ใบความรู้ที่ 4
ชนิดข้อมูลและการใช้งานตัวแปร

4.1 ชนิดข้อมูล (Data type)

ชนิดข้อมูลหรือประเภทของข้อมูล เป็นตัวบอกรายละเอียดของตัวแปรที่ประกาศใช้งาน ว่าตัวแปรดังกล่าวสามารถใช้เก็บข้อมูลอะไรได้ ในภาษาซี จะมีชนิดข้อมูลที่ประกอบไปด้วยตัวอักขระ (Character) จำนวนตัวเลขแบบจำนวนเต็ม (Integer) และจำนวนตัวเลขทศนิยม (Floating point number) ชนิดข้อมูลทั้ง 3 แบบสามารถแบ่งออกเป็นชนิดข้อมูลต่างๆ ได้ตามตารางที่ 4.1 (เฉพาะชนิดข้อมูลที่นิยมใช้ในภาษาซี) ดังนี้

ตารางที่ 4.1 ชนิดข้อมูลในภาษาซี

ชนิดข้อมูล	ความหมาย	ขนาดข้อมูล (Byte)	ค่าที่เป็นไปได้
char	ตัวอักขระ	1	-128 ถึง 127
int	เลขจำนวนเต็ม	2	-32768 ถึง 32767
short int	เลขจำนวนจริงแบบสั้น	2	-32768 ถึง 32767
Long int	เลขจำนวนจริงแบบสั้น	4	-2,147,483,648 ถึง 2,147,483,647
unsigned char	ตัวอักขระ ไม่รวมเครื่องหมาย	1	0 ถึง 255
unsigned short int	เลขจำนวนเต็มแบบสั้น ไม่รวมเครื่องหมาย	2	0 ถึง 65535
unsigned long int	เลขจำนวนเต็มแบบยาว ไม่รวมเครื่องหมาย	2	0 ถึง 65535
signed short int	เลขจำนวนเต็มแบบสั้น รวมเครื่องหมาย	2	-32768 ถึง 32767
float	เลขจำนวนจริง มีทศนิยม	4	1.2E-38 ถึง 3.4E+38
doubled	เลขจำนวนจริง 2 เท่า	8	2.3E-308 ถึง 1.7E+308
long doubled	เลขจำนวนจริง 2 เท่าแบบยาว	10	3.4E-4932 ถึง 1.1E+4932

หมายเหตุ 1 ไบต์เท่ากับ 8 บิต และขอบเขตของชนิดข้อมูลแต่ละตัวยังขึ้นอยู่กับคอมไพเลอร์ที่ใช้งานกับระบบปฏิบัติการด้วย

4.2 ตัวแปร (Variable)

คือการรวมกันของชนิดข้อมูล (Data type) และชื่อ (Identifier) ที่ประกาศใช้ การประกาศใช้งานตัวแปรในภาษาซีจะประกาศได้ 2 รูปแบบคือ ประกาศภายในฟังก์ชั่น ตัวแปรประเภทนี้จะเป็นตัวแปรโลคอล (Local) ใช้งานได้ภายในฟังก์ชั่นเท่านั้น และการประกาศภายนอกฟังก์ชั่น ตัวแปรประเภทนี้ จะเป็นตัวแปรโกลบอล (Global) คือทุกๆฟังก์ชั่นภายในโปรแกรมสามารถใช้งานได้ การประกาศในฟังก์ชั่น จะต้องมาก่อนชุดคำสั่ง ส่วนการประกาศนอกฟังก์ชั่น จะต้องมาก่อนทุกๆฟังก์ชั่นในโปรแกรม รูปแบบการประกาศใช้งานตัวแปรมีดังนี้

```
type variable_name;
```

โดย type คือ ชนิดของตัวแปร (ตารางที่ 4.1)
variable_name คือ ชื่อตัวแปรที่กำหนด

ตัวอย่าง เช่น

```
char name;
int Score;
float=Area;
```

โดยถ้ามีตัวแปรชนิดเดียวอาจประกาศพร้อมกันโดยใช้ คอมมา คั่นระหว่างชื่อของตัวแปร ถ้ามีการกำหนดค่าให้ใช้เครื่องหมาย = และใช้เครื่องหมายแสดงการจบคำสั่งเมื่อสิ้นสุด คำสั่ง ตัวอย่าง เช่น

ตัวอย่าง เช่น

```
char name, Lastname;
char day = 'S'
char surname[20] ="Kodedee";
int  x,y,z;
int A=5, B=10;
float  a,b,c;
float  k=1.234567, m;
```

4.3 ค่าคงที่ (Constants)

