INFOPRESS

คู่มือ Coding ภาษา C

ฉบับสมบูรณ์ 3rd EDITION

- อ่านเข้าใจง่าย มีแบบฝึกหัดและตัวอย่างพร้อมคำอธิบายทุกหัวข้อ
- มีตัวอย่างการพัฒนาโปรแกรมด้วยภาษา C
- แนะนำการเขียนโปรแกรมเกี่ยวกับ Arduino เบื้องต้น
- เรียบเรียงจากอาจารย์ที่มีประสบการณ์สอนมากกว่า 20 ปี
- เหมาะสำหรับนักเรียน นักศึกษา และผู้ที่สนใจเขียนโปรแกรมภาษา C

ดาวน์โหลดฟรี ไฟล์ตัวอย่างภายในเล่ม https://serazu.com/

9786164872363

ผศ.สุดา เรียรมนตรี | ไกรศร ตั้งธนาโอภากุล | กิตินันท์ พลสวัสดิ์

สารบัญ

บทที่ 1 รู้จักภาษา C กับการพัฒนาโปรแกรม 1

ความเป็นมาของภาษา C 1

จุดเด่นของภาษา C 3

โครงสร้างของภาษา C 3

ความรู้พื้นฐานการพัฒนาซอฟต์แวร์ 8

วงจรชีวิตของการพัฒนาซอฟต์แวร์ (Software Development Life Cycle – SDLC) 8

ระเบียบวิธีการพัฒนาซอฟต์แวร์ 10

สรุปเนื้อหาบทที่ 1 12

บทที่ 2 เตรียมพร้อมก่อนเขียนโปรแกรม 15

Visual Studio Community 2019 for Windows 15

ดาวน์โหลด Visual Studio Community 2019 16

ติดตั้งโปรแกรม Visual Studio Community 2019 17

เริ่มต้นเขียนโปรแกรมด้วย Visual Studio Community 2019 19

การบันทึกโปรเจกต์ (Save Project) 27

การเปิดโปรเจกต์ (Open Project) 27

การเขียนคำอธิบาย (Comment) 28

C-Free 5.0 Professional for Windows 29

ดาวน์โหลด C-Free 5.0 Professional 29

ติดตั้งโปรแกรม C-Free 5.0 Professional 30

เริ่มต้นเขียนโปรแกรมด้วย C-Free 5.0 Professional 35

การบันทึกโปรแกรม 36

การเปิดใช้งานโปรแกรม 37

Visual Studio Code for Linux 38

ดาวน์โหลด Visual Studio Code 39

ติดตั้งโปรแกรม Visual Studio Code 40

เริ่มต้นเขียนโปรแกรมด้วย Visual Studio Code 41
การบันทึกโปรแกรม 44
การเปิดใช้งานโปรแกรม 45
สรุปเนื้อหาบทที่ 2 47

บทที่ 3 ตัวแปรและชนิดของข้อมูล 49
ตัวแปร (Variable) 49
กฎการตั้งชื่อ 50
การประกาศตัวแปร 51
ชนิดของข้อมูล (Data Type) 51
Integer Type (ชนิดข้อมูลแบบจำนวนเต็ม) 52
Character Type (ชนิดข้อมูลแบบตัวอักขระ) 57
String Type (ชนิดข้อมูลแบบข้อความ) 59
Floating Point Type (ชนิดข้อมูลแบบจำนวนเลขทศนิยม) 61
ค่าคงที่ (Constants) 65
กฎของการแปลงชนิดของข้อมูล (Data Type Conversion) 67
Implicit Type Conversion 67
Explicit Type Conversion (Casting) 70
รู้จักกับ Null character และ Empty string 72
สรุปเนื้อหาบทที่ 3 74

