ใบความรู้ที่ 5
ตัวดำเนินการ (Operator)

สาระสำคัญ/ความคิดรวบยอด

การเขียนโปรแกรมคอมพิวเตอร์นั้นจะต้องมีการประมวลผลกับข้อมูล โดยข้อมูลจะถูกเก็บอยู่ในหน่วยความจำของคอมพิวเตอร์ในรูปแบบของตัวแปร การประกาศตัวแปรต่างๆ จะใช้หน่วยความจำไม่เท่ากัน และมีช่วงของการเก็บข้อมูลไม่เท่ากัน ผู้เขียนโปรแกรมจะต้องทราบว่าข้อมูลที่ต้องการประมวลผลนั้นเป็นข้อมูลประเภทใด และในการประมวลผลจะต้องมีการกระทำกับตัวแปรต่างๆ ตัวที่นำมากระทำเรียกว่าตัวดำเนินการ ซึ่งมีทั้งการดำเนินการทางคณิตศาสตร์และทางลอจิก ดังนั้นผู้เขียนโปรแกรมจะต้องทำความเข้าใจกับประเภทของข้อมูลและการใช้ตัวดำเนินการ จึงจะสามารถเขียนโปรแกรมให้ทำงานตามที่ต้องการได้

5.1 ตัวดำเนินการทางคณิตศาสตร์ (Arithmetic Operators)

ในภาษาซี มีตัวดำเนินการทางคณิตศาสตร์อยู่ 5 ชนิดด้วยกันดังตารางที่ 5.1 ดังนี้

ตารางที่ 5.1 ตัวดำเนินการทางคณิตศาสตร์

เครื่องหมาย	ความหมาย
+	การบวก (addition)
-	การลบ (subtraction)
*	การคูณ (multiplication)
/	การหาร (division)
%	การหารเอาเศษ (Modulation)

ตัวดำเนินการจะประกอบด้วยสัญลักษณ์ทั้ง 5 ที่แสดงไว้ข้างต้น ในขณะที่ตัวถูกดำเนินการ (Operands) นั้นจะต้องถูกแทนด้วยค่าตัวเลข ซึ่งอาจเป็นตัวแปรหรือค่าคงที่ ทั้งนี้ตัวถูกดำเนินการยังสามารถเป็นไปได้ทั้งเลขจำนวนเต็ม เลขจำนวนจริง และตัวอักขระ เมื่อนำทั้งตัวถูกตัวดำเนินการ และตัวดำเนินการประกอบเข้าด้วยกัน ก็จะเรียกว่านิพจน์นั่นเอง เช่น a/b ถือเป็นนิพจน์ มีการใช้ตัวดำเนินการหาร (/) โดยที่ a เป็นตัวถูกดำเนินการตัวแรกที่ใช้เป็นตัวตั้ง ในขณะที่ b เป็นตัวถูกดำเนินการตัวที่สองที่ใช้เป็นตัวหาร

อย่างไรก็ตาม สำหรับตัวถูกดำเนินการสองตัว ที่นำมาคำนวณผ่านตัวดำเนินการมอดูรัส (%) นั้นจะต้องเป็นจำนวนเต็มทั้งคู่ และตัวถูกดำเนินการที่เป็นตัวหารจะต้องไม่เป็นค่าศูนย์ ในทำนองเดียวกัน ตัวถูกดำเนินการที่นำมาคำนวณผ่านตัวดำเนินการหาร (/) นั้น ตัวถูกดำเนินการที่เป็นตัวหาร ก็จะต้องไม่เป็นค่าศูนย์เช่นกัน ซึ่งจากรายละเอียดของตัวอย่างต่อไปนี้ จะแสดงถึงนิพจน์ที่กำหนดขึ้น และผลลัพธ์ที่ได้

ตัวอย่างที่ 5.1 สมมติว่า a และ b ถูกกำหนดให้เป็นตัวแปรชนิดเลขจำนวนเต็ม ซึ่งถูกกำหนดค่าเป็น 10 และ 3 ตามลำดับ เมื่อนำตัวแปรทั้งสองมาผ่านนิพจน์คณิตศาสตร์ตามตารางต่อไปนี้ ก็จะได้

