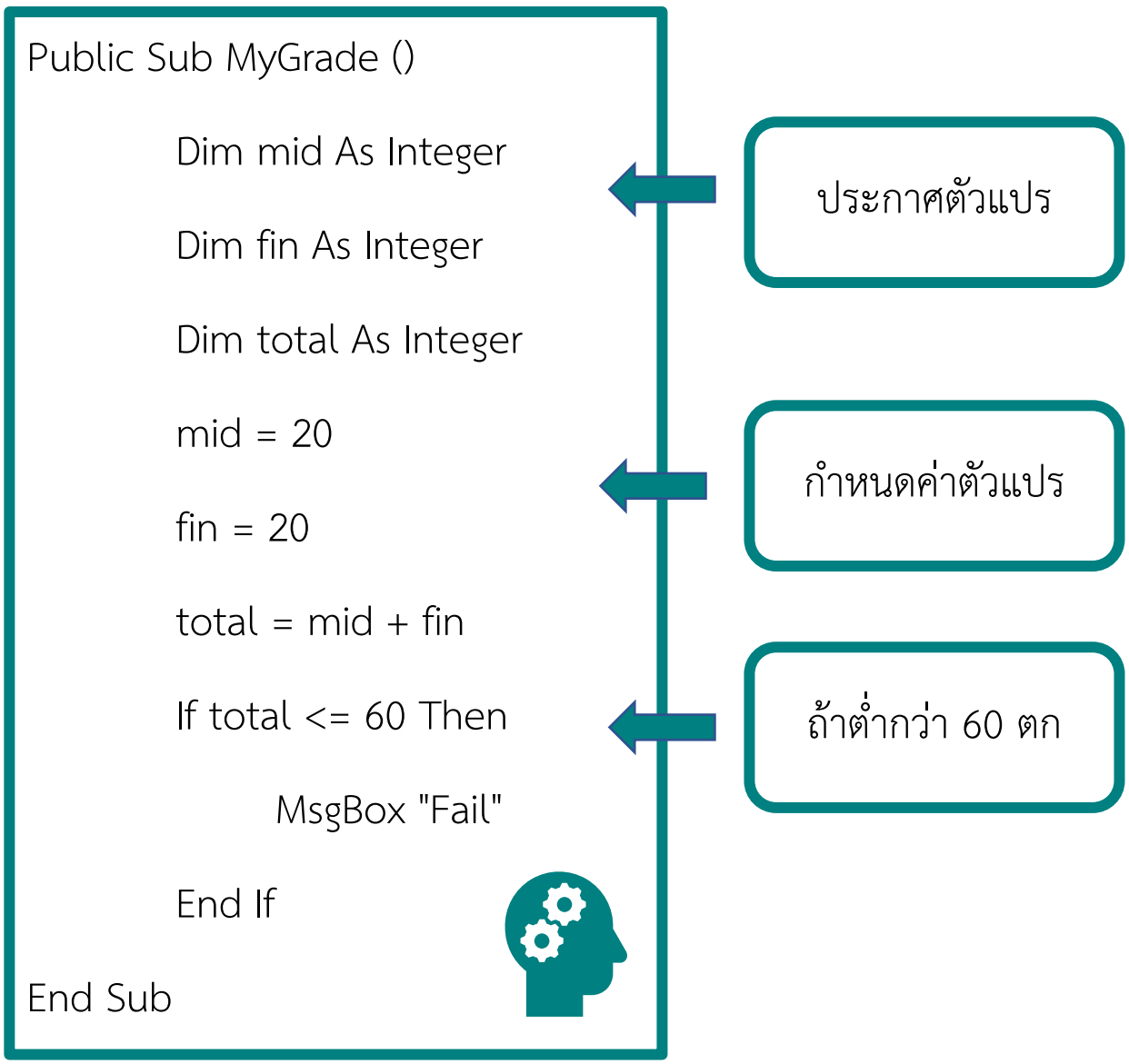
"If ฝนตก then เราต้องกางร่ม"

ชุดคำสั่งจะใช้ในกรณีที่ จะให้ทำเมื่อเป็นจริงตามเงื่อนไข โดยมีโครงสร้างดังภาพที่ 1.6

```
If เงื่อนไข then
        คำสั่งที่ 1
        คำสั่งที่ 2
End if
```

ภาพที่ 1.6 โครงสร้างของ IF

ตัวอย่าง โปรแกรม myGrade เริ่มต้นจากการประกาศตัวแปรเป็นจำนวนเต็ม กำหนดค่าให้ตัวแปร mid และ fin (ซ้ายรับค่า ขวาให้ค่า) หลังจากนั้นนำค่ามารวมกันและเก็บค่าไว้ใน total หลังจากนั้นตรวจสอบเงื่อนไข พบว่าคะแนนน้อยกว่า 60 เมื่อเป็นเช่นนี้จะมี แมชเซส บอกซ์ ขึ้นว่า “fail” แสดงดังภาพที่ 1.7



ภาพที่ 1.7 การใช้ IF ตรวจสอบการตก

การใช้ IF แบบมี Else

นอกจากนี้ผู้อ่านยังอาจแยกเงื่อนไขเป็นสองทางก็ได้โดยโครงสร้างชุดคำสั่งที่ใช้แสดงไว้ดังภาพที่ 1.8 และตัวอย่างแสดงดังภาพที่ 1.9

```
If เงื่อนไข then
    คำสั่งที่ 1
    คำสั่งที่ 2
Else
    คำสั่งที่ 3
    คำสั่งที่ 4
End if
```

ภาพที่ 1.8 โครงสร้างแบบ 2 ทาง

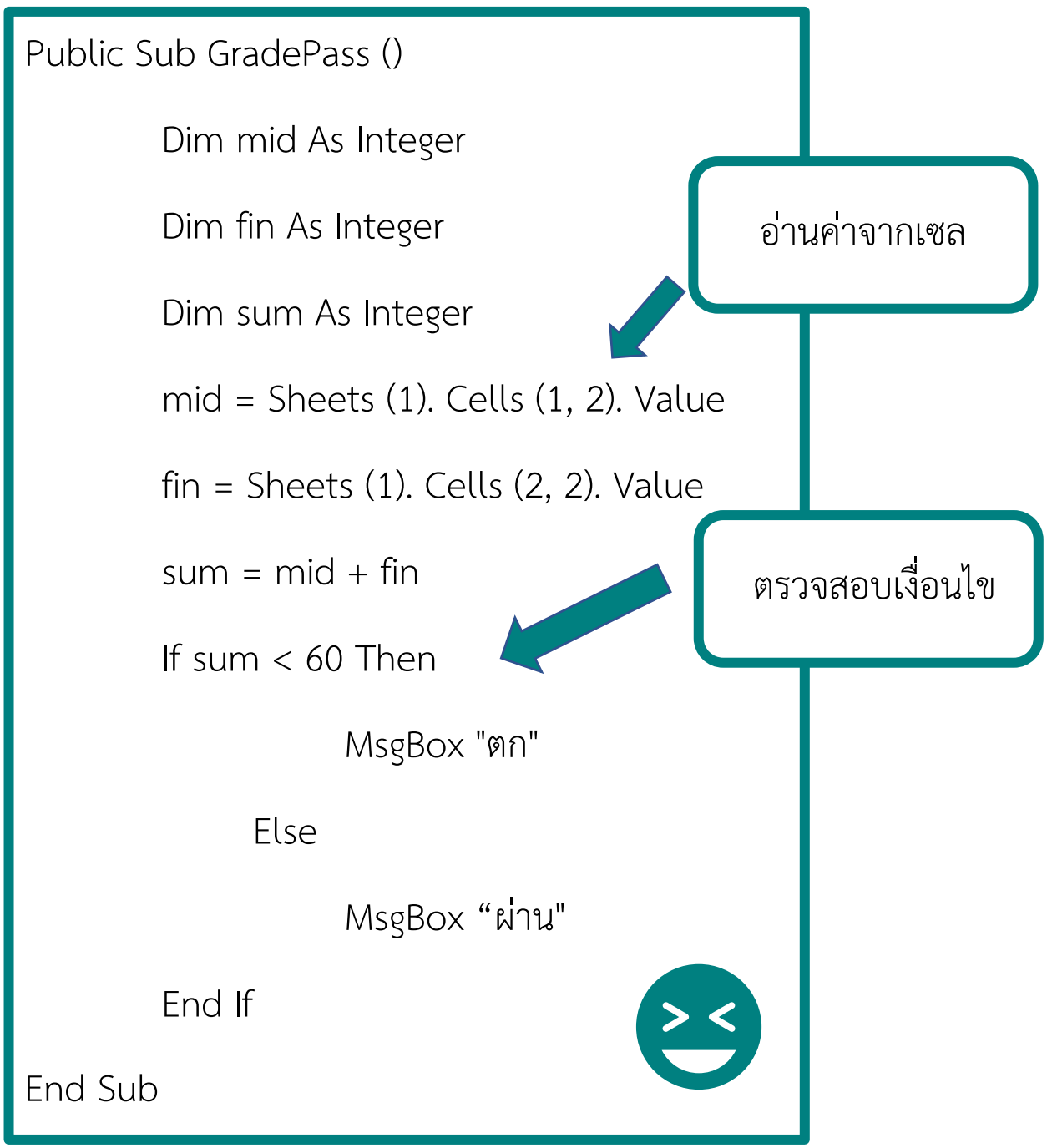
```
Public Sub gradeFistTime ()
        Dim mid As Integer
        Dim fin As Integer
        Dim sum As Integer
        mid = 30
        fin = 40
        sum = mid + fin
        If sum < 60 Then
            MsgBox "Fail"
        Else
            MsgBox "Pass"
        End If
End Sub
```

ภาพที่ 1.9 ตัวอย่างแบบ 2 ทาง

ตัวอย่างที่ 2 แสดงการรวมคำสั่งการอ่านค่าจาก Sheet ไปเก็บไว้ในตัวแปร min fin ด้วยคำสั่ง

```
mid = Sheets(1).Cells(1, 2).Value
fin = Sheets(1).Cells(2, 2).Value
```

ดังแสดงในภาพที่ 1.10



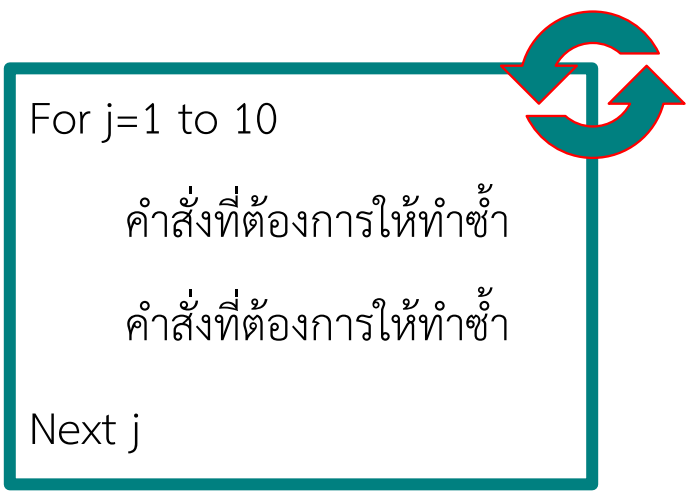
ภาพที่ 1.10 การผสมการอ่านค่าและ if

บทที่ 2 การวนซ้ำ

การวนซ้ำ เป็นประเด็นสำคัญเรื่องหนึ่งที่มนุษย์อยากให้คอมพิวเตอร์ทำงานแทน งานด้านวิศวกรรมหลาย ๆ ครั้งจำเป็นต้องใช้ Excel มาช่วย และบ่อยครั้งที่ทุก ๆ สัปดาห์ หรือทุก ๆ เดือน วิศวกรต้องมานั่งหน้าคอมพิวเตอร์ แล้วทำอะไรซ้ำ ๆ ครั้งละ หลาย ๆ บรรทัด ปัญหาดังกล่าวจะลดลงหากวิศวกรรู้จักใช้คำสั่งการทำงานวนซ้ำ เพื่อให้คอมพิวเตอร์ทำหน้าที่แทนเขา

2.1 การใช้ คำสั่ง For Next เบื้องต้น

คำสั่ง For Next เป็นคำสั่งที่กำหนดให้คอมพิวเตอร์ทำงานซ้ำตามจำนวนครั้งที่กำหนด โดยมีหลักการดัง**ภาพที่ 2.1** เริ่มต้น For j=1 to 10 เป็นการกำหนดให้ j ทำงานซ้ำ 10 รอบ โดยแต่ละรอบ j= 1 หลังจากนั้นจะทำงานทีละบรรทัดจนถึงคำว่า next j เมื่อถึงบรรทัดดังกล่าวแล้ว โปรแกรมจะกระโดษกลับมาบรรทัด For อีกครั้ง ในรอบถัดไป j=2 และจะทำงานจากบนลงล่างทีละบรรทัด ไปเรื่อง ๆ เมื่อ เจอ next j ก็จะกลับมาที่ for ซ้ำ ๆ จน j=10 ก็ยังคงทำอยู่เช่นเดิม เมื่อ j=11 ก็จะหยุดการทำงาน