ค่าคงที่มีลักษณะทำนองเดียวกับตัวแปร แต่เป็นตัวแปรที่มีค่าไม่เปลี่ยนแปลง การประกาศทำได้ 2 แบบ คือใช่คำสั่ง const ดังนี้

```
const  type  variable_name  = value ;
```

โดย type คือ ชนิดของตัวแปร (ตารางที่ 4.1)
variable_name คือ ชื่อตัวแปรที่กำหนด
value คือ ค่าของตัวแปรที่กำหนด

ตัวอย่าง เช่น

```
const  int  num1 = 5;
const  float vat =  0.07;
const  char student[10] = ‘Bush’;
```

ใช้ directive #define ดังนี้

```
#define variable_name   value
```

โดย variable_name คือ ชื่อค่าคงที่
value คือ ค่าคงที่

ตัวอย่าง เช่น

```
#define  NUM1  5
#define VAT 0.07
#define STUDENT Bush
```

4.4 อาเรย์ (Array)

อาเรย์ คือ การเก็บชุดข้อมูลที่เป็นชนิดเดียวกันไว้ในตัวแปรเดียวกัน เช่น หากเราต้องการเก็บค่าอายุของคน 10 คน แทนที่เราจะใช้การสร้างตัวแปร age1 age2 age... ก็เปลี่ยนมาใช้ตัวแปรอาร์เรย์แทน ตัวแปรแบบอาร์เรย์มีคุณสมบัติที่สำคัญอยู่ 3 ประการคือ

- ข้อมูลอยู่ภายใต้ชื่อตัวแปรเดียวกัน
- มีชนิดข้อมูลเหมือนกัน
- การเข้าถึงข้อมูลในอาร์เรย์จะใช้ดัชนี (index) ในการอ้างอิง

นอกจากนี้ในภาษา C/C++ จะใช้งานอาร์เรย์จำเป็นต้องมีการจองพื้นที่หน่วยความจำก่อนใช้งาน มีรูปแบบการประกาศใช้งาน ดังนี้

```
type  variable_name[ขนาดของข้อมูล];
```

โดย type คือ ชนิดของตัวแปร (ตารางที่ 4.1)
variable_name คือ ชื่อตัวแปรที่กำหนด

หรือ

```
type  variable_name[]={ชุดข้อมูล};
```

โดย type คือ ชนิดของตัวแปร (ตารางที่ 4.1)
variable_name คือ ชื่อตัวแปรที่กำหนด

ให้สังเกตว่า การสร้างตัวแปรแบบอาร์เรย์จะแตกต่างจากการสร้างตัวแปรแบบปกติตรงที่การสร้างจะต้องมีเครื่องหมาย [และ] ต่อท้ายชื่อตัวแปรด้วย และการกำหนดข้อมูล จะใช้เครื่องหมาย { และ } ในการครอบชุดข้อมูล โดยข้อมูลแต่ละชุดจะถูกคั่นด้วยเครื่องหมาย , (comma)

หรือถ้าต้องการสร้างตัวแปรอาร์เรย์ที่ต้องการกำหนดค่าทีหลัง จะต้องมีการกำหนดขนาดของอาร์เรย์ด้วย โดยนำค่าของขนาด หรือจำนวนข้อมูลที่ต้องการเก็บไว้ในระหว่างเครื่องหมาย [และ]

การดึงข้อมูลออกมาจากอาร์เรย์จะใช้ดัชนี (index) ในการชี้ตำแหน่ง โดยใช้รูปแบบดังนี้

```
variable_name[ index ]
```

โดย variable_name คือ ชื่อตัวแปรที่กำหนด

Index คือ ดัชนีในการชี้ตำแหน่งของอาร์เรย์

นอกจากนี้ หากต้องการกำหนดหรือเปลี่ยนแปลงค่าในอาร์เรย์ สามารถทำได้แบบเดียวกับการเปลี่ยนค่าตัวแปร แต่ต้องมีการชี้ดัชนีที่ต้องเปลี่ยนด้วย

```
variable_name[ index ] = Value;
```

