ME 747 Introduction to computational fluid dynamics

Lecture 2

Introduction to Fortran programming

By Chainarong Chaktranond

Lecture schedule

Session	Topics
1	**1.Overviews of computational fluid dynamics** **-** Overviews and importance of heat transfer in real applications
2 - 3	**2. Introduction to Fortran programming** - Basic commands in Fortran programming
4	**3. Overviews of governing equations for flow and heat transfer** -Elliptic, Parabolic and Hyperbolic equations
5	**4. Introduction to numerical methods** - Finite different method, Finite volume method, Finite element method, etc.
6 – 7	**5. Introduction to solve engineering problems with finite-different method** - Taylor series expansion, Approximation of the second derivative, Initial condition and Boundary conditions

Contents

- Overviews of Fortran
- Basic commands in Fortran programming

Fortran development

- ภาษาฟอร์แทรน (FORTRAN - **FOR**mula **TRAN**slation)
- ค.ศ. 1954 ทีมนักคอมพิวเตอร์ บริษัท ไอบีเอ็ม (IBM) นำทีมโดย จอห์น แบคคัส (John Backus) เปิดตัวครั้งแรก ด้วย FORTRAN II และ FORTRAN IV
- ต่อมาได้พัฒนาภาษา เป็นมาตรฐานรุ่นแรก เรียกว่า FORTRAN-66
  - ไม่สามารถกำหนดชนิดข้อมูล
  - ไม่สามารถทำงานกับข้อมูลประเภทสายอักขระ
  - ไม่มีคำสั่งที่สามารถกำหนด โครงสร้างได้เหมาะสม
- FORTRAN-77 และ FORTRAN-88
- FORTRAN-90

โครงสร้างเลขฐานสอง (Binary scheme)

ตัวเลขที่ใช้ในเลขฐานสอง (Binary digits, bits) มีแค่สองตัวคือ 0 และ 1 เท่านั้น คำว่าบิต (bit) จะหมายถึงตำแหน่งของตัวเลขที่ใช้ในเลขฐานสอง เช่น 8 bits จะแทนตัวเลข 0 และ 1 ได้ 8

$10000000_2 = 1\times2^7 + 0\times2^6 + 0\times2^5 + 0\times2^4 + 0\times2^3 + 0\times2^2 + 0\times2^1 + 0\times2^0 = 128$

8 bits = 1 byte

2^{10} = 1024 bytes = 1K ของความจำ (Memory)

$512K = 512\times2^{10} = 2^9\times 2^{10} = 2^{19}$ = 524,288 bytes

หรือเทียบเท่า $2^{19} \times2^3 = 2^{22}$ = 4,194,304 bits

หน่วยความจำ

ความพิวเตอร์ส่วนมากเก็บเลขจำนวนเต็มในระบบตัวเลขฐานสอง เลขจำนวนเต็มหนึ่งจำนวนที่ใช้ที่เก็บในหน่วยความจำเท่ากับ 1 คอมพิวเตอร์เวิร์ด (Word) ซึ่งประกอบด้วยกี่บิตนั้นขึ้นอยู่กับระบบคอมพิวเตอร์ ทั้งนี้อาจมีตั้งแต่ 16 – 60 บิต

จำนวนบิตใน 1 เวิร์ด	ค่าสูงสุดของเลขจำนวนเต็ม	จำนวนหลักของเลขฐานสิบ
16	$2^{15}-1$	5
32	$2^{31}-1$	10
60	$2^{60}-1$	16

หน่วยความจำ

โดยขนาดของเวิร์ดที่ถูกกำหนดจะจำกัดช่วงของค่าจำนวนเต็ม (Integer) ซึ่งสามารถถูกเก็บไว้ภายใน ตัวอย่างเช่น ค่าจำนวนเต็มบวกที่มากที่สุดที่สามารถถูกเก็บในเวิร์ด 16 บิต คือ

$$0111111111111111_2 = 2^{15} - 1 = 32767$$

และตัวเลขลบที่น้อยที่สุดคือ

$$1000000000000000_2 = -2^{15} = -32768$$

ค่าของจำนวนเต็มที่ออกนอกช่วงที่กำหนดจะต้องการบิตมากกว่าที่สามารถถูกเก็บในเวิร์ดตัวหนึ่ง (A single word) ปรากฏการณ์นี้เรียกว่า overflow ข้อจำกัดดังกล่าวนี้สามารถแก้ไขได้โดยการใช้เวิร์ดมากกว่าหนึ่งเพื่อที่จะเก็บไว้ในเลขจำนวนเต็มตัวเดียว

การระบุค่าในโปรแกรมฟอร์แทรน

- เลขจำนวนเต็ม (Integer number)
- เลขจำนวนจริง (Real number)
- อักขระ (Character)

เลขจำนวนเต็ม (Integer number)

ค่าคงที่จำนวนเต็ม (Integer constant) เป็นตัวเลขที่เรียงแถวกัน ซึ่งไม่ต้องใส่เครื่องหมายลูกน้ำ “ , “ (Comma) และต้องเป็นตัวเลขที่ไม่จุดทศนิยม อาจจะเป็นค่าบวกหรือลบก็ได้ ตัวอย่างเช่น

0

137

-2516

+17745

ส่วนรูปแบบที่เขียนไม่ถูกต้องสำหรับการกำหนดค่าคงที่จำนวนเต็มเช่น

5,280 (ห้ามใส่ลูกน้ำ)