ภาพที่ 2.1 การเขียน For next แบบทั่วไป

การวนซ้ำ for next ไม่จำเป็นต้องเพิ่มครั้งละ 1 เสมอไป อาจเพิ่มครั้งลละสอง เช่นการนับ 1 3 5 7 9 จนถึง 15 หรือนับถอยหลังจาก 10 9 8 7 จนถึง 1 ก็ได้ หรือนับถอยหลังจาก 10 ลงไปเรื่อย ๆ จนถึงค่าที่กำหนดก็ได้ดังแสดงใน**ภาพที่ 2.2** (a ,b และ c) ทั้งสามขั้นตอนสามารถกระทำได้โดยเติมคำว่า step เข้าไปต่อท้าย เช่น step -1 หรือ -2 เป็นต้น

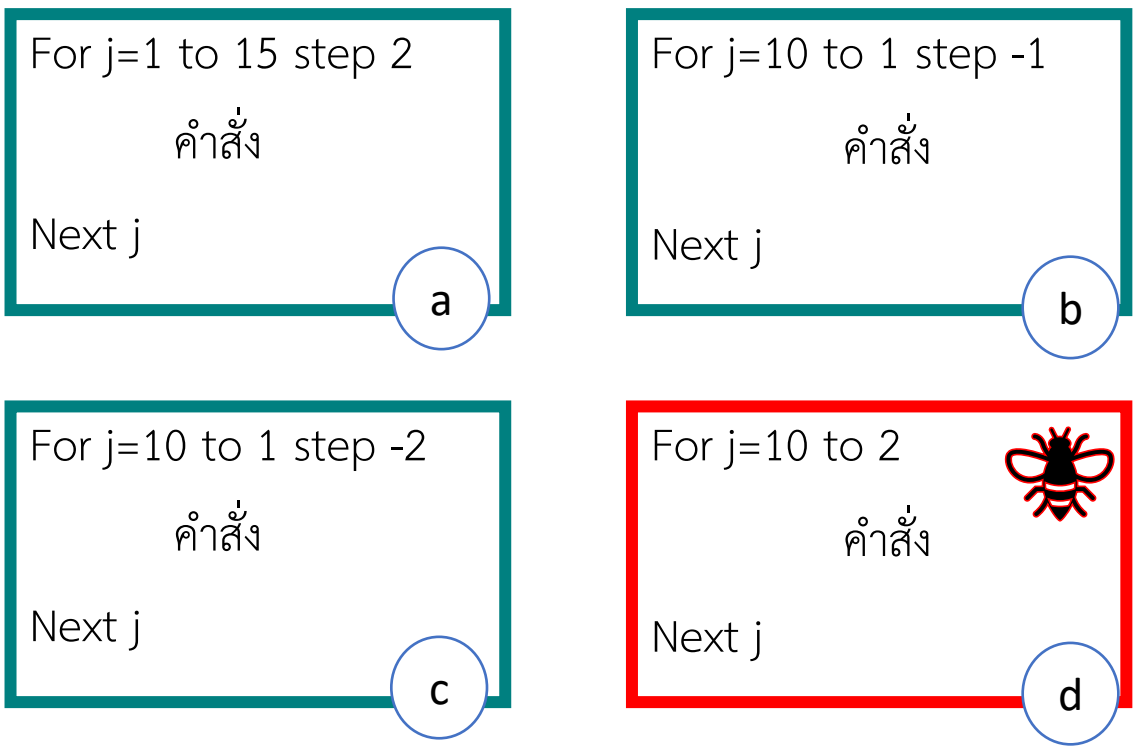
จากตัวอย่าง

ภาพ a ค่า j จะมีค่าตั้งแต่ 1 3 5 7 9 10 11 13 15

ภาพ b ค่า j จะมีค่าตั้งแต่ 10 9 8 7 6 5 4 3 2 1

ภาพ c ค่า j จะมีค่าตั้งแต่ 10 8 6 4 2

ในกรณีที่นับถอยหลังแต่ step ไม่ติดเครื่องหมายลบ ดังภาพที่ 2.2 d คอมพิวเตอร์จะไม่สามารถเข้าไปทำงานในการวนซ้ำ for next ครั้งนั้นได้



ภาพที่ 2.2 การเขียน For next แบบอื่น ๆ

เพื่อให้ได้ผลลัพธ์ดัง**ภาพที่ 2.3** สดมภ์แรกซึ่งมีเลข 1 ทั้งหมด 6 บรรทัดสามารถเขียน code ได้สองลักษณะคือเขียนแบบเยอะ เมื่อไม่ใช้คำสั่ง for next และเขียนแบบน้อย โดยใช้คำสั่ง for next ดังแสดงใน**ภาพที่ 2.4 และ 2.5** ตามลำดับ ถ้าต้องการผลลัพธ์เป็นเลข 1 จำนวน 6 บรรทัด สามารถทำได้โดยเขียนคำสั่งลักษณะเดียวกันทั้ง 6 บรรทัดโดยทำการเปลี่ยนค่าใน Cells(1,1) จนถึง Cells (6,1) ดังแสดงใน**ภาพที่ 2.4** การทำดังกล่าวต้องเขียนคำสั่งซ้ำเดิมจำนวนมาก และเปลี่ยนตัวเลขแค่เพียงจุดเดียว (ดูจะเป็นการลำบากถ้าหากต้องการ 100 บรรทัด) อย่างไรก็ดีผู้อ่านสามารถเขียนคำสั่งให้ได้ผลลัพธ์เดิมแต่ทำสั่งสั้นลงได้ดังแสดงใน**ภาพที่ 2.5**

	A	B	C
1	1	1	item1
2	1	2	item2
3	1	3	item3
4	1	4	item4
5	1	5	item5
6	1	6	item6

ภาพที่ 2.3 ผลลัพธ์ที่ต้องการ

```
Public Sub lab6_1()
    Sheets(1).Cells(1, 1).Value = 1
    Sheets(1).Cells(2, 1).Value = 1
    Sheets(1).Cells(3, 1).Value = 1
    Sheets(1).Cells(4, 1).Value = 1
    Sheets(1).Cells(5, 1).Value = 1
    Sheets(1).Cells(6, 1).Value = 1
End Sub
```

ภาพที่ 2.4 การเขียนไม่ใช้ for

จาก**ภาพที่ 2.5** จุดใดที่ต้องการเปลี่ยนตัวเลขตั้งแต่เลข 1 จนถึง 6 สามารถทำได้โดยนำตัวแปร j ไปแทนที่ และ สั่งให้ j เริ่มต้นจาก 1 ถึง 6 และปิดคำสั่งด้วย next j

```
Public Sub lab6_1()
    Dim j As Integer
    For j = 1 To 6
        Sheets(1).Cells(j, 1).Value = 1
    Next j
End Sub
```

ภาพที่ 2.5 การเขียนโดยใช้ for

รูปแบบการเขียนถัดมาเป็นตัวอย่างการเขียนให้ได้ผลลัพธ์ดังภาพที่ 2.6 ผลลัพธ์ที่ได้จะมีตัวเลขเรียงกันตามลำดับตั้งแต่ 1 ถึง 6 ผลดังกล่าวสามารถเขียนคำสั่งได้ สองวิธีดังภาพที่ 2.7 และ 2.8

	A	B	C
1	1	1	item1
2	1	2	item2
3	1	3	item3
4	1	4	item4
5	1	5	item5
6	1	6	item6

ภาพที่ 2.6 ผลลัพธ์ที่ต้องการ

```
Public Sub lab6_2()
        Sheets(1).Cells(1, 2).Value = 1
        Sheets(1).Cells(2, 2).Value = 2
        Sheets(1).Cells(3, 2).Value = 3
        Sheets(1).Cells(4, 2).Value = 4
        Sheets(1).Cells(5, 2).Value = 5
        Sheets(1).Cells(6, 2).Value = 6
End Sub
```

ภาพที่ 2.7 การเขียนแบบยาว

```
Public Sub lab6_2()
    Dim j As Integer
    For j = 1 To 6
        Sheets(1).Cells(j, 2).Value = j
    Next j
End Sub
```

ภาพที่ 2.8 การเขียนแบบสั้น

ภาพที่ 2.7 เป็นการเขียนแบบยาวโดยไม่ใช้คำสั่งวนลูป จำเป็นต้องเขียนทั้งหมด 6 บรรทัดตามจำนวนแถวที่มี ส่วน**ภาพที่ 2.8** เป็นการเขียนแบบวนลูปโดยใช้ for การเขียนดังกล่าวจะใช้ โคร์ดคำสั่งสั้นมาก ในภาพดังกล่าวมีจุดน่าสังเกตุ 2 จุดคือ ประกาศตัวแปร j เป็น integer และนำค่า j ไปแทนในตัวเลขสองตำแหน่งในส่วนของ แถว และหลังเท่ากับ Sheets(1).Cells(j,2).Value=j

ตัวอย่างการเขียนวนลูป เรียงลำดับลงมาที่คำที่เขียนประกอบด้วยตัวอักษรและตัวเลขเรียงตามลำดับ หากต้องการเขียนข้อความที่มีตัวอักษรเรียงกันดัง**ภาพที่ 2.9** ในสดมภ์ c สามารถทำได้**ดังภาพที่ 2.10**

	A	B	C
1	1	1	item1
2	1	2	item2
3	1	3	item3
4	1	4	item4
5	1	5	item5
6	1	6	item6

ภาพที่ 2.9 ผลลัพธ์ที่ต้องการ

เริ่มต้นมีการประกาศตัวแปร j เป็นจำนวนเต็มตามปกติ หลังจากนั้นกำหนดวนลูปการทำงานซ้ำ 6 ครั้งจาก for j=1 to 6 ปิดท้ายด้วยคำสั่ง next j จุดสังเกตุที่น่าสนใจในการเขียนโปรแกรมคือ การเขียนตัวอักษรต่อกับค่า j ใช้คำสั่ง “item” & j เพื่อทำการต่อคำว่า item กับค่าของ j ที่เปลี่ยนไปเข้าด้วยกัน

```
Public Sub lab6_3()
    Dim j As Integer
    For j = 1 To 6
        Sheets(1).Cells(j, 3).Value = "item" & j
    Next j
End Sub
```

ภาพที่ 2.10 การเขียนคำสั่ง

ตัวอย่างการรวมคำสั่ง จาก 3 ชุดคำสั่งที่กล่าวมาตอนต้นจากภาพที่ 2.5 2.8 2.10 ทั้งหมดสามารถรวมชุดทำสั่งได้ภายใต้การวนซ้ำ j เพียงครั้งเดียว โดยเบื้องต้นให้ประกาศตัวแปร j เป็น จำนวนเต็ม สั่งวนซ้ำลูป j ทั้งหมด 6 รอบ หลังจางนั้นทำชุดคำสั่งทั้งหมดมาเขียนรวมไว้ในครังเดียวดัง **ภาพที่ 2.11**

```
Public Sub lab6_all()
        Dim j As Integer
        For j = 1 To 6
                Sheets(1).Cells(j, 1).Value = 1
                Sheets(1).Cells(j, 2).Value = j
                Sheets(1).Cells(j, 3).Value = "item" &
        Next j
End Sub
```

ภาพที่ 2.10 การเขียนรวมคำสั่ง

2.2 การใช้คำสั่ง for ที่ซับซ้อน

การเขียนโปรแกรมติดต่อกับ excel ยิ่งฝึกเขียนปัญหาที่ซับซ้อนก้จะยิ่งทำให้มีทักษะในการพลิกแพลงการเขียนได้มากยิ่งขึ้นในตัวอย่างถัดมา ตัวอย่างแสดงการเขียนที่ซับซ้อนยิ่งขึ้นดังภาพที่ 2.11

	A	B	C	D
1	11	20	2	1
2	12	19	4	3
3	13	18	6	5
4	14	17	8	7
5	15	16	10	9
6	16	15	12	11
7	17	14	14	13
8	18	13	16	15
9	19	12	18	17
10	20	11	20	19

ภาพที่ 2.11 ผลลัพธ์ที่ซับซ้อน

จากภาพที่ 2.11 ในสดมภ์แรก เป็นการเรียงลำดับตัวเลขตั้งแต่ 11 ถึง 20 ในแถวที่ 1 ถึง 10 สดมภ์ที่ 2 เป็นการนับถอยหลังตั้งแต่ 20 จนถึง 11 ในสดมภ์ถัด ๆ ไปยิ่งมีตัวเลขคนละชุดกัน ดังนั้นจากตัวอย่างนี้จะทำให้ผู้อ่านให้ทดลองคิดหาวิธีที่จะทำให้ได้ตัวเลขต่าง ๆ ตามที่กำหนด ซึ่งการจะเขียนสดมภ์แรกจะไม่สามารถเขียนได้ดังภาพที่ 2.12 เพราะจะทำให้เกิดผลลัพธ์ผิดเพี้ยนไปจากเดิม (ดังแสดงในภาพที่ 2.13)

```
Public Sub lab7_1()
    Dim j As Integer
    For j = 11 To 20
        Sheets(1).Cells(j, 1).Value = j
    Next j
End Sub
```