โดย variable_name คือ ชื่อตัวแปรที่กำหนด

Index คือ ดัชนีในการชี้ตำแหน่งของอาร์เรย์

Value คือ ค่าที่ต้องการเปลี่ยนแปลงหรือกำหนด

<u>ตัวอย่าง</u> เช่น

int Score[4]; หมายถึง ประกาศตัวแปรอาร์เรย์แบบไม่กำหนดข้อมูลจำนวน 5 ดัชนีดังนี้

Score[0]	Score[1]	Score[2]	Score[3]	Score[4]

int age[] = {18, 12, 16, 15} หมายถึง ประกาศตัวแปรอาร์เรย์พร้อมกับการกำหนดข้อมูล

age [0]	age [1]	age [2]	age [3]
18	12	16	15

A=age[0] หมายถึง กำหนดให้ A มีค่าเท่ากับตัวแปร Age ดัชนี 18

age[1] = 10; หมายถึง กำหนดให้อาร์เรย์ดัชนี 1 มีค่าเท่ากับ 10

age [0]	age [1]	age [2]	age [3]
18	10	16	15

4.5 การตั้งชื่อตัวแปร

1. ต้องขึ้นต้นด้วยตัวอักษรหรือเครื่องหมาย underscore เท่านั้น
2. ตัวต่อไปจะเป็นตัวเลขหรือตัวอักษรหรือเครื่องหมาย underscore
3. ชื่อมีความยาวไม่จำกัด (ตั้งยาวเกินได้ แต่ compiler จำแนกได้เพียง 31 ตัวแรกเท่านั้น ตั้งเกินจึงไม่มีประโยชน์)
4. ห้ามตั้งชื่อตรงกับคำสงวนในภาษาซี
5. ชื่อตัวแปรควรสื่อความหมายของตัวแปรเพื่อป้องกันความสับสนของการพิจารณาโปรแกรม

4.6 การเขียนโปรแกรมสื่อสารทางพอร์ตอนุกรม

4.6.1 ฟังก์ชั่นกำหนดความเร็วในการสื่อสารทางพอร์ตอนุกรม เพื่อให้สามารถสื่อสารระหว่างอุปกรณ์ทั้งสองฝั่งได้จะต้องกำหนดอัตราเร็วในการสื่อสารหรือเรียกว่าอัตราบอด (Baudrate) ซึ่งค่าความเร็วนั้นมีหน่วยเป็นบิตต่อวินาที (bps: bit per second) ค่าความเร็วนี้ได้แก่ 300, 600, 1200, 2400, 4800, 9600, 14400, 19200, 28800, 38400, 57600, หรือ 115200

```
Serial.begin(speed);
```

speed: ตัวเลขของอัตราเร็วในการสื่อสารผ่านพอร์ตอนุกรม

ตัวอย่าง เช่น

Serial.begin(9600); หมายถึง กำหนดอัตราเร็วของการสื่อสารทางพอร์ตอนุกรมเป็น 9600 บิตต่อ 1 วินาที

4.6.2 ฟังก์ชั่นส่งข้อมูลออกพอร์ต เป็นฟังก์ชั่นที่ใช้ในการส่งข้อมูลออกทางพอร์ตอนุกรมหรืออาจเรียกว่าฟังก์ชั่นพิมพ์ ข้อมูลออกทางพอร์ตเพื่อแสดงผลที่จอคอมพิวเตอร์ที่เชื่อมต่อกับวงจร Arduino ฟังก์ชั่นนี้เมื่อพิมพ์เสร็จตัว เคอร์เซอร์จะรออยู่ที่ท้ายสิ่งที่พิมพ์นั้น ๆ

```
Serial.print(variable)
```

```
Serial.print(variable, format)
```

โดย variable คือ ชื่อตัวแปรที่กำหนด
format คือ รูปแบบของการแสดงผล

ตัวอย่าง เช่น

Serial.print(A) ผลที่ได้คือแสดงข้อมูลของตัวแปร A
Serial.print(20) ผลที่ได้คือแสดงตัวเลข 20
Serial.print("Hello") ผลที่ได้คือแสดงข้อความ Hello

4.6.3 ฟังก์ชั่นส่งข้อมูลออกพอร์ต คล้ายกับฟังก์ชั่น Serial.print ต่างกันตรงที่เมื่อพิมพ์เสร็จตัวเคอร์เซอร์ขึ้นมารอยังบรรทัดใหม่ ดังนั้นเมื่อสั่งพิมพ์ครั้งถัดไปข้อมูลที่ปรากฏจะอยู่ที่บรรทัดใหม่ แทนที่จะต่อท้ายหมือนกับฟังก์ชั่น Serial.print

```
Serial.println(val)
```

ตัวอย่าง เช่น กำหนดให้ A=20

Serial.println(A) ผลที่ได้คือแสดงข้อความ 20

Serial.println(78) ผลที่ได้คือแสดงข้อความ 78

Serial.println("Hello") ผลที่ได้คือแสดงข้อความ Hello

Serial Monitor
20
78
Hello

ตัวอย่างที่ 4.1 การเขียนโปรแกรมแสดงข้อมูลตัวแปร 2 จำนวน โดยกำหนดให้ A=20, B=30 จากนั้นแสดงผลลัพธ์ออกทางหน้าต่าง Serial Monitor ซึ่งขั้นตอนการแก้ปัญหา มีดังนี้ คือ