บทที่ 4 นิพจน์และตัวดำเนินการ (Expressions and Operator) 75
นิพจน์ (Expression) 75
ตัวดำเนินการ (Operator) 76
ตัวดำเนินการกำหนดค่า (Assignment Operators) 76
ตัวดำเนินการทางคณิตศาสตร์ (Arithmetic Operators) 80
ตัวดำเนินการยูนารี (Unary Operators) 84
ตัวดำเนินการเปรียบเทียบ (Comparison Operators) 87
ตัวดำเนินการตรรกะ (Logical Operators) 91
ตัวดำเนินการแบบมีเงื่อนไข (Conditional Operators) 93
ตัวดำเนินการบอกขนาดชนิดข้อมูล (sizeof Operators) 95
ตัวดำเนินการระดับบิต (Bitwise Operators) 96

Bitwise Shift Left (<<) 101
Bitwise Shift Right (>>) 103
One's Complement (~) 106
ลำดับความสำคัญของตัวดำเนินการ (Operator of Precedence) 109
สรุปเนื้อหาบทที่ 4 112

บทที่ 5 การแสดงผลและการรับข้อมูล 113
การแสดงผลข้อมูล 113
การแสดงผลทีละตัวอักษรด้วยคำสั่ง putchar() 113
การแสดงผลเป็นข้อความด้วยคำสั่ง puts() 114
การแสดงผลข้อมูลทุกชนิดด้วยคำสั่ง printf() 116
รหัสรูปแบบการแสดงผลของคำสั่ง printf() 117
การจัดรูปแบบการแสดงผลข้อมูลของคำสั่ง printf() 120
การกำหนดการแสดงผลข้อมูลของคำสั่ง printf() 122
การจัดการพื้นที่แสดงผลข้อมูลของคำสั่ง printf() 124
การรับข้อมูล 127
การรับข้อมูลทีละตัวอักษรด้วยคำสั่ง getch() และ getchar() 127
การรับข้อมูลชนิดข้อความด้วยคำสั่ง gets_s() 130
การรับข้อมูลทุกชนิดด้วยคำสั่ง scanf() หรือ scanf_s() 132
การกำหนดลำดับการรับข้อมูลของคำสั่ง scanf() หรือคำสั่ง scanf_s() 136
การกำหนดจำนวนตัวอักษรในการรับค่าข้อมูลชนิดข้อความ
ของคำสั่ง scanf() หรือคำสั่ง scanf_s() 138
การกำหนดรูปแบบการรับค่าข้อมูลของคำสั่ง scanf() หรือคำสั่ง scanf_s() 139
การนับจำนวนตัวอักษรที่รับค่าข้อมูลด้วยคำสั่ง scanf() หรือคำสั่ง scanf_s() 141
สรุปเนื้อหาบทที่ 5 142

บทที่ 6 ควบคุมทิศทางการทำงานของโปรแกรมด้วยคำสั่งตัดสินใจ 145
คำสั่ง if – คำสั่งตัดสินใจแบบหนึ่งทางเลือก 145
คำสั่ง if...else – คำสั่งตัดสินใจแบบสองทางเลือก 148
คำสั่ง if...else if – คำสั่งตัดสินใจแบบหลายทางเลือก 151
คำสั่ง switch-case – คำสั่งตัดสินใจแบบหลายทางเลือก 155
สรุปเนื้อหาบทที่ 6 160

บทที่ 7 ควบคุมการทำงานของโปรแกรมด้วยคำสั่งวนลูป 163
คำสั่ง for – คำสั่งวนซ้ำด้วยจำนวนรอบที่แน่นอน 163
คำสั่ง while – คำสั่งวนซ้ำด้วยจำนวนรอบที่ไม่แน่นอน 166
คำสั่ง do...while – คำสั่งวนซ้ำโดยตรวจสอบเงื่อนไขหลังการทำงาน 169
สรุปเนื้อหาบทที่ 7 176