ภาพที่ 2.12 ตัวอย่างคำสั่งที่ผิดพลาด

การจะทำให้ผลลัพธ์ออกมาตามที่ต้องการจำเป็นต้องมีการคิดเลขในใจทดลองผิดทดลองถูกเสียก่อนโดยนำแถวที่จะแสดงผล และค่าของผลลัพธ์มาเปรียบเทียบกัน จากการคิดคำนวนพบว่า เมื่อ j เริ่มต้นที่ 11 ให้เขียนในแถวที่ 1 และเมื่อ j มีค่า 12 ให้เขียนในแถวที่ 2 จากความสัมพันธุ์ดังกล่าวสามารถปรับปรุงการเขียนโปรแกรมได้ใหม่ดัง **ภาพที่ 2.14 a**

	A	B
1		
2		
3		
4		
5		
6		
7		
8		
9		
10		
11	11	
12	12	
13	13	
14	14	
15	15	
16	16	
17	17	
18	18	
19	19	
20	20	

ภาพที่ 2.13 ผลลัพธ์ที่ผิดพลาด

```
Public Sub lab7_1()
        Dim j As Integer
        For j = 11 To 20
                Sheets(1).Cells(j - 10, 1).Value = j
        Next j
End Sub
```

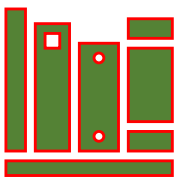
a

```
Public Sub lab7_2()
        Dim j As Integer
        For j = 1 To 10
                Sheets(1).Cells(j, 1).Value = j + 10
        Next j
End Sub
```

b

ภาพที่ 2.14 ตัวอย่างคำสั่งที่ถูกต้อง

สำหรับภาพ 2.14 b เป็นการเขียนอีกลักษณะหนึ่งโดยการวนลูป j จะเริ่มตั้งแต่ 1 จนถึง 10 แต่การแสดงผลจะมีการเพิ่มตัวเลขเพิ่มไปอีก 10 เมื่อผู้อ่านสังเกตุเห็นทั้ง 2 วิธีจะเข้าใจได้ว่าการเขียนให้ได้ผลลัพธ์อย่างเดียวกันสามารถเขียนได้หลายวิธีตามแต่ จะออกแบบให้เหมาะกับแต่ละบุคคล



แบบฝึกหัด

ข้อ 2.1 จงเขียน code ภายใต้ชื่อ

public sub lab2.1

.....

end sub

เพื่อให้เกิดผลลัพธ์ดังนี้

	A	B	C
1	1	1	item1
2	1	2	item2
3	1	3	item3
4	1	4	item4
5	1	5	item5
6	1	6	item6

ข้อ 2.2 จงเขียน code ภายใต้ชื่อ

public sub lab2.2

.....

end sub

เพื่อให้เกิดผลลัพธ์ดังนี้

	A	B	C	D
1	11	20	2	1
2	12	19	4	3
3	13	18	6	5
4	14	17	8	7
5	15	16	10	9
6	16	15	12	11
7	17	14	14	13
8	18	13	16	15
9	19	12	18	17
10	20	11	20	19

ข้อ 2.3 จงเขียน code ภายใต้ชื่อ

public sub lab2.3

.....

end sub

เพื่อให้เกิดผลลัพธ์ดังนี้

	A	B	C	D	E	F	G	H	I	J
1	1	2	3	4	5	6	7	8	9	10
2	1	2	3	4	5	6	7	8	9	10
3	1	2	3	4	5	6	7	8	9	10
4										
5										
6										
7										
8										

ข้อ 2.5 จงเขียน code ภายใต้ชื่อ

public sub lab2.5

.....

end sub

	A	B	C	D	E	F	G	H	I	J
1	1	2	3	4	5	6	7	8	9	10
2	1	2	3	4	5	6	7	8	9	10
3	1	2	3	4	5	6	7	8	9	10
4	1	2	3	4	5	6	7	8	9	10
5	1	2	3	4	5	6	7	8	9	10
6	1	2	3	4	5	6	7	8	9	10
7	1	2	3	4	5	6	7	8	9	10
8	1	2	3	4	5	6	7	8	9	10

ข้อ 2.6 จงเขียน code ในการรวมคะแนน midterm และ final ภายใต้ชื่อ

public sub lab2.6

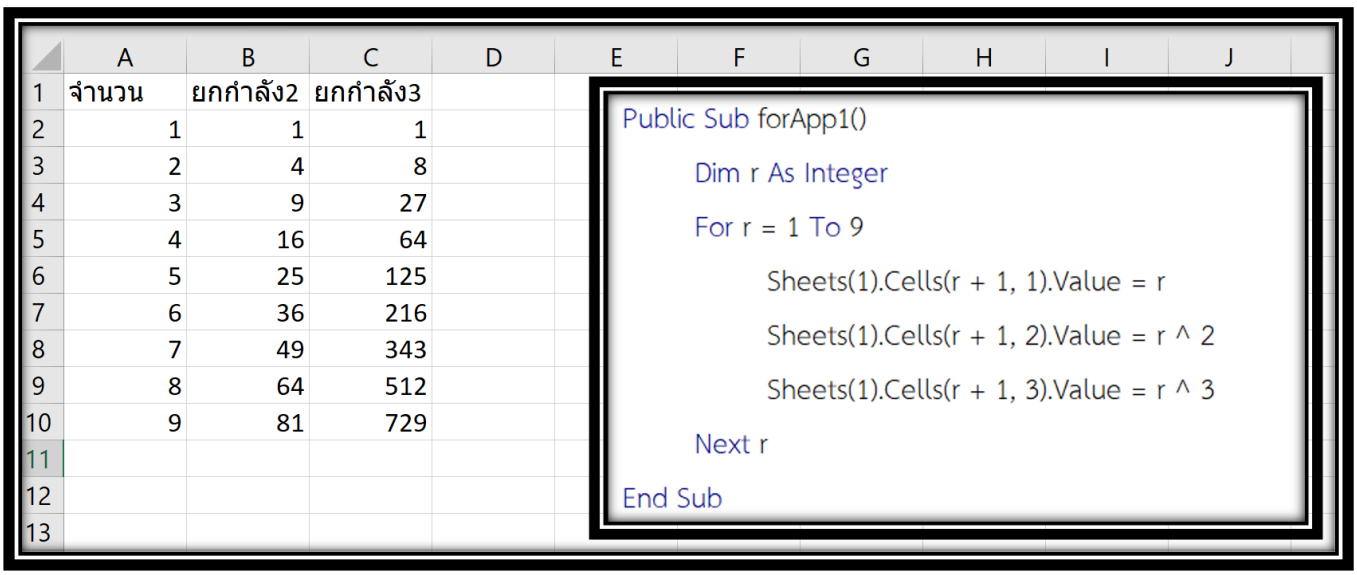
.....

end sub

	A	B	C	D
1	StudentID	midterm	Final	Total
2	Student 1	50	15	
3	Student 2	20	10	
4	Student 3	25	45	
5	Student 4	5	25	
6	Student 5	40	25	
7	Student 6	10	15	
8	Student 7	30	35	

การใช้ for กับการคำนวณ

ตัวอย่างถัดมาเป็นการเขียนโปรแกรมโดยใช้คำสั่ง for ทำงานซ้ำ ๆ เพื่อคำนวณอะไรบางอย่าง เช่น ต้องการหาเลขยกกำลังสองและสามของตัวเลขตั้งแต่ 1 ถึง 9 โดยแสดงวิธีการดังภาพที่ 2.15 จากภาพเป็นการทำงานซ้ำ 9 รอบแต่ละรอบค่า r จะเปลี่ยนค่าจาก 1 จนถึง 9 เพื่อคำนวณและจะแสดงผลในบรรทัดที่ 2 จนถึง 10 จากคำสั่ง "cells(r+1," ในตัวอย่าง การยกกำลัง 2 และ 3 แสดงด้วย r^2 และ r^3 ตามลำดับ



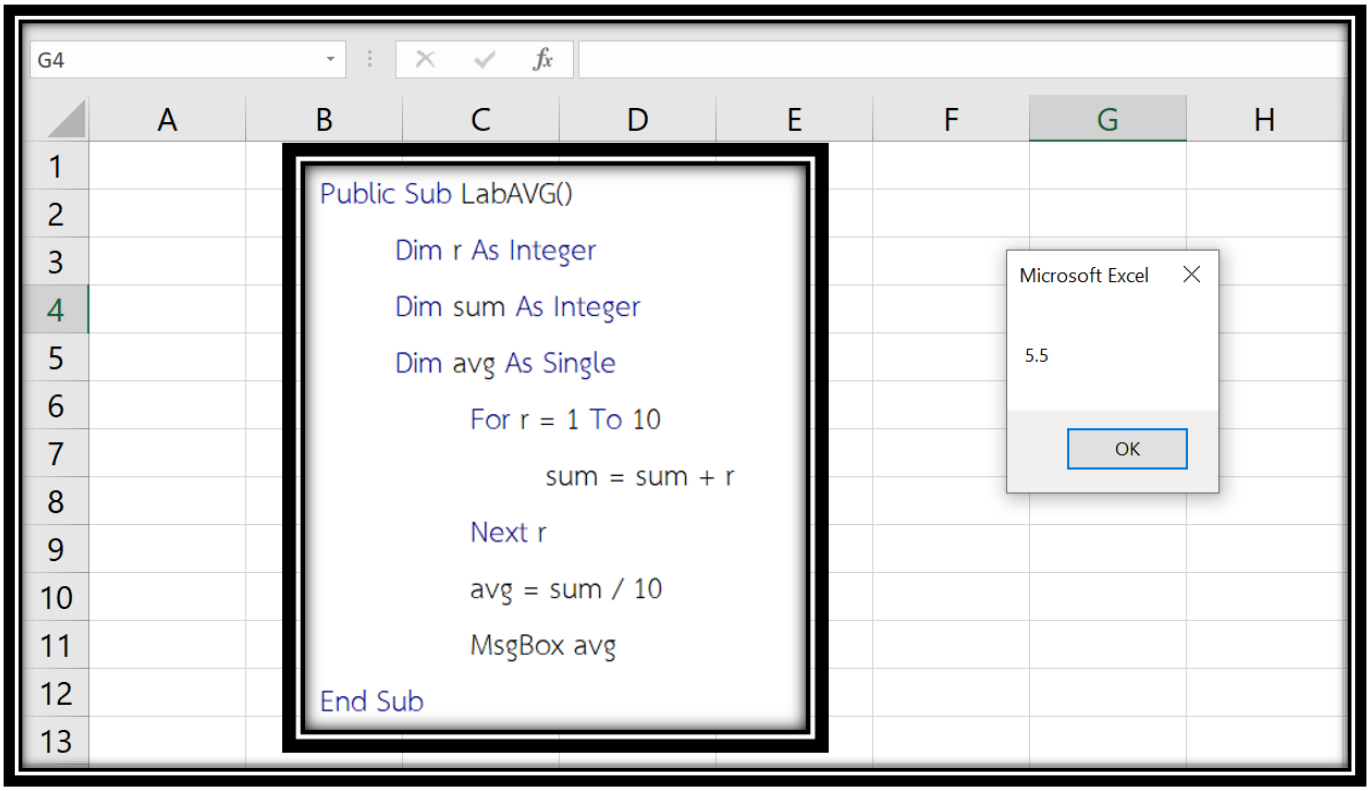
	A	B	C
1	จำนวน	ยกกำลัง2	ยกกำลัง3
2	1	1	1
3	2	4	8
4	3	9	27
5	4	16	64
6	5	25	125
7	6	36	216
8	7	49	343
9	8	64	512
10	9	81	729
11			
12			
13			

```
Public Sub forApp1()
    Dim r As Integer
    For r = 1 To 9
        Sheets(1).Cells(r + 1, 1).Value = r
        Sheets(1).Cells(r + 1, 2).Value = r ^ 2
        Sheets(1).Cells(r + 1, 3).Value = r ^ 3
    Next r
End Sub
```

ภาพที่ 2.15 การทำซ้ำยกกำลังสองและสาม

เพื่อแสดงให้เห็นว่าคำสั่ง for มีประโยชน์มากในการเขียนโปรแกรมตัวอย่างถัดมาผู้เขียนจะนำเสนอการหาค่าเฉลี่ยของผลรวมตัวเลขตั้งแต่ 1 จนถึง 10 ซึ่งเป็นตัวอย่างที่เข้าใจได้ง่ายและเป็นพื้นฐานที่จำเป็นในการเขียนโปรแกรมเพื่อคำนวณค่าต่างๆทางวิทยาศาสตร์ ตัวอย่างการหาค่าเฉลี่ยแสดงดังภาพที่ 2.15

ในตัวอย่างประกาศตัวแปรมา 3 ตัวคือ ตัวแปร r สำหรับวนซ้ำการทำงาน ตัวแปร sum สำหรับหาผลรวม สำหรับตัวแปรทั้งสองควรเลือกใช้ integer สำหรับเก็บข้อมูล ตัวแปรตัวสุดท้ายคือ avg ผู้เขียนแนะนำว่าควรใช้ single สำหรับเก็บข้อมูลเพราะมีโอกาสเกิดเลขทศนิยมจากการคำนวณ