16.0 (ห้ามใส่จุดทศนิยม)

- - 5 (ใส่เครื่องหมายพีชคณิตได้เพียงตัวเดียว)

7 - (เครื่องหมายพีชคณิตต้องอยู่หน้าตัวเลข)

เลขจำนวนจริง (Real number)

ค่าคงที่จำนวนจริง (Real constant) หรือที่รู้จักทั่วไปว่าค่าความละเอียดเดี่ยว (Single precision data) ซึ่งสามารถกำหนดอยู่ในรูปของค่าที่มีจุดทศนิยมหรือค่าที่อยู่ในรูปเอ๊กซ์โปเนนเชียล (Exponential notation) ได้ โดยใส่เครื่องหมายจุดทศนิยมที่ด้านหลังตัวเลขตัวสุดในกรณีที่ไม่มีทศนิยม และการกำหนดจะต้องไม่ใส่เครื่องหมายลูกน้ำในระหว่างตัวเลขเช่นเดียวกับการกำหนดค่าคงที่จำนวนเต็ม ตัวอย่างเช่น

1.234

- .01536

+ 56473.

เลขจำนวนจริง (Real number)

ส่วนรูปแบบที่เขียนไม่ถูกต้องสำหรับการกำหนดค่าคงที่จำนวนจริงเช่น

12,345 (ห้ามใส่ลูกน้ำ)

63 (หลังตัวเลขตัวสุดท้ายต้องตามด้วยเครื่องหมายจุดทศนิยม)

การกำหนดค่าในรูปของค่าทางวิทยาศาสตร์ (Scientific representation) ซึ่งเป็นค่าคงที่จำนวนจริงประกอบด้วยค่าจำนวนเต็มหรือค่าที่มีจุดทศนิยมแล้วตามด้วยอักษร E และเลขยกกำลังตามลำดับ ตัวเช่นค่า 337.456 สามารถเขียนได้เป็น 3.37456E2 ซึ่งมีความหมาย 3.37456×10^2 หรืออาจจะเขียนอยู่ในรูปแบบอื่นๆ ได้อีกเช่น

0.337456E3

337.456E0

33745.6E-2

33745.6E-3

อักขระ (Character)

ซึ่งเรียกว่าค่า string เป็นลำดับของสัญลักษณ์ที่ถูกเลือกจากเซตอักขระของโปรแกรมฟอร์แทรน ค่าอักขระมาตรฐาน ANSI (American National Standards Institute standard character)

Character	Meaning
blank	blank or space
0, . . . , 9	digits
A, . . . , Z	uppercase letters
$	dollar sign
'	apostrophe (single quote)
(	left parenthesis
)	right parenthesis
*	asterisk
+	plus sign
−	minus sign
/	slash
,	comma
.	period
:	colon
=	equals sign

การประกาศค่าของตัวแปร (Variable)

ส่วนต้นของตัวโปรแกรม เป็นการกำหนดค่าของตัวแปรที่ใช้ในคำสั่งว่าเป็นแบบใด (จำนวนจริง จำนวนเต็ม หรือ อักขระ) เช่น

Real list1

ซึ่งหมายถึงตัวแปรที่ชื่อว่า list1 มีค่าเป็นเลขจำนวนจริง

และ

Integer list2

ซึ่งหมายถึงตัวแปรที่ชื่อว่า list2 มีค่าเป็นเลขจำนวนจริง

แต่โดยทั่วไปแล้วถ้าหากกำหนดชื่อตัวแปรโดยขึ้นต้นด้วยอักษร a ถึง h หรือ o ถึง z จะหมายถึงการกำหนดตัวแปรตัวนั้นเป็นเลขจำนวนจริง ส่วนอักษร i ถึง n จะหมายถึงการกำหนดให้เป็นตัวแปรของเลขจำนวนเต็ม

การประกาศค่าของตัวแปร (Variable)