ตารางที่ 5.2 ตัวอย่างการดำเนินการทางคณิตศาสตร์กับจำนวนเต็ม

นิพจน์	ผลลัพธ์
a+b	13
a-b+1	8
a*b	30
a/b	3
a%b	1

ตัวอย่างที่ 5.2 สมมติว่า n1 และ n2 ถูกกำหนดให้เป็นตัวแปรชนิดเลขจำนวนจริง ซึ่งถูกกำหนดค่าเป็น 12.5 และ 2.0 ตามลำดับ เมื่อนำตัวแปรทั้งสองมาผ่านนิพจน์คณิตศาสตร์ตามตารางต่อไปนี้ ก็จะได้

ตารางที่ 5.3 ตัวอย่างการดำเนินการทางคณิตศาสตร์กับจำนวนจริงทศนิยม

นิพจน์	ผลลัพธ์
n1+ n2	14.5
n1- n2+1	10.5
n1*n2	25.0
n1/n2	6.25

ตัวอย่างที่ 5.3 สมมติว่า c1 และ c2 ถูกกำหนดให้เป็นตัวแปรชนิดตัวอักษร ซึ่งถูกกำหนดค่าเป็น P และ T ตามลำดับ เมื่อนำตัวแปรทั้งสองมาผ่านนิพจน์คณิตศาสตร์ตามตารางต่อไปนี้ ก็จะได้

ตารางที่ 5.4 ตัวอย่างการดำเนินการทางคณิตศาสตร์กับตัวอักษร

นิพจน์	ผลลัพธ์
c1	80
c2	84
c1+c2	164
c1+c2+5	169
c1+c2+ ‘5’	217

5.2 ตัวดำเนินการยูนารี (Unary Operator)

ภาษาซี ยังผนวกตัวดำเนินการที่นำมาใช้กับตัวถูกดำเนินการใดๆ ในการสร้างค่าใหม่ขึ้นมา นั่นก็คือ ตัวดำเนินการยูนารี และหนึ่งในนั้นคือ ตัวดำเนินการยูนารีติดลบ (Unary Minus) ตัวอย่างตัวถูกดำเนินการชนิดโดดๆ เช่น 882 เมื่อมีการนำตัวดำเนินการยูนารี – นำหน้าค่าก็จะถูกเปลี่ยนเป็นค่าติดลบในทันทีคือ -882

นั่นเอง อย่างไรก็ตาม เครื่องหมายลบดังกล่าวมีความแตกต่างกันอย่างสิ้นเชิง กับตัวดำเนินการทางคณิตศาสตร์ที่ใช้สำหรับการลบ (Subtraction)

ตัวอย่างที่ 5.4 ตัวอย่างการใช้ตัวดำเนินการยูนารีค่าติดลบ

-0743 -0X7FFF -0.2

-root1 -(x+y) -3*(x+y)

จากตัวอย่างข้างต้นจะพบว่า ตัวดำเนินการยูนารีสามารถนำไปใช้กับตัวถูกดำเนินการที่เป็นค่าคงทีแบบเลขจำนวนเต็ม เลขจำนวนจริง ตัวแปร และนิพจน์ก็ได้

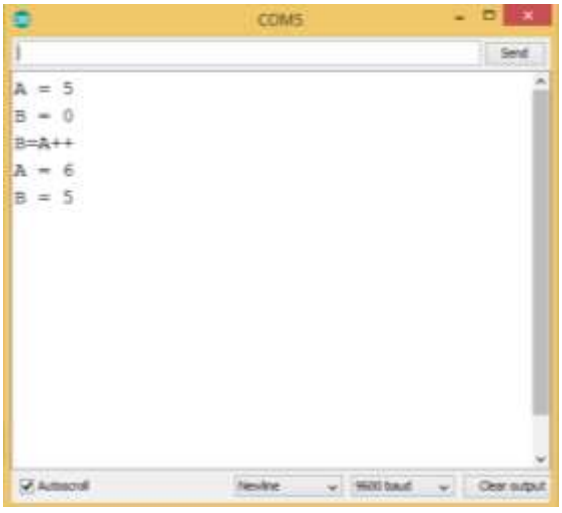
นอกจากนี้ตัวดำเนินการยูนารี ก็ยังประกอบด้วย ตัวดำเนินการเพิ่มค่า (Increment Operator) ซึ่งใช้เครื่องหมาย ++ และตัวดำเนินการลดค่า (Decrement Operator) ซึ่งใช้เครื่องหมาย – โดยการเพิ่มค่าหรือลดค่าด้วยตัวดำเนินการดังกล่าว จะเพิ่มขึ้นที่ละหนึ่งหรือลดทีละหนึ่ง และตัวถูกดำเนินการที่นำตัวดำเนินการนี้มาใช้ จะต้องเป็นตัวแปรโดดๆ โดยสามารถใช้เครื่องหมาย ++ และ – แต่ทั้งสองแบบนี้จะมีวิธีจัดการกับค่าที่แตกต่างกัน ตามรายละเอียดต่อไปนี้