ภาพที่ 2.15 การทำซ้ายกกำลังสองและสาม

จากภาพที่ 2.15 ประเด็นที่น่าสนใจสำหรับการหาผลรวมคือคำสั่ง for r=1 to 10 ใช้สำหรับวนซ้ำ 10 ครั้ง แต่ละครั้งค่า r จะเริ่มจาก 1 ถึง 10 และคำสั่ง sum = sum +r เป็นการสะสมค่าที่นำไปเก็บไว้ในตัวแปร sum ผลการคำนวณในแต่ละรอบแสดงดังตารางที่ 2.1

รอบที่ 1 ค่า r=1 ค่า sum เริ่มจาก 0 คำนวณ sum+r ได้ 1 แล้วเอาไปเก็บในตัวแปร sum

รอบที่ 2 ค่า r=2 ค่า sum เริ่มจาก 1 คำนวณ sum+r ได้ 3 แล้วเอาไปเก็บในตัวแปร sum

รอบที่ 3 ค่า r=3 ค่า sum เริ่มจาก 3 คำนวณ sum+r ได้ 6 แล้วเอาไปเก็บในตัวแปร sum

การทำซ้ำจะดำเนินการไปเรื่อย ๆดังตารางที่ 2.1 จนกระทั้งรอบสุดท้าย รอบที่ 10 ค่า r=10 ค่า sum เริ่มจาก 45 คำนวณ sum+r ได้ 55 แล้วเอาไปเก็บในตัวแปร sum

ตารางที่ 2.1 ลำดับการหาผลรวมแต่ละรอบ

รอบที่	ค่า r	ค่า sum เริ่มต้น	ค่า sum ล่าสุด
1	1	0	1
2	2	1	3
3	3	3	6
4	4	6	10
5	5	10	15
6	6	15	21
7	7	21	28
8	8	28	36
9	9	36	45
10	10	45	55

อ้างอิงภาพที่ 2.15 เช่นเดิม หลังจากได้ค่า sum แล้วก็นำไปหาค่าเฉลี่ยด้วยคำสั่ง avg=sum/10 แล้วแสดงผลลัพธ์ด้วยกล่องข้อความจากคำสั่ง MsgBox แล้วตามด้วยชื่อตัวแปร avg เช่น MsgBox avg เป็นอันเสร็จสิ้นขั้นตอนของโปรแกรม

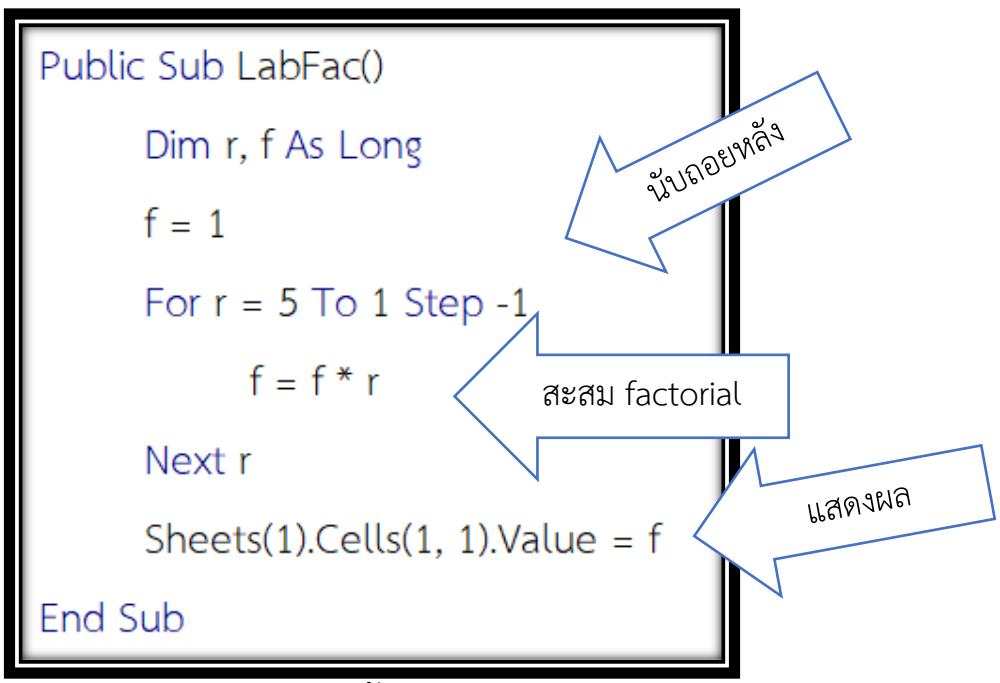
การเขียนโปรแกรมคำนวณ factorial ด้วยคำสั่ง for

หากผู้อ่านเข้าใจตัวอย่างที่แล้วเป็นอย่างดี การเขียนโปรแกรมหา factorial ก็ไม่ไช่เรื่องที่เขียนโปรแกรมยากแต่อย่างใด ก่อนอื่นขอทบทวนการหาค่า factorial อย่างง่ายเสียก่อน ค่าแฟคตอเรียลคำนวณจากผลคูณนับถอยหลังของตัวเลขจำนวนเต็มบวกปัจจุบันจนถึง 1 เช่น

$$5!=5x4x3x2x1$$

$$7!=7x6x5x4x3x2x1$$

เป็นต้น การเขียนชุดคำสั่งก็ใกล้เคียงกับการหาผลรวม แต่แตกต่างกันเพียงเครื่องหมายเริ่มต้นที่เป็นเครื่องหมายคูณ ดังแสดงในภาพที่ 2.16 และการนับที่นับถอยหลัง จึงใช้คำสั่ง for r=5 to 1 แล้วตามด้วย step -1 เพื่อให้คอมพิวเตอร์นับเลขถอยหลัง ในตัวอย่างตัวแปรถูกกำหนดเป็น Long เนื่องจาก มีโอกาสที่ตัวแปร integer จะเก็บค่าการคำนวณได้ไม่ครบถ้วนเนื่องจากตัวเลขเพิ่มขึ้นอย่างรวดเร็ว



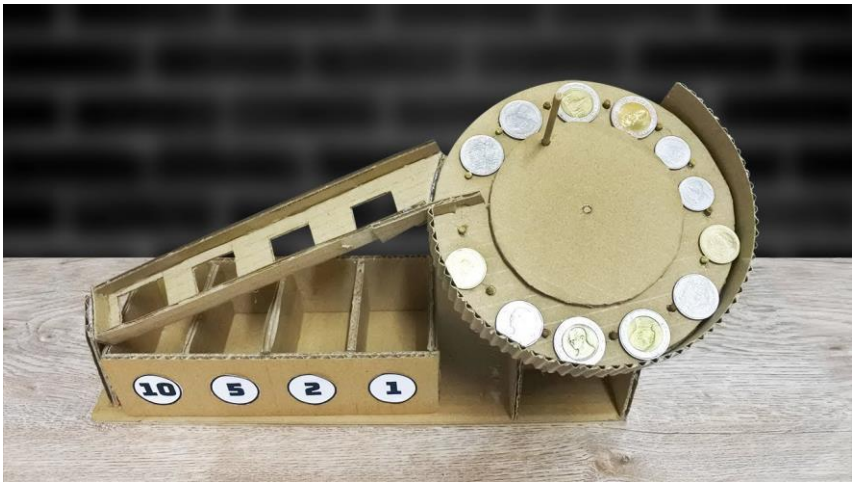
ภาพที่ 2.16 การทำซ้ำยกกำลังสองและสาม

ชุดคำสั่งที่ซับซ๊อน

การใช้คำสั่ง IF สำหรับตรวจสอบหลายๆ เงื่อนไข

ในชุดคำสั่งนี้จะตรวจสอบเงื่อนไขแรกก่อนถ้าเป็นจริงจะทำคำสั่งที่ 1 และ 2 แล้วออกมาที่ end if ทันที แต่ถ้าเงื่อนไขแรกเป็นเท็จจึงมาตรวจสอบเงื่อนไขที่ 2 และถ้าเงื่อนไขที่ 2 เป็นเท็จ จึงจะมาตรวจสอบเงื่อนไขที่ 3 ตามลำดับ สุดท้ายถ้าไม่มีเงื่อนไขใดเป็นจริงเลยจะทำคำสั่งที่ 7 และ 8 แล้วไปที่ end if

ชุดคำสั่งดังกล่าวอุปมาดั่งเครื่องแยกเหรียญโดยเหรียญแต่ละอันจะถูกลำเลียงไปยังรูขนาดต่างๆ ถ้าขนาดเล็กกว่ารูใดเหรียญก็จะตกลงรูนั้น แต่ละรูปก็เปรียบเหมือน if และ elseif แต่ละกรณีที่ตรวจสอบดังภาพที่ 2.17 โครงสร้างและตัวอย่างของ IF แบบหลายทางแสดงดังภาพที่ 2.18 และ 2.19 ในหน้าถัดไป



ที่มา https://youtu.be/P7dNyhzDGfw

ภาพที่ 2.17 เครื่องแยกเหรียญ

ภาพที่ 2.18 โครงสร้าง if แบบหลายทาง

```
Public Sub GradeABCDF ()
    Dim score As Integer = 82
    Dim grade As String
    If score > 80 Then
        grade = "A"
    ElseIf score > 70 Then
        grade = "B"
    ElseIf score > 60 Then
        grade = "C"
    ElseIf score > 40 Then
        grade = "D"
    Else
        grade = "f"
    End If
    MsgBox (grade)
End Sub
```

ภาพที่ 2.19 ตัวอย่าง if แบบหลายทาง

Select case

ในบางครั้งเราต้องการตรวจสอบตัวแปรแค่เพียง 1 ตัวแปร แต่มีหลายเงื่อนไขให้ตรวจสอบ ชุดคำสั่ง select case จึงเหมาะที่จะนำมาใช้งานในกรณีดังกล่าว โครงสร้างและตัวอย่างการใช้งานตัดเกรดแสดงดังภาพที่ 2.20 และ 2.21 ตามลำดับ

```
Select case ตัวแปร
    Case แรก
        ชุดคำสั่งที่ 1
    Case สอง
        ชุดคำสั่งที่ 2
    Case สาม
        ชุดคำสั่งที่ 3
    Case else
        ชุดคำสั่งที่ 4
End select
```

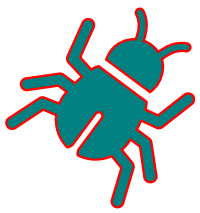
ภาพที่ 2.20 โครงสร้าง Select case

```
Dim s As Integer
    s = 82
    Select Case s
        Case Is > 80
            Msgbox "A"
        Case Is > 70
            Msgbox "B"
        Case Is > 60
            Msgbox "C"
        Case Else
            Msgbox "D"
    End Select
```

ภาพที่ 2.21 ตัวอย่าง Select case

การกำหนดแบบที่ 2 (ภาพที่ 2.22) การกำหนดต่อไปนี้โปรแกรมจะให้ผลผิดพลาดเนื่องจากการเรียงลำดับผิด เพราะ s=82 จะมีค่ามากกว่า 60 จึงสงผลให้ ได้เกรด C ทันทีและไม่ได้เกรด A เหตุผลเนื่องจาก select case จะเลือกเพียงกรณีใดกรณีหนึ่งแล้วข้ามไปยังคำสั่ง end select ทันที

```
Public Sub BadProgram
Dim s As Integer
    s = 82
    Select Case s
        Case Is > 60
            MessageBox.Show("c")
        Case Is > 70
            MessageBox.Show("B")
        Case Is > 80
            MessageBox.Show("A")
        Case Else
            MessageBox.Show("D")
    End Select
End Sub
```