ขั้นตอนที่ 1 การวิเคราะห์และกำหนดรายละเอียดของปัญหา

1) ปัญหาที่ต้องการแก้ คือ *การเขียนโปรแกรมแสดงข้อมูลตัวเลข 2 จำนวน*
2) ข้อมูลเข้า คือ *-ไม่มี-*
3) ข้อมูลออก คือ *ข้อมูลตัวเลข A และ B*

ขั้นตอนที่ 2 การออกแบบโปรแกรม (รหัสลำลองหรือผังงาน)

เขียนลำดับการทำงานโดยใช้รหัสลำลอง หรือผังงานได้ดังนี้

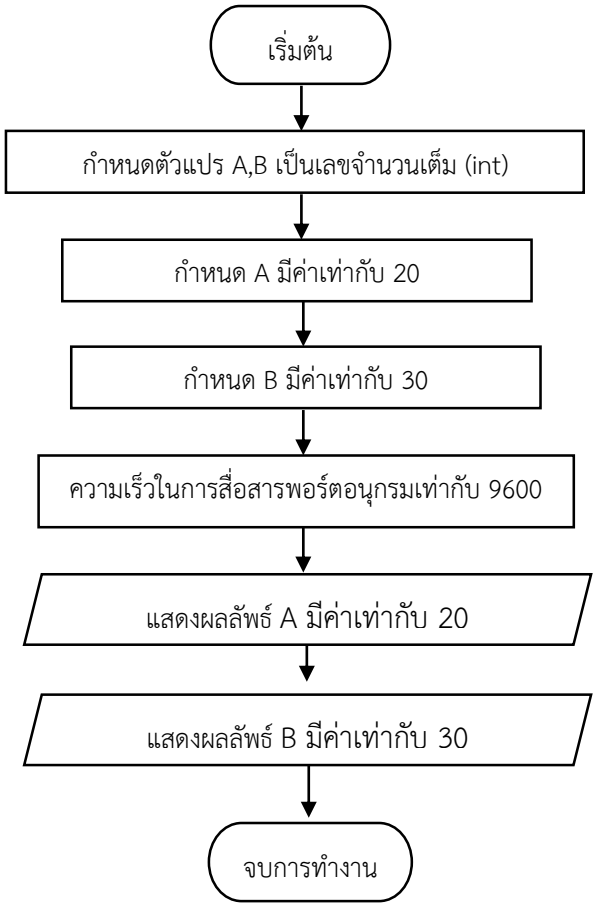
2.1) การเขียนรหัสลำลอง

เริ่มต้น

1. ประกาศตัวแปร A,B และ C เป็นเลขจำนวนเต็ม
2. กำหนดตัวแปร A มีค่าเท่ากับ 20
3. กำหนดตัวแปร B มีค่าเท่ากับ 30
4. กำหนดความเร็วในการสื่อสารพอร์ตอนุกรมเท่ากับ 9600
5. แสดงข้อมูล A มีค่าเท่ากับ 20
6. แสดงข้อมูล B มีค่าเท่ากับ 30

จบการทำงาน

2.2) การเขียนผังงาน



ขั้นตอนที่ 3 เขียนโค้ดคำสั่งโปรแกรมที่ใช้ในการทดลอง

	คำสั่ง	คำอธิบายโปรแกรม
1	int A,B;	//ประกาศตัวแปร A และ B เป็นเลขจำนวนเต็ม
2	void setup() {	
3	A=20;	//กำหนด A มีค่าเท่ากับ 20
4	B=30;	//กำหนด B มีค่าเท่ากับ 30
5	Serial.begin(9600);	// กำหนดอัตราเร็วของการสื่อสารเป็น 9600 บิตต่อ 1 วินาที
6	Serial.print("A = ");	//แสดงข้อมูลข้อความ "A="
7	Serial.println(A);	//แสดงข้อมูลตัวแปร A
8	Serial.print("B = ");	//แสดงข้อมูลข้อความ "B="
9	Serial.println(B);	//แสดงข้อมูลตัวแปร B
10	}	
11	void loop() {	
12	}	

ขั้นตอนที่ 4 การทดสอบโปรแกรม

4.1) **ทำการอัพโหลดโปรแกรม**

4.2) เปิดหน้าต่าง Serial Monitor

4.3) สังเกตผลการทำงาน ดังภาพ