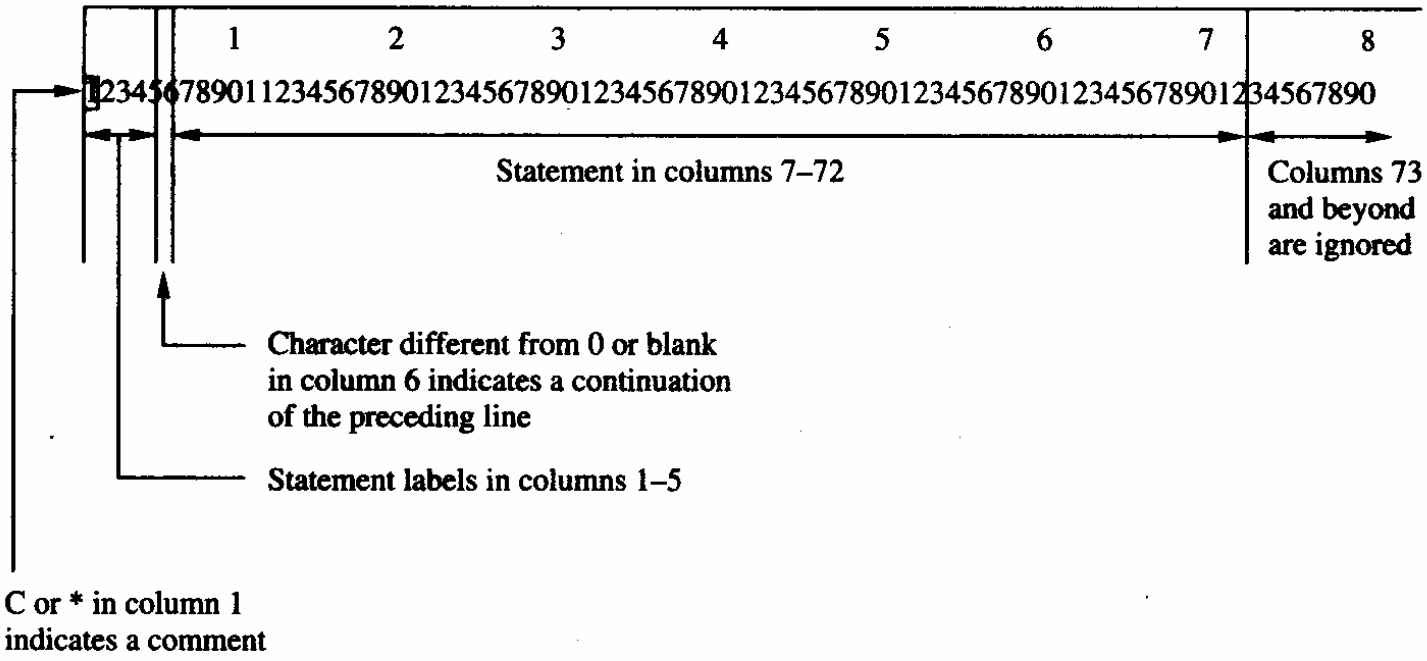
Character*n, list3

จะหมายถึงการกำหนดตัวแปรให้เป็นอักขระและขนาดความยาวของตัวแปรมีค่าไม่เกิน n ซึ่งเป็นเลขจำนวนเต็มเท่านั้น ส่วนคำว่า list3 ที่แสดงด้านบน คือ ชื่อของตัวแปรไม่ใช่ค่าของตัวแปรที่เรากำหนด ดังนั้นถ้าหากเรากำหนดเป็น

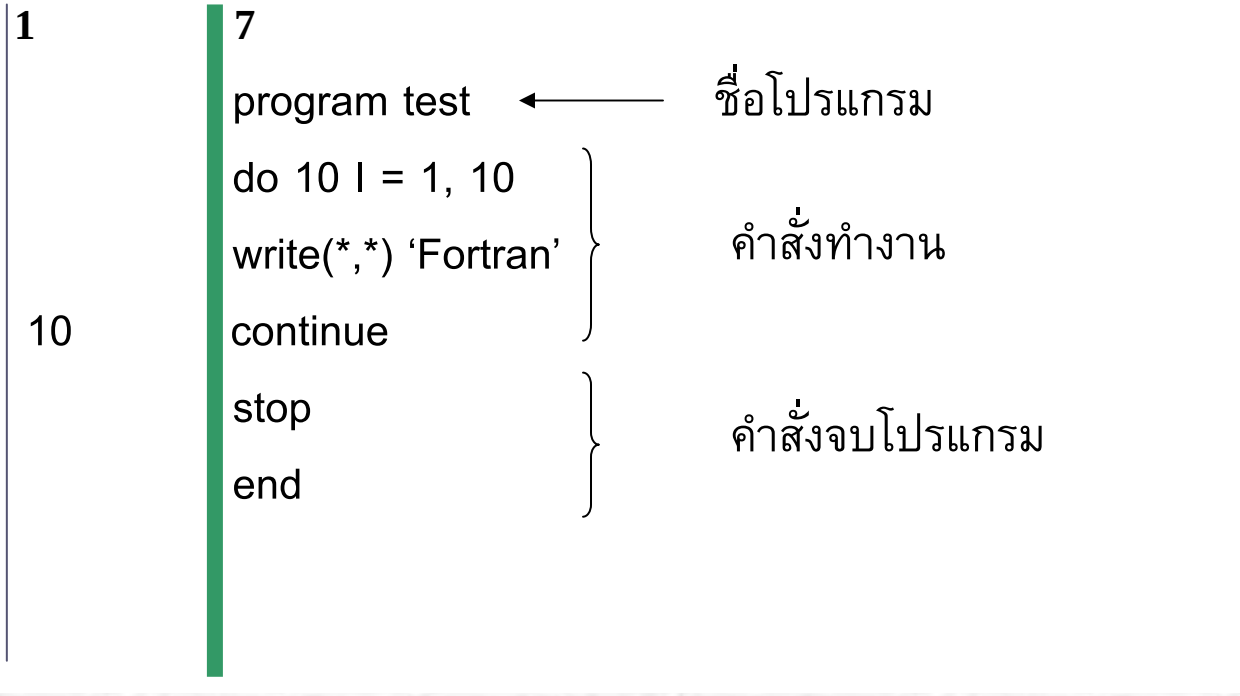
Character*10 Fname, Lname*20, Int*1

จะหมายถึง ค่าของตัวแปรเป็นแบบอักขระ ชื่อตัวแปร Fname มีค่าจำนวนอักขระไม่เกิน 10 ตัว ชื่อตัวแปร Lname มีค่าจำนวนอักขระไม่เกิน 20 ตัว ส่วนชื่อตัวแปร Int มีค่าจำนวนอักขระไม่เกิน 1 ตัว เป็นต้น