82>60

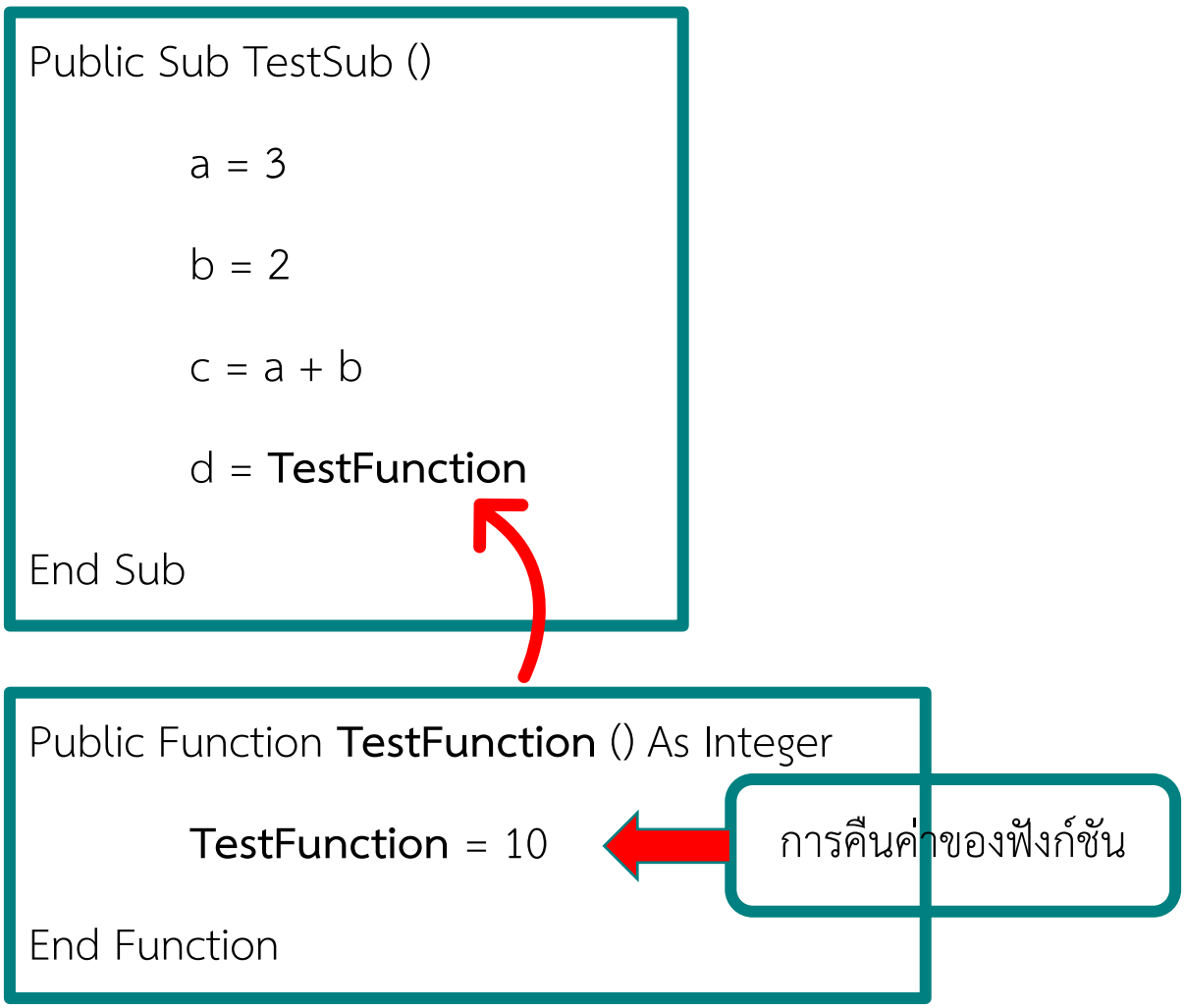
82>70

82>80

ภาพที่ 2.22 ตัวอย่าง Select case ที่ผิดพลาด

ซับและฟังก์ชัน Sub and function

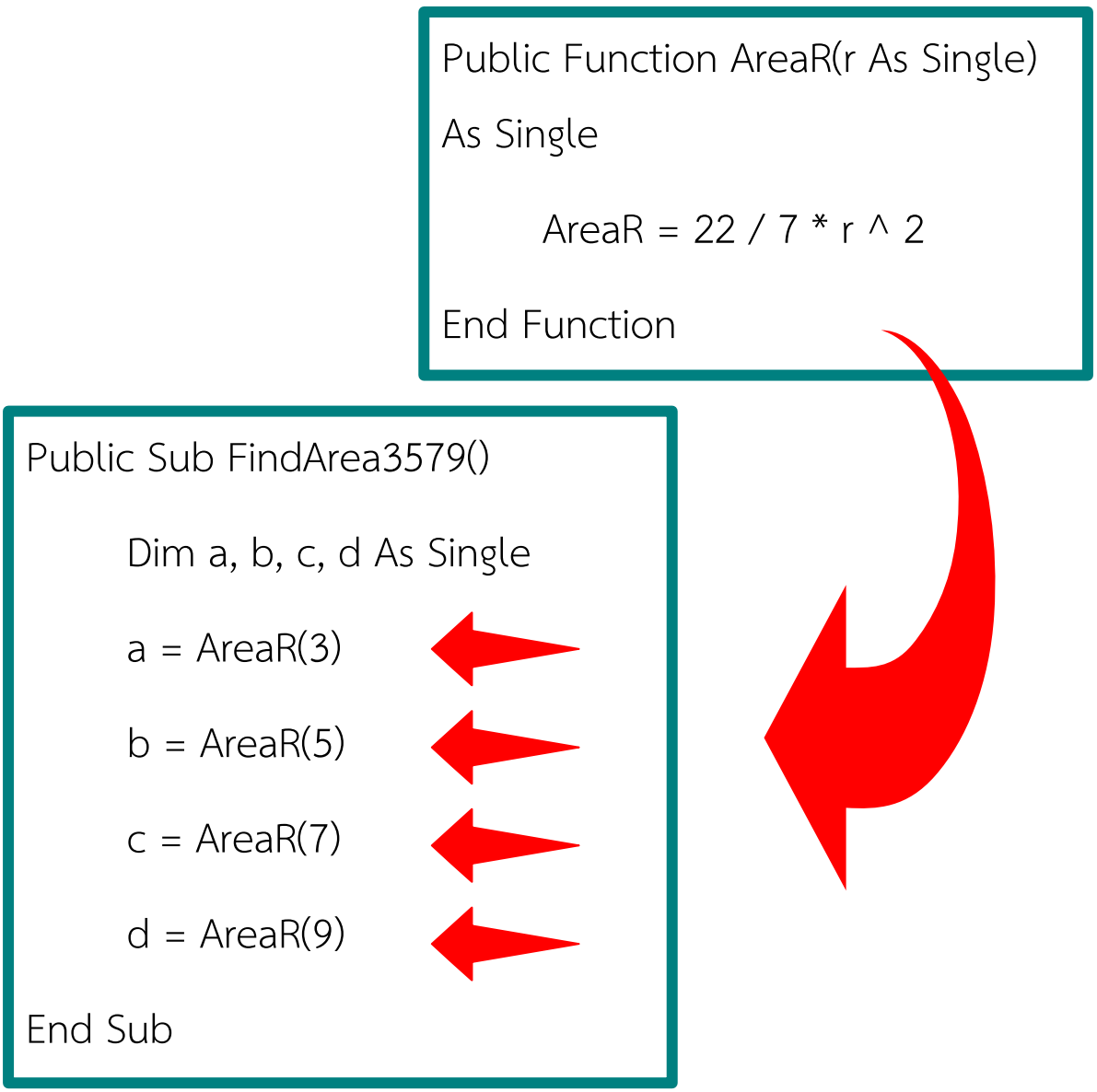
ในการเขียนโปรแกรม Visual basic ในเวอร์ชันต่าง ๆ สิ่งที่ขาดไม่ได้คือ sub และฟังกชัน โดย sub จะเป็นชุดคำสั่งในการทำงาน ส่วนฟังก์ชันคือชุดคำสั่งที่มีการคืนค่ากลับมาให้โปรแกรมหลัก ภาพที่ 2.23 แสดงการเรียกฟังก์ชันของ TestSub และการคืนค่าของฟังก์ชัน Test Function สำหรับ TestSub พบว่าตัวแปร a b c รับค่าเป็นตัวเลขโดยตรง ส่วนตัวแปร d รับค่าจากฟังก์ชัน ข้อสังเกตุที่สอง การคืนค่าออกจากฟังก์ชั่น TestFunction จะคืนค่าโดยการอ้างถึงชื่อฟังก์ชัน



ภาพที่ 2.23 การเรียกฟังก์ชันและการคืนค่าฟังก์ชัน

ตัวอย่าง จงหาพื้นที่ของวงกลมรัศมี 3 5 7 9 ตามลำดับ

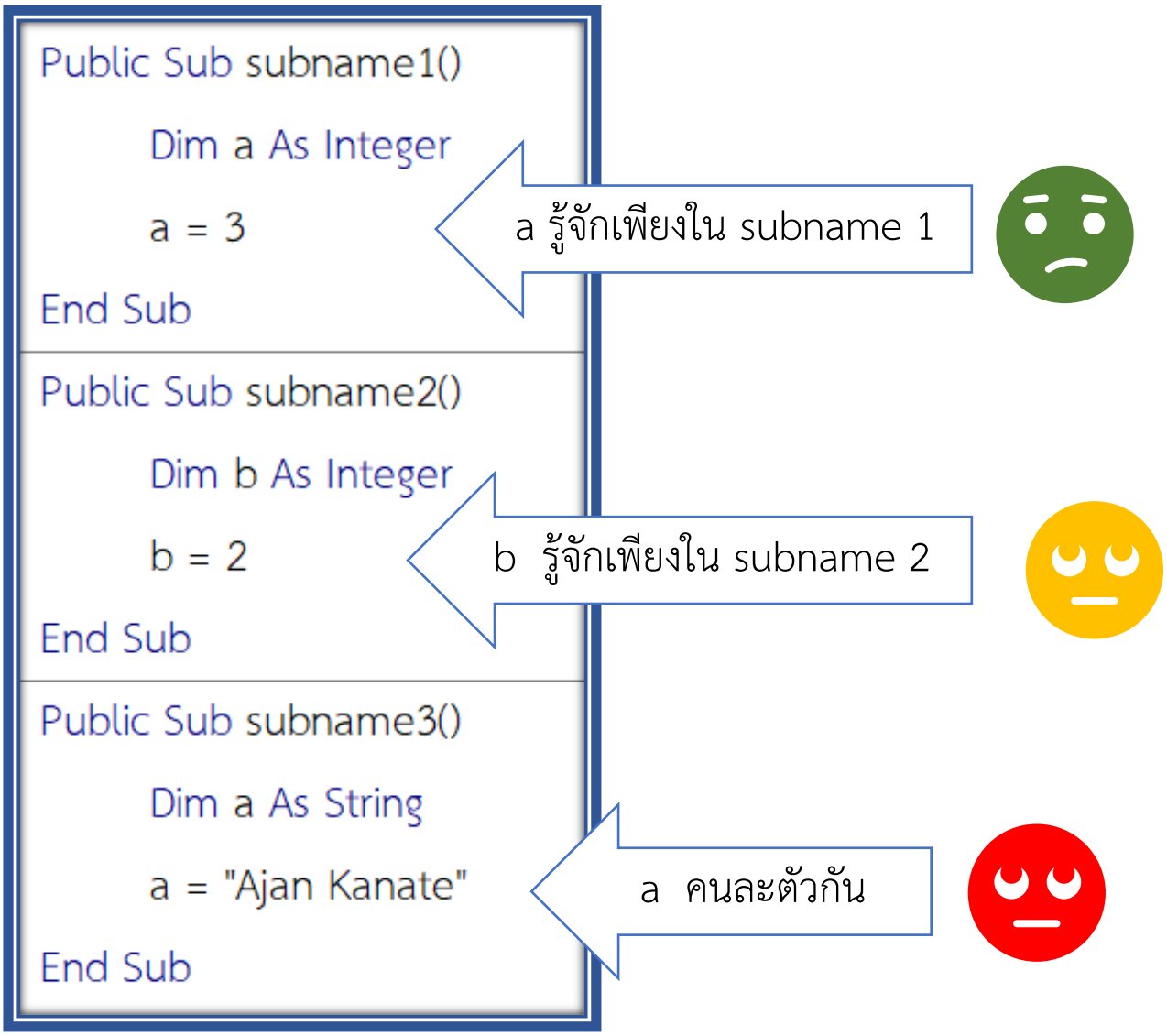
จากภาพที่ 2.24 แสดงการเรียกใช้ ฟังก์ชั่น AreaR ผ่านซับ Area3579 โดยฟังก์ชันดังกล่าวรับตัวแปรเข้า 1 ตัวแปร โดยเก็บไว้ที่ตัวแปร r (r as integer) หลังจากนั้นนำค่า r ไปคำนวณ สังเกตุว่าฟังก์ชันดังกล่าวเรียกใช้ได้หลาย ครั้ง เพียวแต่กำหนดค่า r ใหม่ทุก ๆ ครั้งที่เรียก โดยเริ่มจาก 3 5 7 และ 9 ตามลำดับ