ตารางที่ 5.5 ตัวดำเนินการยูนารี

ตัวดำเนินการ	นิพจน์	ความหมาย
++ (prefix)	++a	เพิ่มค่าขึ้นอีกหนึ่งให้กับ a ก่อน แล้วจึงนำค่าใหม่ของ a ในนิพจน์นี้ไปใช้
++ (postfix)	a++	นำค่าปัจจุบันของ a ในนิพจน์นี้ไปใช้งานก่อนแล้วจึงเพิ่มค่า a ขึ้นอีกหนึ่ง
-- (prefix)	--b	ลดค่าลงอีกหนึ่งให้กับ b ก่อน แล้วจึงนำค่าใหม่ของ b ในนิพจน์นี้ไปใช้
-- (postfix)	b--	นำค่าปัจจุบันของ b ในนิพจน์นี้ไปใช้งานก่อนแล้วจึงลดค่า b ลงอีกหนึ่ง

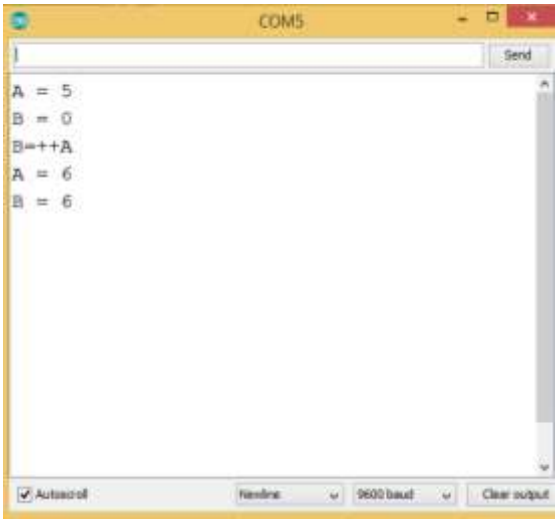
ตัวอย่างที่ 5.5 โปรแกรมและผลลัพธ์ที่ได้จากการนำตัวดำเนินการ ++ (prefix) ไปใช้งาน

```
1   void setup() {
2     int A=5,B=0;
3     Serial.begin(9600);
4     Serial.print("A = ");  Serial.println(A);
5     Serial.print("B = ");  Serial.println(B);
6     B=++A;
7     Serial.println("B=++A");
8     Serial.print("A = ");  Serial.println(A);
9     Serial.print("B = ");  Serial.println(B);
10  }
11  void loop() {
12
13  }
```



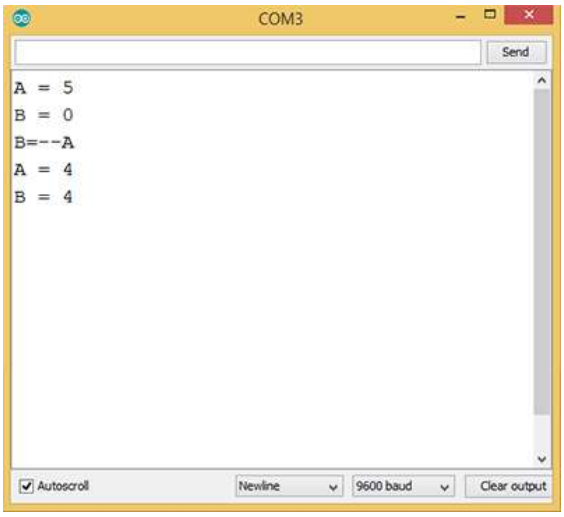
ตัวอย่างที่ 5.6 โปรแกรมและผลลัพธ์ที่ได้จากการนำตัวดำเนินการ ++ (postfix) ไปใช้งาน

```
1   void setup() {
2     int A=5,B=0;
3     Serial.begin(9600);
4     Serial.print("A = ");  Serial.println(A);
5     Serial.print("B = ");  Serial.println(B);
6     B=++A;
7     Serial.println("B=++A");
8     Serial.print("A = ");  Serial.println(A);
9     Serial.print("B = ");  Serial.println(B);
10  }
11  void loop() {
12
13  }
```