รูปแบบการเขียนภาษา Fortran



ตัวอย่าง


1
7
program test
ชื่อโปรแกรม
do 10 I = 1, 10
write(*,*) 'Fortran'
คำสั่งทำงาน
10
continue
stop
end
คำสั่งจบโปรแกรม

คำสั่งทั่วไปที่มักใช้ในการเขียนโปรแกรมฟอร์แทรน

End	หมายถึง	จบการทำงานของโปรแกรม
If(……)then	หมายถึง	ให้ทำงานถ้า (……..)
Else	หมายถึง	เป็นอย่างอื่น โดยใช้ควบคู่กับ If(……..)then
Endif	หมายถึง	สิ้นสุดเงื่อนไขโดยใช้ควบคู่กับ If(…….)then
Format	หมายถึง	การกำหนดรูปแบบการแสดงผล
Open	หมายถึง	การเปิด file ที่ต้องการใช้งาน เช่น การเปิดไฟล์ข้อมูลเพื่อนำค่ามาใช้งาน หรือ การเปิดไฟล์ใหม่เพื่อบันทึกข้อมูลลงไป
Write	หมายถึง	การให้แสดงผล อาจจะแสดงผลบนหน้าจอคอมพิวเตอร์ หรือบันทึกลงใน file ที่ต้องการเก็บ
Read	หมายถึง	ให้อ่านค่า
Stop	หมายถึง	สิ้นสุดการทำงานของโปรแกรมโดยไม่มีการคำนวณใดๆ ต่อ

ตารางคำสั่งตรรกะ

สัญลักษณ์	สัญลักษณ์ทางคณิตศาสตร์	ความหมาย
.LT.	$<$	น้อยกว่า
.GT.	$>$	มากกว่า
.EQ.	$=$	เท่ากับ
.LE.	$\leq$	น้อยกว่าหรือเท่ากับ
.GE.	$\geq$	มากกว่าหรือเท่ากับ
.NE.	$\neq$	ไม่เท่ากับ

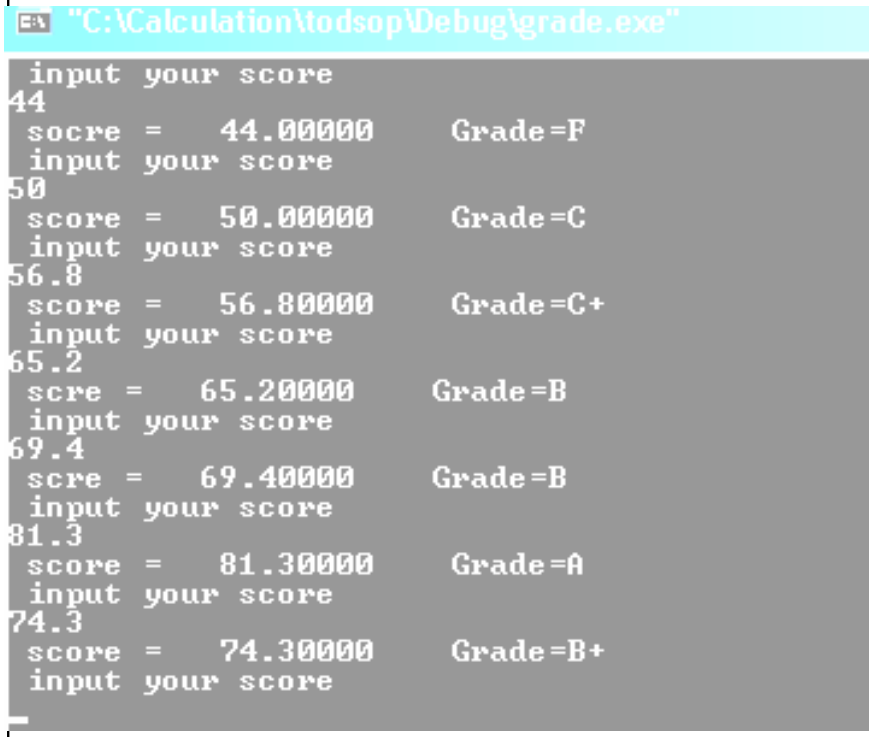
การอ่าน (Read) การรับค่า (Input) และการแสดงค่า (Output)

การอ่าน (Read) เป็นคำสั่งที่ต้องการให้โปรแกรมอ่านค่าจากชุดข้อมูลจากไฟล์อื่นหรือจากข้อมูลที่เราป้อนให้แต่ละครั้ง เพื่อนำไปใช้ในการคำนวณหรือตัดสินใจ

	A	≥ 80
75 ≤	**A -**	< 80
70 ≤	**B +**	< 75
65 ≤	**B**	< 70
60 ≤	**B -**	< 65
55 ≤	**C+**	< 60
50 ≤	**C**	< 55
45 ≤	**D**	< 50
45 >	**F**	