บทที่ 8 การเขียน Flowchart 177
หลักการเขียน Flowchart 177
แนวคิดการเขียน Flowchart 179
การทำงานแบบตามลำดับ (Sequence) 179
การทำงานแบบใช้คำสั่งตัดสินใจ (Decision) 182
การทำงานแบบใช้คำสั่งวนลูป (Loop) 188
สรุปเนื้อหาบทที่ 8 192

บทที่ 9 ตัวแปรอาร์เรย์ 193
ตัวแปรอาร์เรย์ (Array Variable) 193
ตัวแปรอาร์เรย์ 1 มิติ (One Dimensional Array) 193
ตัวแปรอาร์เรย์หลายมิติ (Multidimensional Array) 194
การเขียนโปรแกรมกับตัวแปรอาร์เรย์ 196
การประกาศตัวแปรอาร์เรย์ 196
การกำหนดค่าให้ตัวแปรอาร์เรย์ 197
การอ้างถึงข้อมูลในตัวแปรอาร์เรย์ 198
สรุปเนื้อหาบทที่ 9 206

บทที่ 10 พอยน์เตอร์ 207
ตัวแปรพอยน์เตอร์ (Pointer Variable) 207
การใช้งานพอยน์เตอร์ 209
การดำเนินการกับพอยน์เตอร์ 213
การใช้งานพอยน์เตอร์กับอาร์เรย์ 215
การจัดการพื้นที่หน่วยความจำแบบไดนามิก (Dynamic Memory Allocation) 222
รู้จักกับ Pointers to functions 229
สรุปเนื้อหาบทที่ 10 231

บทที่ 11 ฟังก์ชัน (Function) 233

รู้จักกับฟังก์ชัน พารามิเตอร์ และอาร์กิวเมนต์ 233

ฟังก์ชันที่สร้างขึ้นเอง (User-Defined Function) 235

ฟังก์ชันที่ไม่มีการส่งค่ากลับและไม่มีการรับค่าพารามิเตอร์ (void function with no parameter) 236

ฟังก์ชันที่ไม่มีการส่งค่ากลับแต่มีการรับค่าพารามิเตอร์ (void function with parameter) 239

ฟังก์ชันที่มีการส่งค่ากลับแต่ไม่มีการรับค่าพารามิเตอร์ (return value function with no parameter) 243

ฟังก์ชันที่มีการส่งค่ากลับและมีการรับค่าพารามิเตอร์ (return value function with parameter) 245