ภาพที่ 2.24 การเรียกฟังก์ชันพื้นที่วงกลมใช้งาน

ขอบเขตของตัวแปร

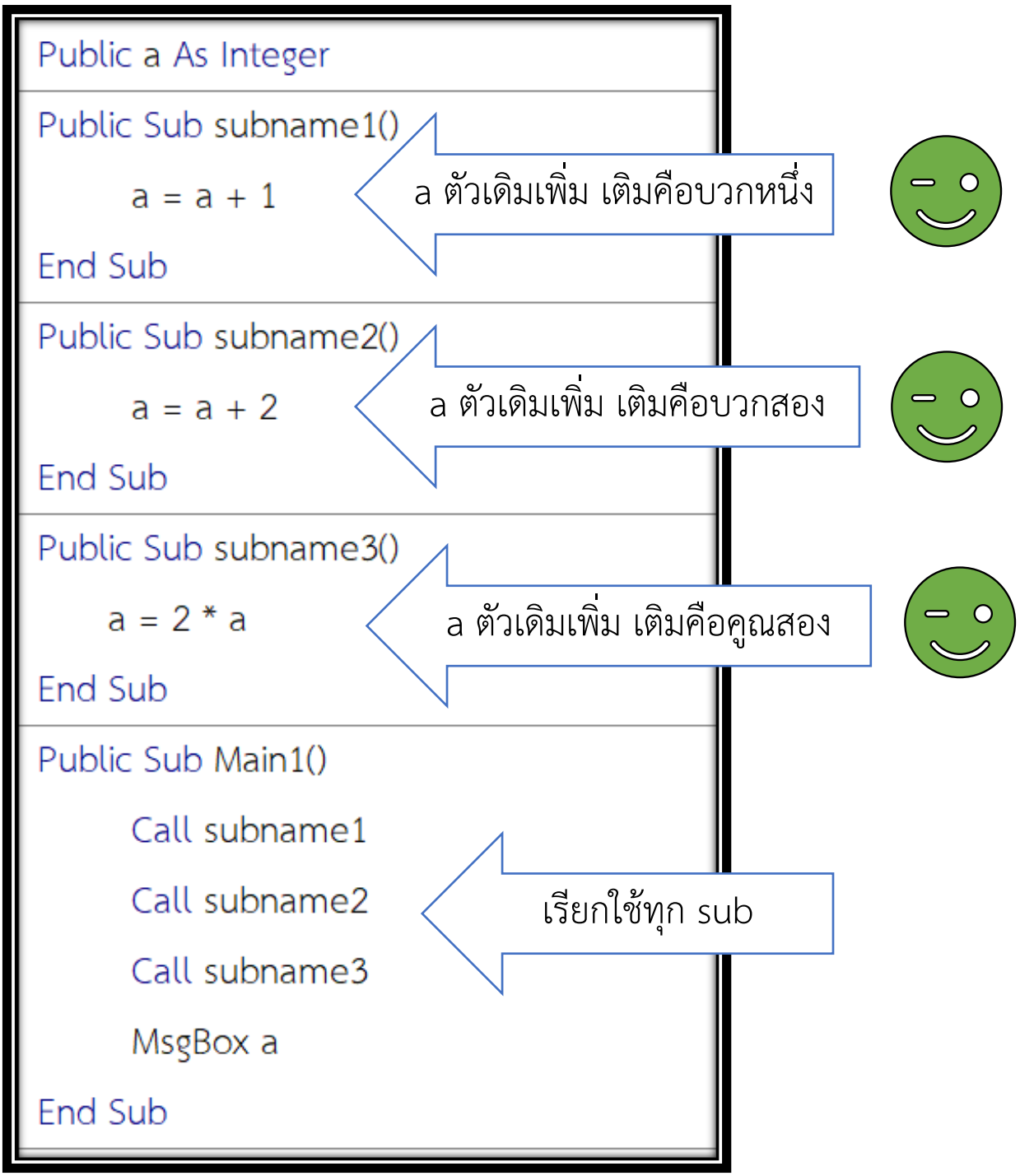
การประกาศตัวแปร ด้วยคำสั่ง dim สามารถเรียกใช้ได้ในแต่ละ sub หรือฟังก์ชัน นั้น ๆที่ทำการประกาศ แต่ sub หรือ ฟังก์ชันอื่นจะไม่รู้จัก และไม่สามารถเรียกใช้ได้ตัวอย่างดังภาพที่ 2.25



ภาพที่ 2.25 ตัวแปรไม่รู้จักกัน

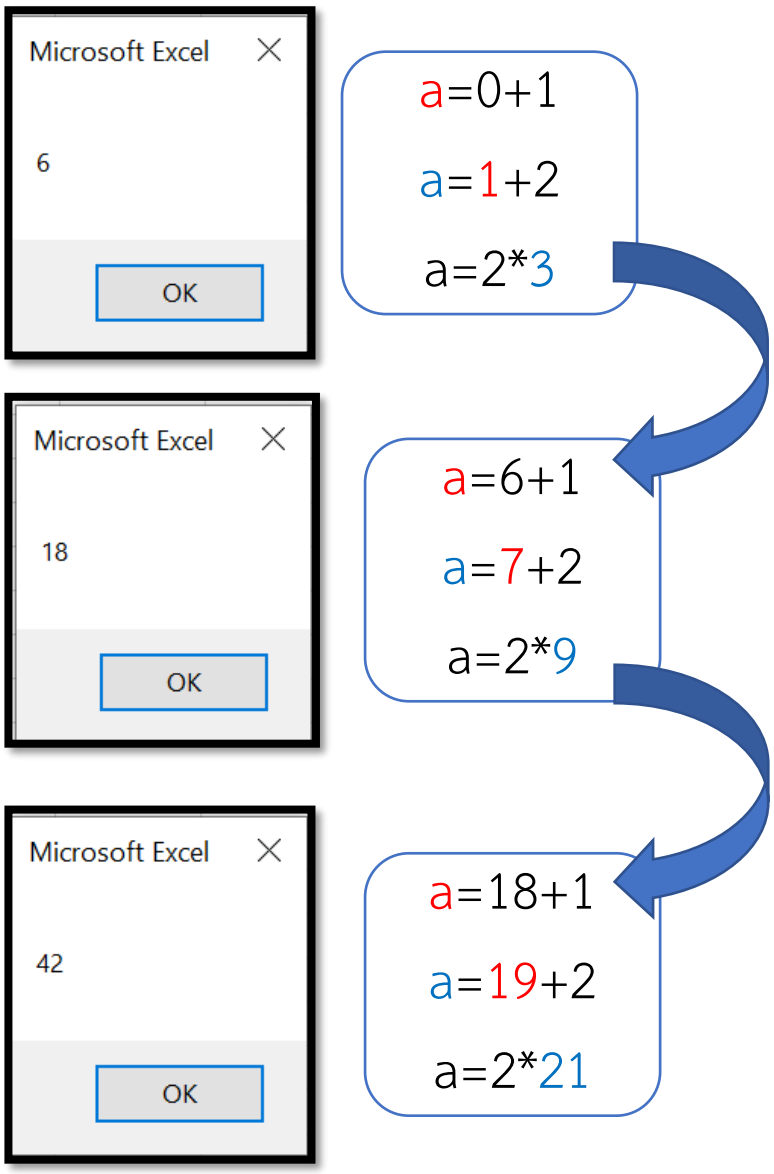
ขอบเขตของตัวแปร(ต่อ)

หากต้องการตัวแปรใดรู้จักกันทั้งหมดในทุก sub (หรือทุก ฟังก์ชัน) สามารถทำได้โดยประกาศตัวแปรไว้ภายนอก และใช้คำว่า private แทนทำว่า dim ดังภาพที่ 2.26 สำหรับ ฟังก์ชัน main1 ใช้สำหรับสั่งการเรียกใช้งาน



ภาพที่ 2.25 ตัวแปรรู้จักกันทั้งหมด

จากภาพที่ 2.25 เมื่อเรียกใช้งานsub main1 จะมีกล่องข้อความปรากฏค่า 6 ถ้ามีการเรียกใช้ sub main1 จำนวน 3 ครั้งจะให้ผลดังภาพที่ 2.26 และถ้าสั่งทำงานซ้ำเรื่อย ๆ ค่าจะเพิ่มขึ้นเรื่อย ๆ ค่าจะเริ่มต้นใหม่เมื่อสั่งหยุดการทำงาน โดยใช้ปุ่ม reset ตามภาพที่ 2.27

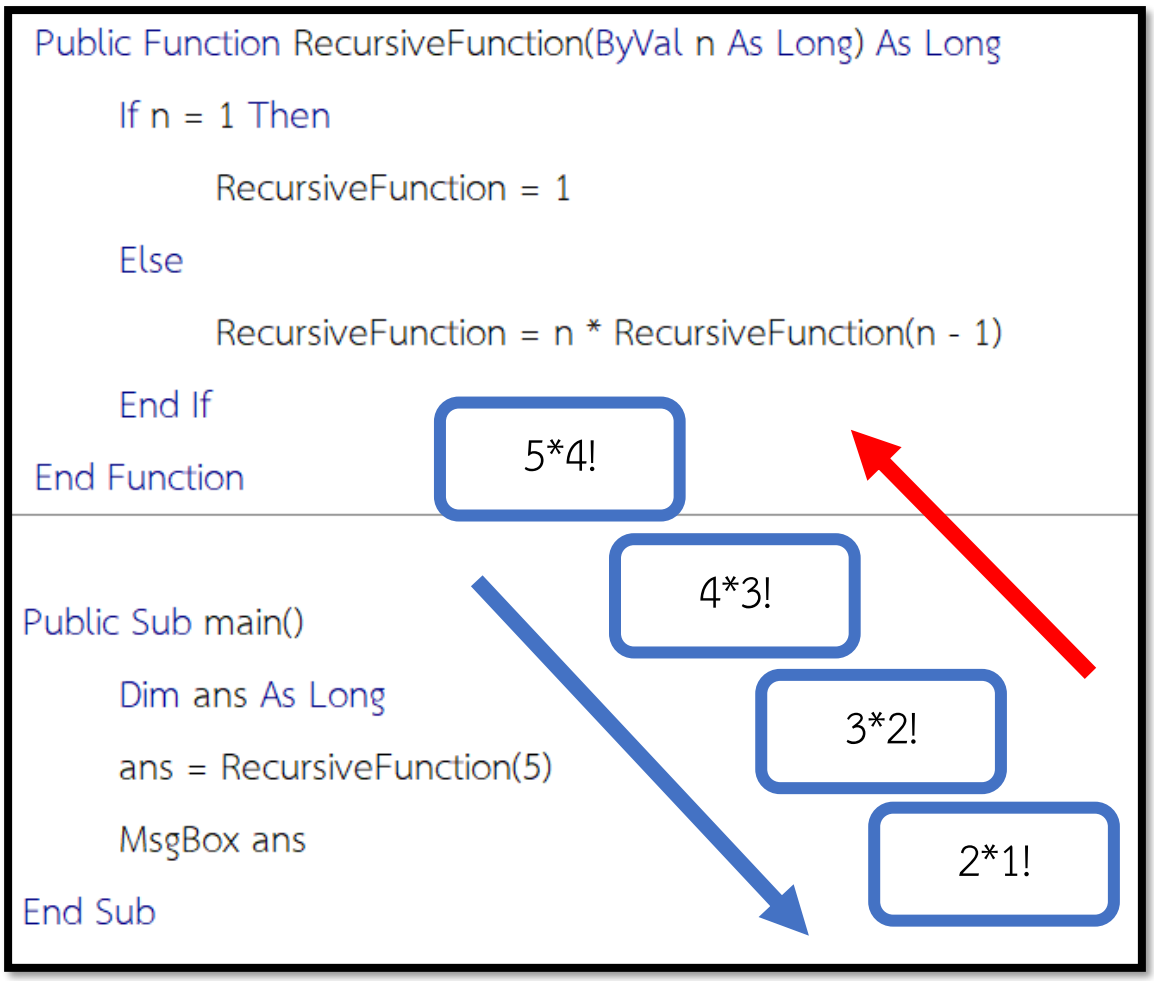


ภาพที่ 2.26 ผลลัพธ์การทำงานสามรอบ

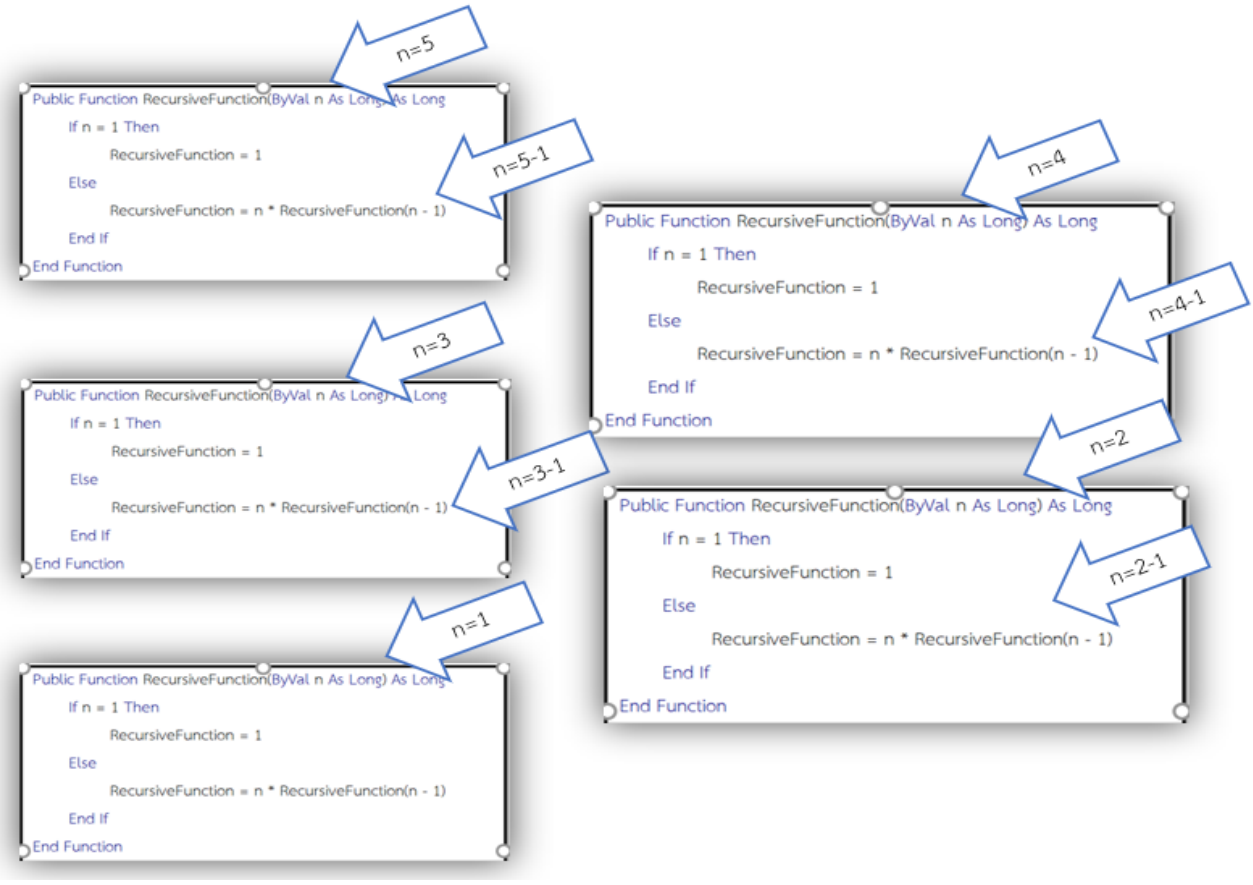
ฟังก์ชันเวียนเกิด (recursive function)