เกณฑ์ในการให้เกรด

```
      Program grade
      i = 0
10    i = i+1
      if(i.gt.10) goto 100
      write(*,*) 'input your score'
      read(*,*) score
      if(score.ge.80.0)then
            write(*,*) 'score =',score,'Grade=', 'A'
      else if((score.lt.80.0).and.(score.ge.75.0))then
            write(*,*) 'score =',score,'Grade=','A -'
      else if((score.lt.75.0).and.(score.ge.70.0))then
            write(*,*) 'score =',score, 'Grade=','B+'
      else if((score.lt.70.0).and.(score.ge.65))then
            write(*,*) 'scre =',score, 'Grade=', 'B'
      else if((score.lt.65.0).and.(score.ge.60))then
            write(*,*) 'score =',score, 'Grade=','B-'
      else if((score.lt.60.0).and.(score.ge.55))then
            write(*,*) 'score =',score, 'Grade=','C+'
      else if((score.lt.55.0).and.(score.ge.50))then
            write(*,*) 'score =',score, 'Grade=','C'
      else if((score.lt.50.0).and.(score.ge.45))then
            write(*,*) 'score =',score, 'Grade=','D'
      else
            write(*,*) 'socre =',score, 'Grade=','F'
      endif
      goto 10
100   stop
```



การนำชุดข้อมูลจากไฟล์อื่นเข้ามาใช้ในการคำนวณ

- การคำนวณความร้อนที่อาคารได้รับในแต่ละเวลา
- ขั้นตอนการนำค่าเข้าสู่โปรแกรม

```
open(1,file='Weather data',status='old')

      Do 10 i = 1,23
      read(1,*) clock(i),temp(i)
10    continue
      close(1)
```

นำค่าจากโปรแกรมคำนวณได้มาเก็บไว้ในไฟล์อื่น

การเปิดไฟล์ที่เป็นแหล่งของข้อมูลที่ต้องการจัดเก็บให้คำสั่ง open เช่น

open(1000,file='Heat Gen Result',status='unknown')

ซึ่ง 1000 หมายถึง การกำหนดให้ไฟล์ 'Heat Gen Result' เป็นไฟล์หมายเลข 1000

ใช้คำสั่ง write หรือ print เพื่อเก็บข้อมูลเข้ากับตัวแปรที่เรากำหนด

```
                Do 10 i = 1,23
                write(1000,1001) clock(i), heatgen(i)
1000            format(e8.2,4x,e8.3)
  10            continue
                close(1000)
```

```
      Do 10 i = 1,23
      write(1000,1001) clock(i), heatgen(i)
1000  format(e8.2,4x,e8.3)
  10  continue
      close(1000)
```

กรณีที่ไม่ต้องการบันทึกค่าใดๆในไฟล์ ‘Heat Gen Result’ แล้วให้ใช้คำสั่ง close(1000)

ต้องระวังเรื่องการกำหนด Array ของตัวแปร heatgen(i) ด้วยซึ่งในที่นี้บันทึกค่าได้ ทั้งหมดเพียง 24 ค่าเท่านั้น ถ้าเราบันทึกมากกว่า 24 แล้ว เวลา Compile โปรแกรมจะแสดง error ออกมา

0.10E+01	.264E+02
0.20E+01	.695E+03
0.30E+01	.183E+05
0.40E+01	.482E+06
0.50E+01	.127E+08
0.60E+01	.335E+09
0.70E+01	.883E+10
0.80E+01	.233E+12
0.90E+01	.613E+13
0.10E+02	.162E+15
0.11E+02	.426E+16
0.12E+02	.112E+18
0.13E+02	.296E+19
0.14E+02	.780E+20
0.15E+02	.206E+22
0.16E+02	.542E+23
0.17E+02	.143E+25
0.18E+02	.376E+26
0.19E+02	.992E+27
0.20E+02	.261E+29
0.21E+02	.689E+30
0.22E+02	.182E+32
0.23E+02	.478E+33

การกำหนด Array ของตัวแปร

กรณีที่ตัวแปรที่เราต้องการรับข้อมูลหรือจัดเก็บมีหลายค่าเราจำเป็นต้องกำหนดขนาดเอง Array ของตัวแปรเหล่านั้นที่ตอนต้นของตัวโปรแกรม

```
program heatload
real clock(24),temp(24),heatgen(24)
open(1,file='Weather data',status='old')
open(1000,file='Heat Gen Result',status='unknown')
```

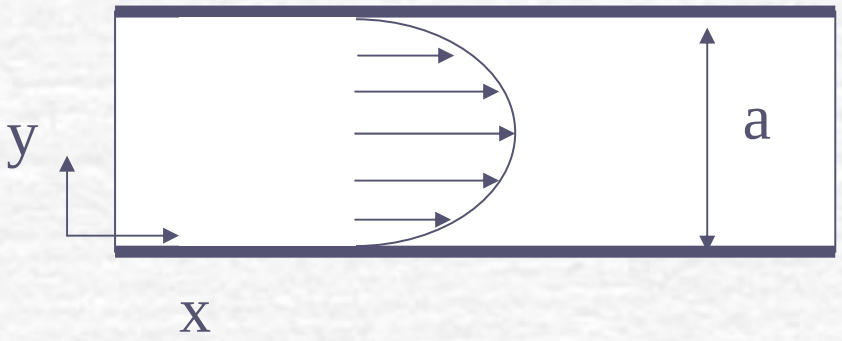
การกำหนดค่า 2 มิติ

```
real u(24,24),v(24,24)
```