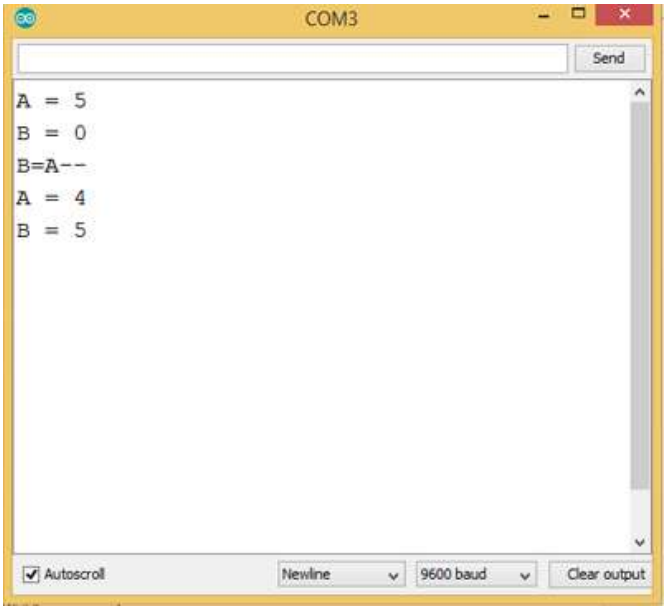
ตัวอย่างที่ 5.6 โปรแกรมและผลลัพธ์ที่ได้จากการนำตัวดำเนินการ -- (prefix) ไปใช้งาน

```
1   void setup() {
2     int A=5,B=0;
3     Serial.begin(9600);
4     Serial.print("A = ");  Serial.println(A);
5     Serial.print("B = ");  Serial.println(B);
6     B=--A;
7     Serial.println("B=--A");
8     Serial.print("A = ");  Serial.println(A);
9     Serial.print("B = ");  Serial.println(B);
10  }
11  void loop() {
12
13  }
```



ตัวอย่างที่ 5.7 โปรแกรมและผลลัพธ์ที่ได้จากการนำตัวดำเนินการ -- (postfix) ไปใช้งาน

```
1   void setup() {
2     int A=5,B=0;
3     Serial.begin(9600);
4     Serial.print("A = ");  Serial.println(A);
5     Serial.print("B = ");  Serial.println(B);
6     B=--A;
7     Serial.println("B=--A");
8     Serial.print("A = ");  Serial.println(A);
9     Serial.print("B = ");  Serial.println(B);
10  }
11  void loop() {
12
13  }
```



5.3 ตัวดำเนินการเปรียบเทียบและตรรกะ (Comparison and Logical Operators)

ใช้ในการดำเนินการเปรียบเทียบค่า 2 ค่าและบอกผลลัพธ์ของการเปรียบเทียบว่าเป็นจริง (True) หรือเป็นเท็จ (False) ดังตารางที่ 5.6 สำหรับเครื่องหมายดำเนินการทางตรรกะ (Logical Operators) ใช้ในการตัดสินใจมีรูปแบบเดียวกับเงื่อนไขคำสั่ง ถ้า (if) แล้วจะดำเนินการอย่างไรต่อไป ผลลัพธ์ที่ได้จะอยู่ในรูปแบบเดียวกับตัวดำเนินการสัมพันธ์ดังตารางที่ 5.7 และตารางที่ 5.8 แสดงผลการเปรียบเทียบด้วยตัวดำเนินการทางตรรกะ