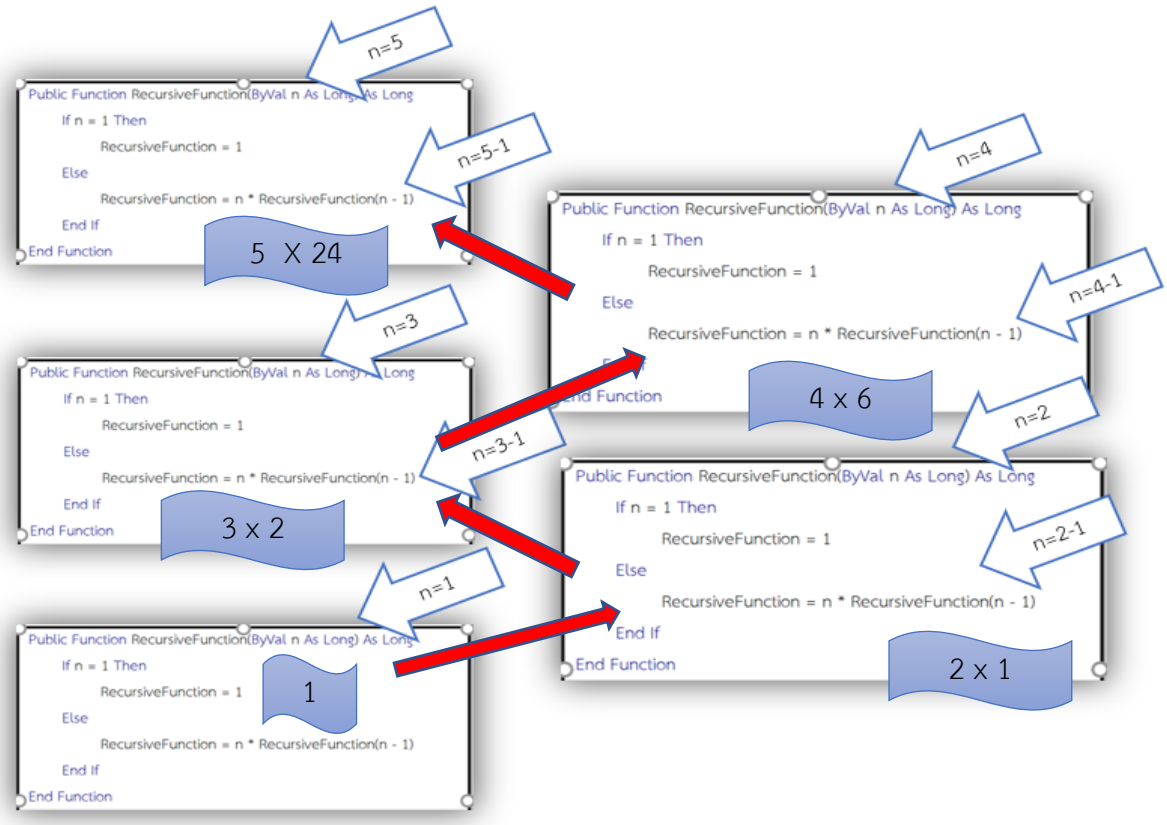
เมื่อผู้อ่านเข้าใจแล้วว่า แต่ละชุดคำสั่งสามาตรเรียกใช้กันได้โดยใช้คำว่า call และถ้าต้องการส่งผ่านค่าจาก ชุดคำสั่งต่าง ๆ ผ่านหากันได้โดยประการตัวแปรด้วยคำสั่ง private โดยประกาศตัวแปรไว้ภายนอกsub ยังมีอีกเทคนิคหนึ่งที่น่าใช้งานและซับซ้อนน่าสนุกสนานในการเรียนรู้ คือฟังก์ชันเวียนเกิด โดยฟังก์ชันดังกล่าวจะสามารถเรียกใช้ตัวมันเองซ้ำ ๆ ได้เรื่อย ๆ ตัวอย่างเช่นฟังก์ชันการหา factorial ดังภาพที่ 2.26

เริ่มจาก sub main ทำงาน เรียกฟังก์ชัน เวียนซ้ำครั้งที่ 1 ผ่านตัวแปร n = 5 หลังจากนั้น ฟังก์ชันดังกล่าวจะเรียกตัวเองซ้ำ ด้วย n=4 n=3 n=2 และ n=1 ตามลำดับ เมื่อเรียก n=1 แล้วค่าของฟังก์ชันจะมีค่าเป็น 1 และจะคืนค่ากลับมาเรื่อย ๆ ตามจำยวยครั้งที่มีการเรียกใช้งาน



ภาพที่ 2.26 ฟังก์ชันเวียนเกิด





ผู้ช่วยศาสตราจารย์ ดร.คเณศ พันธุ์สวาสดิ์

แบบฝึกหัด จงแก้ไข code ให้ถูกต้อง

```
Public Sub CorrectMe()
        Dim s As String
        Dim g As Integer
        s = "85"
        Select Case g
                Case Is >= 40
                        g = "C"
                Case Is >= 60
                        g = "B"
                Case Is >= 80
                        g = "A'"
                Case Else
                        g = "F"
        End Select
        MsgBox s
End Sub
```

แบบฝึกหัด จงเขียนโปรแกรมคำนวณช่องสีเขียวโดยใช้ฟังก์ชัน

	A	B	C	D
1		functionA1	functionA2	functionA3
2	ค่า R	เส้นรอบวงกลม	พื้นที่วงกลม	ปริมาตรลูกบาตร
3	1	6.29	3.14	1
4	3	18.86	28.29	27
5	5	31.43	78.57	125
6	7	44.00	154.00	343
7	9	56.57	254.57	729
8	11	69.14	380.29	1331
9	13	81.71	531.14	2197
10	15	94.29	707.14	3375
11	17	106.86	908.29	4913
12	19	119.43	1134.57	6859
13	21	132.00	1386.00	9261

แบบฝึกหัด ถ้าต้องการผลลัพธ์ตามที่กำหนดต้องเรียกโปรแกรมอะไรก่อนและหลัง

```
Public ans As Integer

Public Sub f1()
        ans = ans + 3
End Sub

Public Sub f2()
        ans = ans - 2
End Sub

Public Sub f3()
        ans = ans + 7
End Sub

Public Sub f4()
        ans = ans * 3
End Sub
```

```
Public Sub Brain1()
        Call
        Call
        Call
        MsgBox ans 'ขอผลลัพธ์ 15
End Sub

Public Sub Brain2()
        Call
        Call
        Call
        Call
        MsgBox ans 'ขอผลลัพธ์ 25
End Sub
```

ผู้ช่วยศาสตราจารย์ ดร.คเณศ พันธุ์สวาสดิ์

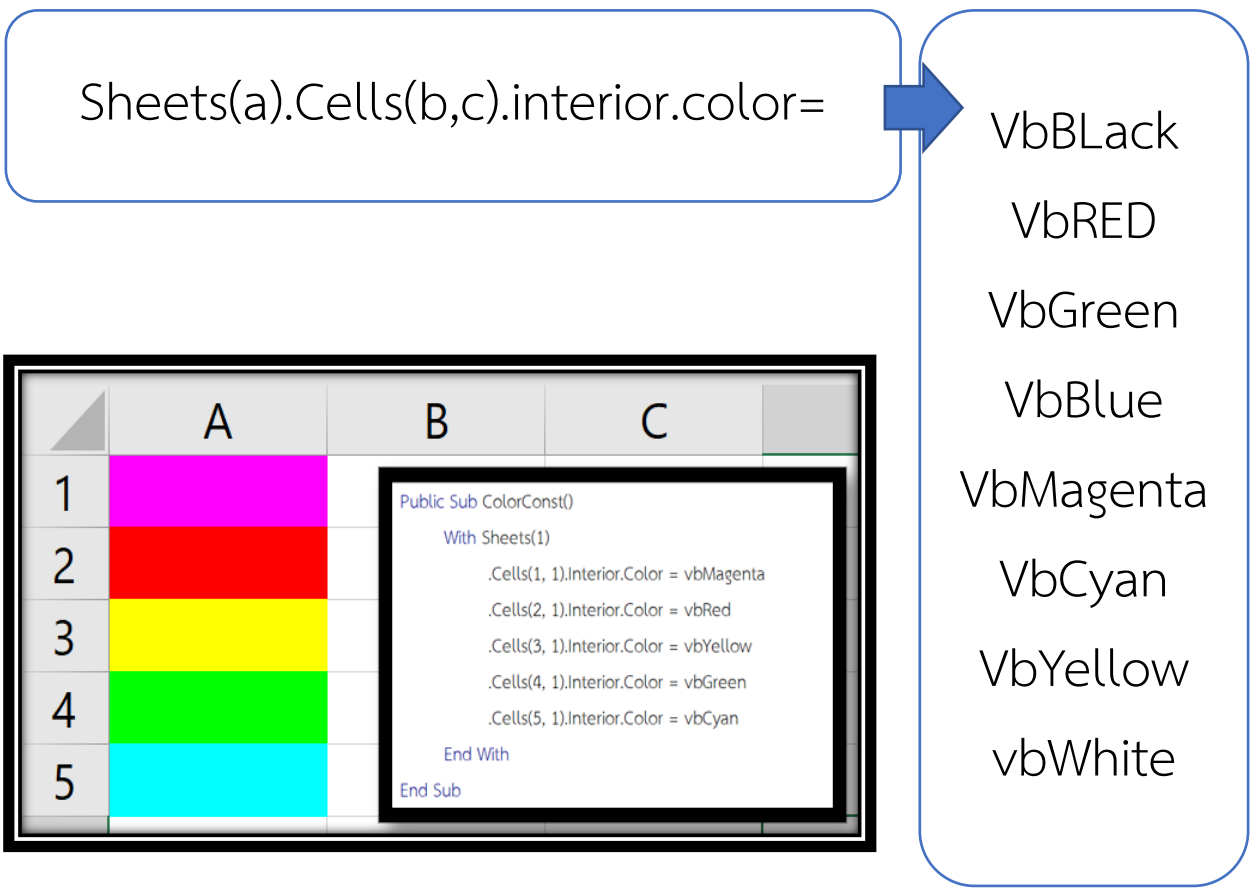
บทที่ 3 เรื่องของสี และการประยุกต์ใช้งาน

3.1 คำสั่งเรื่องของสี

การระบายสีลงในเซลล์ของ excel ทำได้หลายลักษณะ แต่ลักษณะที่ใช้นิยมใช้กันประกอบด้วย

- Cells(r,c).Interior.color =“ชื่อของสี”
- Cells(r,c).Interior.colorindex =หมายเลขของสี
- Cells(r,c).Interior.color =RGB(red,green,blue)

ดังแสดงในภาพที่ 3.1



ภาพที่ 3.1 การใช้ค่าคงที่ของสี

การระบายสีอาจจะใช้การอ้างด้วยค่าคงที่ของสีก็ได้ โดยจะใช้คำสั่ง Sheets(1).Cells(a,b).Interior.ColorIndex = แล้วตามด้วยหมายเลข (ตั้งแต่ 1 จนถึง 56) หมายเลขสีต่าง ๆ แสดงดังภาพที่ 3.2 หากผู้อ่านต้องการเขียนโปรแกรมแสดงสี ต่าง ๆ สามารถทำได้โดยเขียนโคร์ดดังภาพที่ 3.3 ตัวแปร a b แสดงการวนซ้ำ ตัวแปร a แทน แถวและตัวแปร b แทนหลัก ทุกครั้งที่ถึงบรรทัด c=c+1 ค่า c จะเพิ่มขึ้นหนึ่ง หลักงจากนั้นค่า c จะถูกนำไปกำหนดสีของเซลล์ด้วยคำสั่ง “Interior.ColorIndex”