ตัวอย่างการคำนวณค่าความเร็วของของไหลในท่อกลม

โปรไฟล์ความเร็ว (Velocity profile) ในแนวการไหลของของไหลที่ไหลในท่อกลมแบบราบเรียบ (Laminar pipe flow) โดยสมการของการไหล คือ

$$u(y) = \frac{a^2}{2\mu}\left(\frac{\partial p}{\partial x}\right)\left[\left(\frac{y}{a}\right)^2 - \left(\frac{y}{a}\right)\right]$$



กำหนดให้

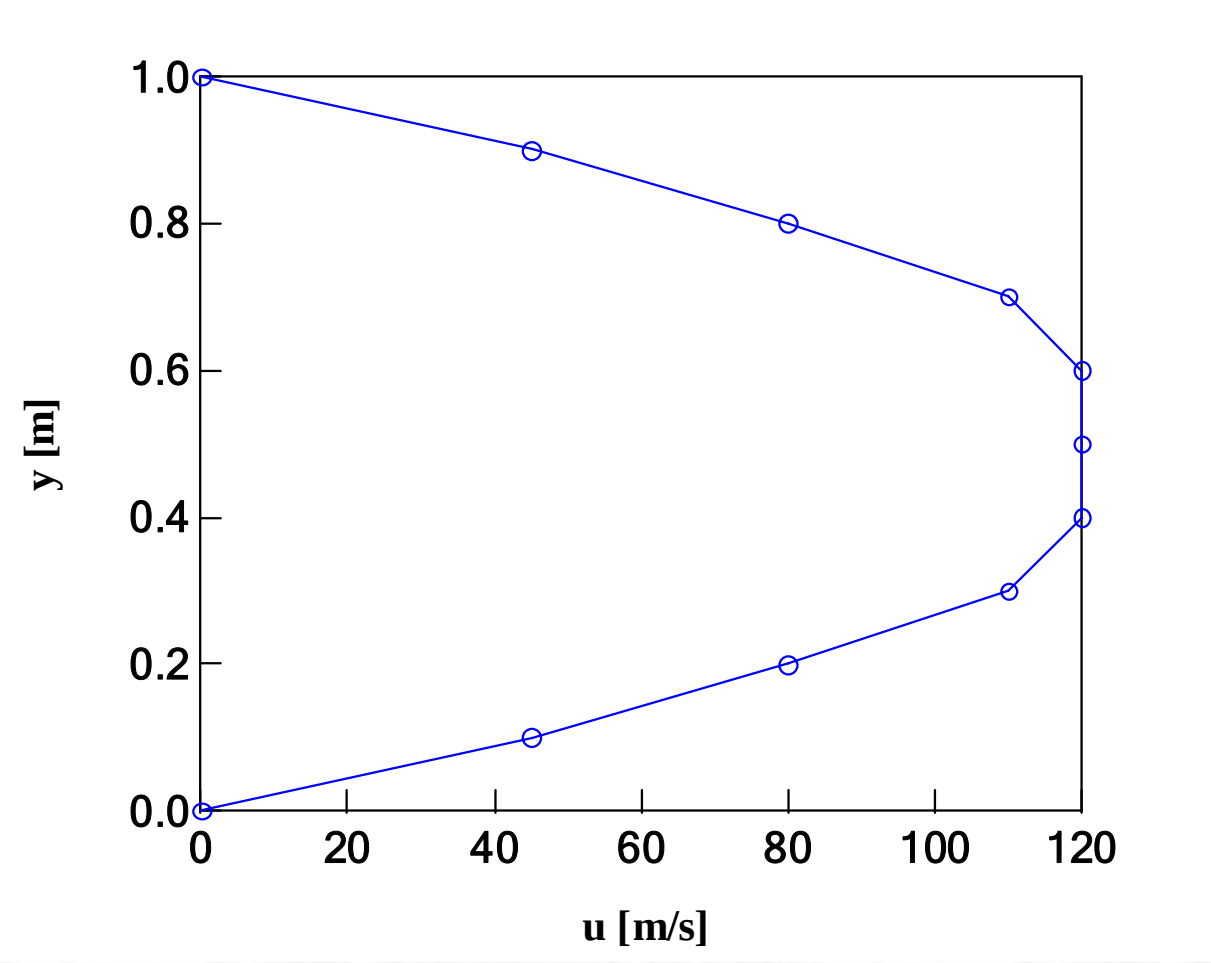
$\partial p / \partial x$ เป็นค่าคงที่เท่ากับ -1 N/m^3

a เป็นรัศมีของท่อมีค่าเท่ากับ 1 เมตร และ

μ เป็นค่าความหนืดของของไหล มีค่าเท่ากับ 10^{-3} kg/(m.s) ตามลำดับ

```
      program velocity
      real u(0:10),y(0:10)
      open(1000,file='U-velocity',status='unknown')
      ny = 10
      a = 1.0
      dy = a/real(ny)
      vis = 10.0**(-3)
      dpdx = -1.0
      do 10 i = 0,10
      y(i) = real(i)*dy
      u(i) = a*a/(2.0*vis)*dpdx*((y(i)/a)**2-(y(i)/a))
      write(1000,1001) y(i),u(i)
1001  format(e8.2,6x,e8.2)
10    continue
      close(1000)
      stop
      end
```

0.00E+00	0.00E+00
0.10E+00	0.45E+02
0.20E+00	0.80E+02
0.30E+00	0.11E+03
0.40E+00	0.12E+03
0.50E+00	0.12E+03
0.60E+00	0.12E+03
0.70E+00	0.11E+03
0.80E+00	0.80E+02
0.90E+00	0.45E+02
0.10E+01	0.00E+00