ตารางที่ 5.6 ตัวดำเนินการเปรียบเทียบ

เครื่องหมาย	ความหมาย	ตัวอย่าง		ผลลัพธ์ที่ได้
>	มากกว่า	a > b	a มากกว่า b	เป็นเท็จ (false หรือ 0)
>==	มากกว่าหรือเท่ากับ	a >= b	a มากกว่าหรือเท่ากับ b	เป็นเท็จ (false หรือ 0)
<	น้อยกว่า	a < b	a น้อยกว่า b	เป็นจริง (true หรือ 1)
<==	น้อยกว่าหรือเท่ากับ	a <= b	a น้อยกว่าหรือเท่ากับ b	เป็นจริง (true หรือ 1)
==	เท่ากับ	a == b	a เท่ากับ b	เป็นเท็จ (false หรือ 0)
!=	ไม่เท่ากับ	a != b	a ไม่เท่ากับ b	เป็นจริง (true หรือ 1)

หมายเหตุ เมื่อกำหนดให้ a=10 และ b=3

ตารางที่ 5.7 ตัวดำเนินการทางตรรกะ

เครื่องหมาย	ความหมาย	ตัวอย่าง	ผลลัพธ์ที่ได้
&&	และ (AND)	(a>b) && (b<a)	เป็นจริง (true หรือ 1)
||	หรือ (OR)	(a>b) || (b>a)	เป็นจริง (true หรือ 1)
!	ไม่ (NOT)	!(12>a)	เป็นจริง (true หรือ 1)

หมายเหตุ เมื่อกำหนดให้ a=10 และ b=3

ตารางที่ 5.8 ผลการเปรียบเทียบด้วยตัวดำเนินการทางตรรกะ

A	B	A&&B	A||B	!A	!B
0	0	0	0	1	1
0	1	0	1	1	0
1	0	0	1	0	1
1	1	1	1	0	0

หมายเหตุ เมื่อกำหนดให้ 0=False และ 1=True

ข้อแตกต่างระหว่างตัวดำเนินการ “=” และ “==”

ตัวดำเนินการ “=” เป็นตัวดำเนินการกำหนดค่าให้กับตัวแปร เช่น a=b หมายความว่านำค่า b ไปใส่หรือกำหนดค่าให้กับตัวแปร a เพราะฉะนั้น a จะมีค่าที่เก็บอยู่เท่ากับ b

ตัวดำเนินการ “==” เป็นตัวดำเนินสำหรับการเปรียบเทียบ เช่น a==b หมายความว่าเป็นการเปรียบเทียบว่า a มีค่าเท่ากับ b หรือไม่ ถ้าเท่ากันจะให้ผลเป็นจริง หากไม่เท่ากันจะให้ผลเป็นเท็จ

5.4 ลำดับความสำคัญของเครื่องหมาย

ส่วนใหญ่นิพจน์ที่เขียนขึ้นในโปรแกรมมักจะซับซ้อน มีการดำเนินการหลายอย่างปะปนอยู่ภายในนิพจน์เดียวกัน

ลำดับความสำคัญ	ลำดับความสำคัญจากสูงไปต่ำ
1	( )
2	!, ++, - -
3	*, /, %
4	+, -
5	<, <=, >, >=
6	==, !=
7	&&
8	||
9	*=, /=, %=, +=, -=

ตัวอย่างเช่น

a-b*c

1. ดำเนินการกับเครื่องหมาย “*” และตามด้วยเครื่องหมาย “-”
2. ผลลัพธ์ที่เกิด a-(b*c)

a+ ++b

1. ดำเนินการกับเครื่องหมาย “++” และตามด้วยเครื่องหมาย “+”
2. ผลลัพธ์ที่เกิด a+(++b)

a/b%c

1. ดำเนินการกับเครื่องหมาย “/” และตามด้วยเครื่องหมาย “%”
2. ผลลัพธ์ที่เกิด (a/b)%c

ตัวอย่างที่ 5.8 การเขียนโปรแกรมแสดงข้อมูลผลการคำนวณหาพื้นที่สี่เหลี่ยมเมื่อกำหนดค่าความกว้างเท่ากับ 8 และความยาวเท่ากับ 5 จากนั้นแสดงผลลัพธ์ออกทางหน้าต่าง Serial Monitor ซึ่งขั้นตอนการแก้ปัญหา มีดังนี้ คือ

ขั้นตอนที่ 1 การวิเคราะห์และกำหนดรายละเอียดของปัญหา

1) ปัญหาที่ต้องการแก้ คือ *การคำนวณหาพื้นที่สี่เหลี่ยม*
2) ข้อมูลเข้า คือ *-ไม่มี-*
3) ข้อมูลออก คือ *ผลลัพธ์ของการคำนวณพื้นที่สี่เหลี่ยม*