	A	B	C	D
1		15	29	43
2	2	16	30	44
3	3	17	31	45
4	4	18	32	46
5	5	19	33	47
6	6	20	34	48
7	7	21	35	49
8	8	22	36	50
9	9	23	37	51
10	10	24	38	52
11	11	25	39	53
12	12	26	40	54
13	13	27	41	55
14	14	28	42	56

ภาพที่ 3.2 สีและหมายเลข

```
Public Sub ColorIndexTable()
        Dim a, b, c As Integer
        For b = 1 To 4
            For a = 1 To 14
                c = c + 1
                Sheets(1).Cells(a, b).Value = c
                Sheets(1).Cells(a, b).Interior.ColorIndex = c
            Next a
        Next b
End Sub
```

ภาพที่ 3.3 การเขียนโปรแกรมแสดงสี

ตัวอย่าง การเขียนโปรแกรมหาค่าสูงสุด แล้วระบายสีแดงลงไปที่ค่าสูงสุดด้วย ในภาพที่ 3.3 3.4 หลังจากนั้นจงเขียนโปรแกรมระบายสีภาพที่ 3.5 ตามสีของ ColorIndex

	A	B
1	month	demand
2	jan	210
3	feb	390
4	mar	300
5	apr	300
6	may	390
7	jun	270
8	jul	300
9	aug	120
10	sep	140
11	oct	140
12	nov	350
13	dec	130

Hint. วนหนึ่งรอบเพื่อหาค่าสูงสุด วนซ้ำรอบที่สอง เพื่อระบายสี

ภาพที่ 3.3 ค่าสูงสุดแบบหนึ่งมิติ

	A	B	C	D
1	22	2	16	51
2	45	21	44	1
3	29	23	17	9
4	25	42	23	5
5	46	33	3	13

ภาพที่ 3.4 ค่าสูงสุดแบบสองมิติ

ตัวแปรที่มักจะใช้กับคำสั่งวนซ้ำ

ในการวนซ้ำบางครั้งจำเป็นต้องสร้างตัวแปรลำดับขึ้นมา เนื่องจากการประกาศตัวแปรแบบ ปกติ เช่น

Dim a, b, c, d, f, g, h as integer

ยากที่จะนำไปใช้งานในการวนซ้ำ ดังนั้นผู้อ่านควรทำความรู้จักกับตัวแปรลำดับให้ชำนาญในการใช้งาน

ตัวแปรลำดับหนึ่งมิติ

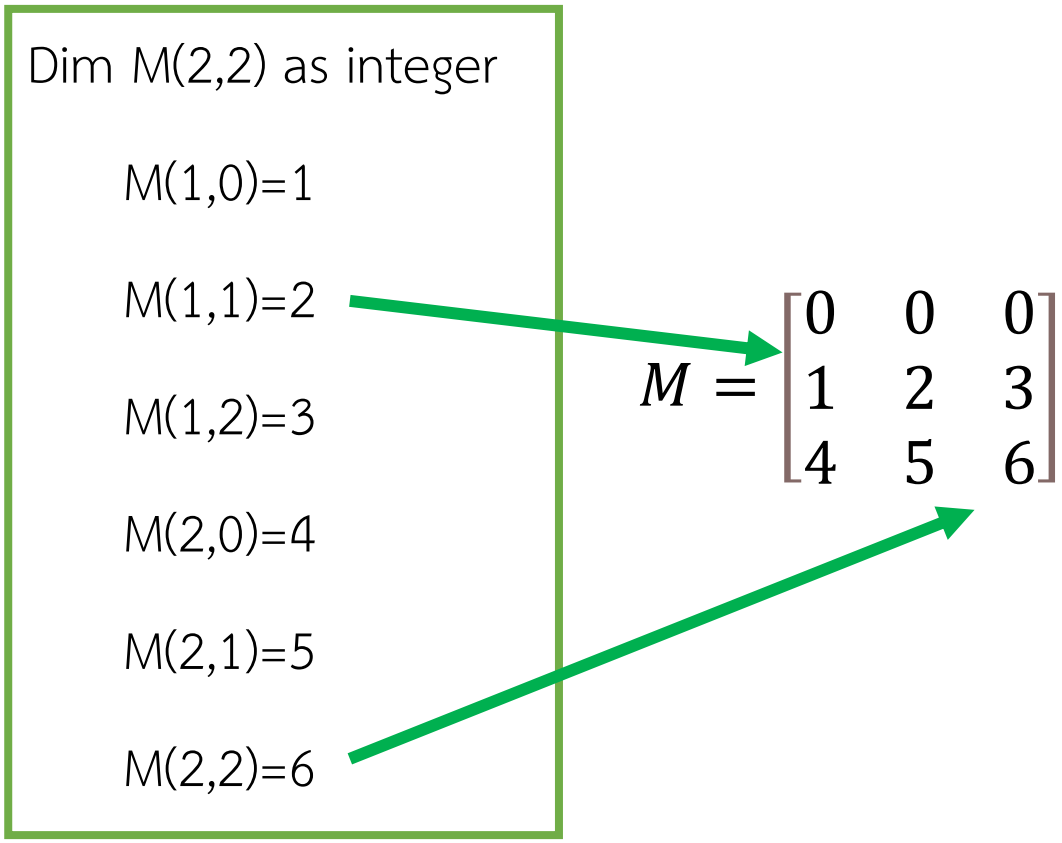
ตัวแปรลำดับ 1 หนึ่งมิติมีลักษณะ คล้ายกับ เวคเตอร์แถว หรือเวคเตอร์สดมภ์นั่นเอง ผู้อ่านอาจนึงภาพดังเวกเตอร์ต่อไปนี้จะเข้าใจได้ง่าย

$$A = [0 \quad 4 \quad 6] \text{ หรือ } A = \begin{bmatrix} 0 \\ 4 \\ 6 \end{bmatrix}$$

```
Dim A(2) as integer
    A(1)=4
    A(2)=6
```


ตัวแปรลำดับสองมิติ

ลักษณะการจัดเก็บมีความคล้ายคลึงกับ เมทริกที่ใช้งานในสายวิศวกรรมตัวอย่างเช่น



ตัวแปรลำดับแบบ ไดนามิค

การกำหนดตัวแปรแบบ dynamic array

```
Dim D() as integer
```

การกำหนดขนาด dynamic array ใหม่ โดยค่าเก่าที่เก็บไว้โดนล้างทั้งหมด

```
Redim D(10)
```

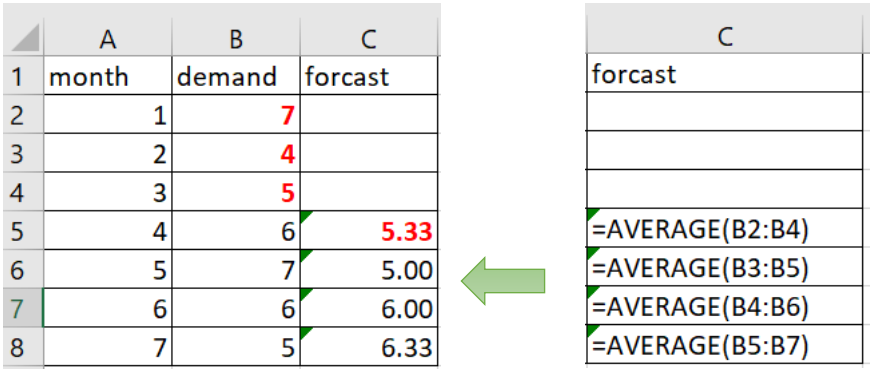
การกำหนดขนาด dynamic array ใหม่โดยค่าเก่าที่เก็บไว้ยังคงอยู่ครบถ้วน

```
Redim Preserve D(15)
```

บทที่ 3

การเขียนโปรแกรมกับการวางแผนผลิต

ในบทนี้จะพูดถึงการประยุกต์การเขียนโปรแกรมเพื่อใช้ในการวางแผนการผลิตโดยจะยกตัวอย่างง่าย ๆ เพื่อเป็นเบื้องต้นในการเรียนรู้การเขียนโปรแกรมเพื่อประยุกต์ใช้งาน โดยจะยกตัวอย่างการพยากรณ์แบบค่าเฉลี่ยเคลื่อนที่ (moving average forecasting) ซึ่งหลักการในการพยากรณ์คือ นำค่าเฉลี่ยของ สามเดือนก่อนหน้า ตัวอย่างเช่น พยากรณ์เดือนที่ 4 โดยใช้ค่าเฉลี่ยเดือนที่ 3 2 1 หากต้องการพยากรณ์เดือนที่ 5 ก็ต้องใช้ค่าเฉลี่ยของเดือนที่ 4 3 2 เป็นต้นแสดงดังภาพที่ 3.1



	A	B	C
1	month	demand	forcast
2	1	7	
3	2	4	
4	3	5	
5	4	6	5.33
6	5	7	5.00
7	6	6	6.00
8	7	5	6.33

C
forcast
=AVERAGE(B2:B4)
=AVERAGE(B3:B5)
=AVERAGE(B4:B6)
=AVERAGE(B5:B7)

ภาพที่ 3.1 การพยากรณ์แบบค่าเฉลี่ยเคลื่อนที่

สูตรการพยากรณ์แบบค่าเฉลี่ยเคลื่อนที่ แสดงไว้ดังนี้

กำหนดให้

ดัชนี

t แทนช่วงเวลาปัจจุบัน

n แทนจำนวนช่วงเวลาที่ใช้เฉลี่ย

พารามิเตอร์

F_t แทน การพยากรณ์ในเดือนที่ t

A_t แทน ข้อมูลจริงของช่วงเวลาที่ t

การพยากรณ์ค่าเฉลี่ยเคลื่อนที่คำนวณจากสมการต่อไปนี้

$$Ft=\frac{\sum_{t=1}^{n}(A_t)}{n}$$

หากเชื่อมโยงกับภาพที่ 3.1(ภาพก่อนหน้า) นั่นหมายถึง $F_4=(A_2+A_3+A_4)/n$ และ $F_5=(A_4+A_3+A_2)/n$

ขั้นตอนการในการเขียนโปรแกรม คิดก่อนว่าถ้าคำนวณด้วยมนุษย์ต้องทำอะไรก่อนอะไรหลัง เมื่อคิดแล้วพบว่า

ขั้นตอนที่ 1 นำค่า A สามค่ามารวมกันก่อน

ขั้นตอนที่ 2 นำผลรวมมาหาร n

ขั้นตอนที่ 3 ทำซ่ำเรื่อย ๆ จนครบทุกค่า F

ดังนั้นจึงสามารถแปลงจากขั้นตอนเป็นโปรแกรมได้ภาพที่ 3.2

	A	B	C
1	month	demand	forcast
2	1	7	
3	2	4	
4	3	5	
5	4	6	5.33
6	5	7	
7	6	6	
8	7	5	

```
Public Sub moving_average()
    Dim t, f As Single
    t = Sheets(1).Cells(2, "B").Value
    t = t + Sheets(1).Cells(3, "B").Value
    t = t + Sheets(1).Cells(4, "B").Value
    f = t / 3
    Sheets(1).Cells(5, "C").Value = f
End Sub
```

ภาพที่ 3.2 ตัวอย่างการเขียนโคร์ดแบบยังไม่เสร็จสมบูรณ์

เมื่อเขียนให้คำนวณค่าแรกได้แล้ว หลังจากนั้นควรเขียนโปรแกรมเพื่อวนซ้ำให้ทำงานจนครบทุกรอบ การจะวนซ้ำได้ต้องใช้ for และ เปลี่ยนหมายเลขบรรทัดที่เดิมเคยเป็นค่าคงที่ให้เป็นตัวแปรที่วนซ้ำ ในการเขียนจะเพิ่มตัวแปร r เพื่อใช้ในการวนซ้ำตั้งแต่บรรทัดที่ 5 ถึงบรรทัดที่ 8

```
Public Sub moving_average_complete()
    Dim t, f As Single
    Dim r As Integer
    For r = 5 To 8
        t = Sheets(1).Cells(r - 3, "B").Value
        t = t + Sheets(1).Cells(r - 2, "B").Value
        t = t + Sheets(1).Cells(r - 1, "B").Value
        f = t / 3
        Sheets(1).Cells(r, "C").Value = f
    Next r
End Sub
```

ภาพที่ 3.3 ตัวอย่างการเขียนโคร์ดแบบเสร็จสมบูรณ์

	A	B	C
1	month	demand	forcast
2	1	7	
3	2	4	
4	3	5	
5	4	6	5.33
6	5	7	5.00
7	6	6	6.00
8	7	5	6.33

ภาพที่ 3.4 ผลการโปรแกรม