การเขียนโปรแกรมย่อย (Subroutine)

- โดยทั่วไปถ้าหากโปรแกรมมีความยาวมากๆ จะทำการแยกส่วนที่เป็นคำสั่งการคำนวณไว้ต่างหากที่โปรแกรมย่อย (Subroutine)
- การเขียนโปรแกรมย่อย ประกอบด้วยส่วนที่คล้ายๆ กับการเขียนโปรแกรมทั่วๆ ไป แตกต่างตรงที่ชื่อโปรแกรมย่อย และการคำสั่งจบโปรแกรม เช่น

การกำหนดชื่อโปรแกรมย่อย

Subroutine *name (formal-argument-list)*

โดย *name* หมายถึง ชื่อของโปรแกรมย่อย ซึ่งจะต้องเป็นชื่อที่ไม่ซ้ำกับโปรแกรมหลัก (Main program) หรือ โปรแกรมย่อยอื่นๆ และต้องไม่เป็นชื่อของคำสั่งทำงาน ส่วน *formal-argument-list* เป็นตัวแปรที่เราต้องการนำไปคำนวณ ซึ่งตัวแปรที่ส่งไปคำนวณระหว่างโปรแกรมหลักและโปรแกรมย่อยจะต้องมี Array ที่เท่ากัน

การเขียนโปรแกรมย่อย (Subroutine)

การจบโปรแกรมจะใช้คำสั่ง

End

แต่ในกรณีที่มีการส่งค่าที่คำนวณได้คืนกับไปที่โปรแกรมหลักจะใช้คำสั่ง

Return

End

การเรียกใช้โปรแกรมย่อย จะใช้คำสั่ง

Call *name (actual-argument-list)*

ตัวอย่างการใช้โปรแกรมย่อย

จงเขียนโปรแกรมเพื่อคำนวณหาค่า x จากสมการ

$$x = \frac{b \pm \sqrt{b^2 - 4ac}}{2a}$$

a	b	c	x1	x2
1.000000	9.000000	2.000000	8.772002	0.2279981
2.000000	10.00000	3.000000	4.679450	0.3205505
3.000000	11.00000	4.000000	3.257334	0.4093327
4.000000	12.00000	5.000000	2.500000	0.5000000
5.000000	13.00000	6.000000	2.000000	0.6000000

```
      Program root
      real a(10),b(10),c(10),x(10)
      open(1000,file='Root data',status='unknown')
      a(1) = 1.0
      b(1) = 9.0
      c(1) = 2.0
      call Cal(a(1),b(1),c(1))
      do 10 i = 2,5
      a(i) = a(i-1)+1.0
      b(i) = b(i-1)+1.0
      c(i) = c(i-1)+1.0
      call Cal(a(i),b(i),c(i))
10    continue
      stop
      end

      subroutine Cal(a1,b1,c1)
      real a1,b1,c1,x1,x2

      x1 = (b1+sqrt((b1*b1)-4.0*a1*c1))/(2.0*a1)
      x2 = (b1-sqrt((b1*b1)-4.0*a1*c1))/(2.0*a1)
      write(1000,*) a1,b1,c1,x1,x2
      end
```

คำสั่งการคำนวณทางคณิตศาสตร์

Function	Description	Type of Argument(s)*	Type of Value
ABS(x)	Absolute value of x	Integer or real	Same as arguments
COS(x)	Cosine of x radians	Real	Real
EXP(x)	Exponential function	Real	Real
INT(x)	Integer part of x	Real	Integer
LOG(x)	Natural logarithm of x	Real	Real
MAX($x_1, \ldots, x_n$)	Maximum of $x_1, \ldots, x_n$	Integer or real	Same as arguments
MIN($x_1, \ldots, x_n$)	Minimum of $x_1, \ldots, x_n$	Integer or real	Same as arguments
MOD(x, y)	x (mod y); $x - \text{INT}(x/y) * y$	Integer or real	Same as arguments
NINT(x)	x rounded to nearest integer	Real	Integer
REAL(x)	Conversion of x to real type	Integer	Real
SIN(x)	Sine of x radians	Real	Real
SQRT(x)	Square root of x	Real	Real

MOD(x,y) หมายถึง การเลือกใช้ค่าเศษ เช่น 11 หารด้วย 10 ค่าเศษมีค่าเท่ากับ 1 และ

MOD(223,20) = 3

PAI=2.0*ASIN(1.0)

ฟังก์ชั่น (Function)