ขั้นตอนที่ 2 การออกแบบโปรแกรม (รหัสลำลองหรือผังงาน)

เขียนลำดับการทำงานโดยใช้รหัสลำลอง หรือผังงานได้ดังนี้

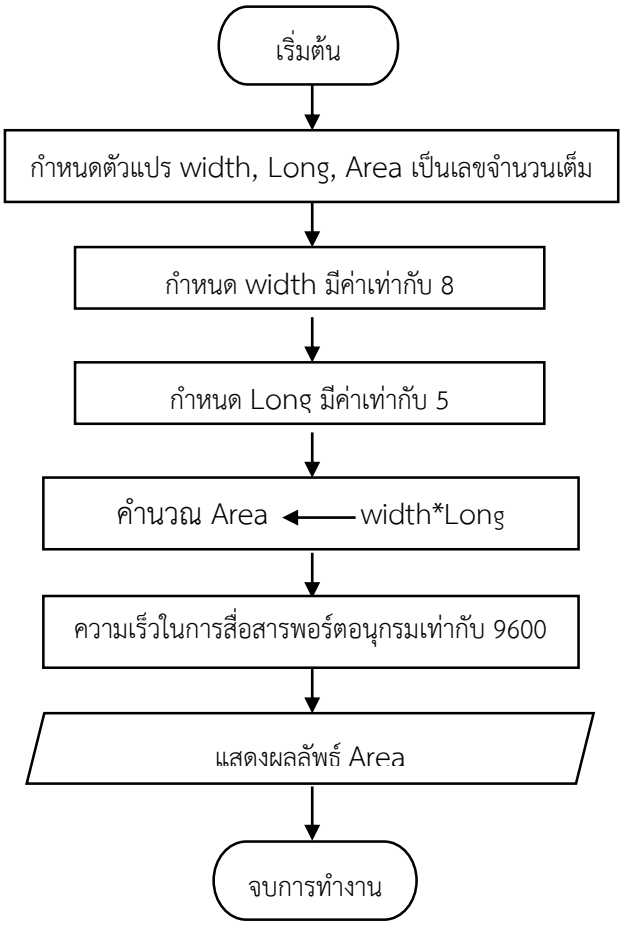
2.1) การเขียนรหัสลำลอง

เริ่มต้น

1. ประกาศตัวแปร width, Long, Area เป็นเลขจำนวนเต็ม เพื่อเก็บข้อมูลความกว้าง ความยาว และผลลัพธ์ของพื้นที่สี่เหลี่ยมตามลำดับ
2. กำหนดตัวแปร width มีค่าเท่ากับ 8
3. กำหนดตัวแปร Long มีค่าเท่ากับ 5
4. คำนวณ Area ⟵ width*Long
5. กำหนดความเร็วในการสื่อสารพอร์ตอนุกรมเท่ากับ 9600
6. แสดงข้อมูล Area

จบการทำงาน

2.2) การเขียนผังงาน



ขั้นตอนที่ 3 เขียนโค้ดคำสั่งโปรแกรมที่ใช้ในการทดลอง

คำสั่ง		คำอธิบายโปรแกรม
1	void setup() {	
2	int width,Long,Area;	//ประกาศตัวแปร width, Long และ Area เป็นเลขจำนวนเต็ม
3	width=8;	//กำหนด width มีค่าเท่ากับ 8
4	Long=5;	//กำหนด Long มีค่าเท่ากับ 5
5	Area=width*Long;	// คำนวณ Area=width*Long
6	Serial.begin(9600);	// กำหนดอัตราเร็วของการสื่อสารเป็น 9600 บิตต่อ 1 วินาที
7	Serial.print("Area = ");	//แสดงข้อมูลข้อความ “Area = ”
8	Serial.println(Area);	//แสดงข้อมูลตัวแปร Area
9	}	
10	void loop() {	
11		
12	}	

ขั้นตอนที่ 4 การทดสอบโปรแกรม

4.1) ทำการอัพโหลดโปรแกรม

4.2) เปิดหน้าต่าง Serial Monitor

4.3) สังเกตผลการทำงาน ดังภาพ