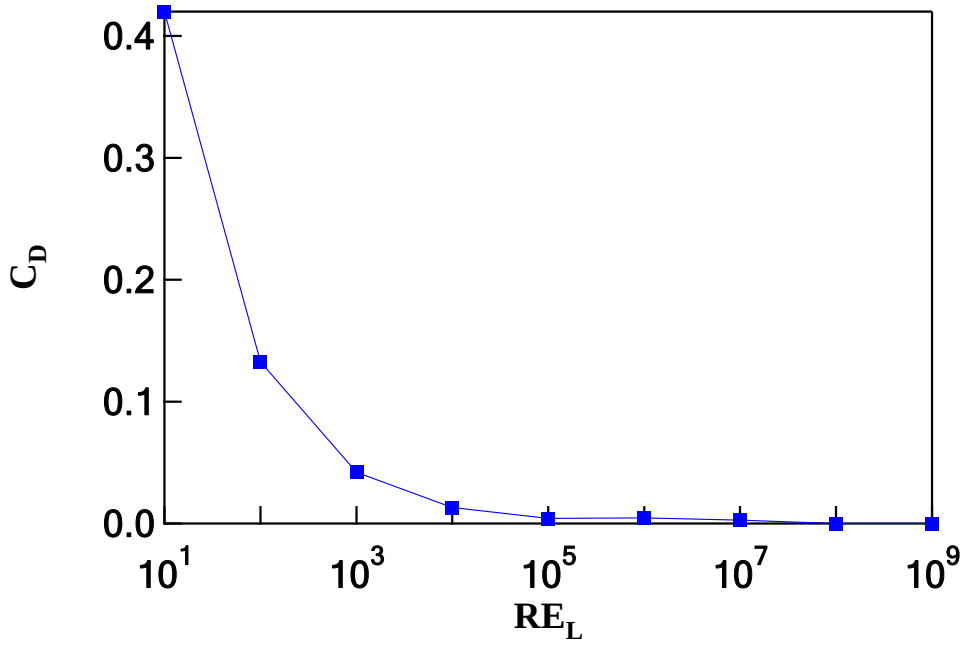
คำสั่งฟังก์ชั่นมีจุดประสงค์คล้ายกับการใช้โปรแกรมย่อย แต่ไม่นิยมใช้ในการเขียนโปรแกรมเพราะอาจจะทำให้เกิดความสับสนระหว่างตัวแปรที่เป็น Array และคำสั่งฟังก์ชั่น

ตัวอย่างเช่น เราต้องการคำนวณสัมประสิทธิ์แรงต้านของอากาศ (Drag coefficient) ที่ไหลผ่านแผ่นราบ (Flat plate) ด้วย Reynolds number (Re) ที่แตกต่างกัน สมมติให้

$$C_D = \begin{cases} \dfrac{1.328}{\sqrt{\mathrm{Re}_L}} & when \quad \mathrm{Re}_L < 5\times10^5 \\ \dfrac{0.074}{\mathrm{Re}_L^{1/5}} & when \quad 5\times10^5 \le \mathrm{Re}_L < 10^7 \\ \dfrac{0.455}{\left(\log \mathrm{Re}_L\right)^{2.58}} & when \quad 10^7 \le \mathrm{Re}_L \le 10^9 \end{cases}$$

```
      program drag
      REAL CD(10),RE(10)
      OPEN(1000,FILE='DRAG')
      RE(1) = 10.0
      CD(1) = F(RE(1))
      WRITE(1000,*) RE(1),CD(1)
      DO 10 I = 2,9
      RE(I) = RE(I-1)*10.0
      CD(I) = F(RE(I))
      WRITE(1000,*) RE(I),CD(I)
10    CONTINUE
      STOP
      END

      FUNCTION F(W)
      IF(W.GE.10.E+7.AND.W.LE.10.E+9) F = 0.455/(LOG(W)**2.58)
      IF(W.GE.10.E+5.AND.W.LT.10.E+7) F = 0.074/(W**0.2)
      IF(W.LT.10.E+5) F = 1.328/SQRT(W)
      RETURN
      END
```



การกำหนดตัวแปรร่วม (Common)

เพื่อกำหนดค่าตัวแปรระหว่างโปรแกรมหลัก (Main program) กับ โปรแกรมย่อย (Subroutine) และ ฟังก์ชั่น (Function) เป็นค่าเดียวกัน ข้อควรระวังในการใช้คำสั่งคอมมอน คือ ค่าตัวแปรที่เราส่งไปที่โปรแกรมย่อยหรือฟังก์ชั่นอาจจะถูกเปลี่ยนแปลงได้ถ้าเรากำหนดค่าตัวแปรนั้นๆ ใหม่ในโปรแกรมย่อยหรือฟังก์ชั่น

ตัวอย่าง

เมื่อค่า dens, vis, และ dia ถูกประกาศ (Declare) ในโปรแกรมหลัก โดยใช้คำสั่งคอมมอนทั้งในโปรแกรมหลักและฟังก์ชั่น ค่าเหล่านี้จะถูกส่งไปยังฟังก์ชั่นด้วย

```
      program common1
      common dens,vis,dia
      real RE(10)
      dens = 10.e+3
      vis = 10.0e-3
      dia = 1.0
      do 10 i = 1,5
      U  = 2.0*10.0**(i)
      RE(i) = F1(U)
      write(*,*) U,RE(i)
10    continue
      stop
      end

      function F1(U)
      common dens,vis,dia
      F1 = U*dens*dia/vis
      return
      end
```

Assignment 1

จากรูปจงเขียนโปรแกรมเพื่อคำนวณลักษณะการเคลื่อนที่ ของหญ้าที่ถูกตัดจากเครื่องตัดหญ้า โดยกำหนดให้ค่าการเปลี่ยนแปลงของเวลามีค่า = 0.05, 0.1 และ 0.15 วินาที และเปรียบเทียบผลกับคำตอบแม่นตรง นอกจากนี้ให้แสดงวิธีการคำนวณเพื่อหาค่าคำตอบแม่นตรง (Exact solution)