การส่งค่าผ่านพารามิเตอร์ (Parameter Passing) 249

ฟังก์ชันประเภทส่งผ่านค่า (Call by Value) 249

ฟังก์ชันประเภทส่งผ่านตัวอ้างอิง (Call by Reference) 251

ขอบเขตการทำงานของตัวแปร 253

ฟังก์ชันเรียกตัวเอง (Recursive Function) 256

ฟังก์ชันที่มาตรฐานภาษา C ได้สร้างมาให้แล้ว (Standard Library Function) 258

ไลบรารีฟังก์ชันการคำนวณทางคณิตศาสตร์ 258

ไลบรารีฟังก์ชันสำหรับตัวอักษร 259

ไลบรารีฟังก์ชันสำหรับข้อความ 260

สรุปเนื้อหาบทที่ 11 262

บทที่ 12 ตัวแปรประเภทโครงสร้างข้อมูล 263

โครงสร้างข้อมูลแบบ struct 263

การประกาศโครงสร้างข้อมูลแบบ struct 264

การประกาศโครงสร้างข้อมูลแบบ struct ด้วยคำสั่ง typedef 266

การใช้งานตัวแปร struct 268

การกำหนดค่าข้อมูลให้กับ struct 269

การอ่านค่าข้อมูลจาก struct 270

อาร์เรย์กับโครงสร้างข้อมูล 273

พอยน์เตอร์กับโครงสร้างข้อมูล 277

โครงสร้างข้อมูลซ้อนโครงสร้างข้อมูล 279

การใช้งานโครงสร้างข้อมูลกับฟังก์ชัน 282
การใช้งานโครงสร้างข้อมูลกับฟังก์ชันที่มีการส่งค่ากลับแต่ไม่มีการรับค่าพารามิเตอร์ 282
การใช้งานโครงสร้างข้อมูลกับฟังก์ชันที่ไม่มีการส่งค่ากลับแต่มีการรับค่าพารามิเตอร์ 286
การใช้งานโครงสร้างข้อมูลกับฟังก์ชันที่มีการส่งค่ากลับและมีการรับค่าพารามิเตอร์ 289
โครงสร้างข้อมูลแบบยูเนียน (union) 293
การประกาศโครงสร้างข้อมูลแบบ union 294
ประเภทข้อมูลแบบ enum (Enumeration Constants) 298
สรุปเนื้อหาบทที่ 12 300

บทที่ 13 รู้จักและใช้งาน Linked List 301
โครงสร้างข้อมูลแบบเชิงเส้น (Linear Data Structure) 301
Dense List 302
Linked List 302
สแตติกลิงก์ลิสต์ (Static Linked List) 302
ไดนามิกลิงก์ลิสต์ (Dynamic Linked List) 304
เริ่มต้นเขียนโปรแกรมกับ Linked List 305
สรุปเนื้อหาบทที่ 13 314

บทที่ 14 แฟ้มข้อมูล (File) 315
รู้จักกับแฟ้มข้อมูล 315
การเปิดและปิดไฟล์ (File Open/Close) 316
การเปิดไฟล์ (File Open) 316
การปิดไฟล์ (File Close) 319
การทำงานกับ Text File 319
การอ่านข้อมูลจากไฟล์ทีละตัวอักษรด้วยฟังก์ชัน getc() และฟังก์ชัน fgetc() 319
การเขียนข้อมูลลงไฟล์ทีละตัวอักษรด้วยฟังก์ชัน putc() และฟังก์ชัน fputc() 324
การอ่านข้อมูลจากไฟล์ทีละบรรทัดด้วยฟังก์ชัน fgets() 328
การเขียนข้อความลงไฟล์ด้วยฟังก์ชัน fputs() 331
การอ่านข้อมูลจากไฟล์โดยใช้ฟังก์ชัน fscanf() 333

การเขียนข้อมูลลงไฟล์ด้วยฟังก์ชัน fprintf() 334
การทำงานกับ Binary File 341
การอ่านข้อมูลจากไฟล์ด้วยฟังก์ชัน fread() 341
การเขียนข้อมูลลงไฟล์ด้วยฟังก์ชัน fwrite() 341
การตรวจสอบสถานะของไฟล์ (File Status Function) 348
การตรวจสอบ EOF (end of file) ด้วยฟังก์ชัน feof() 349
การตรวจสอบ error ที่เกิดขึ้นกับไฟล์ด้วยฟังก์ชัน ferror() 349
การหาตำแหน่งที่อยู่ของไฟล์พอยน์เตอร์ด้วยฟังก์ชัน ftell() 349
การย้ายตำแหน่งไฟล์พอยน์เตอร์ไปยังตำแหน่งต้นไฟล์ด้วยฟังก์ชัน rewind() 350
การย้ายตำแหน่งไฟล์พอยน์เตอร์ด้วยฟังก์ชัน fseek() 352
การดำเนินการเกี่ยวกับไฟล์ (File Operation) 356
การลบไฟล์ 356
การเปลี่ยนชื่อไฟล์ 357
สรุปเนื้อหาบทที่ 14 359

บทที่ 15 กรณีศึกษา ระบบบันทึกข้อมูลนักศึกษา 361
รายละเอียดระบบบันทึกข้อมูลนักศึกษา 362
ส่วนฟังก์ชันหลักและเมนูต่างๆ 363
ส่วนฟังก์ชันสำหรับเพิ่มข้อมูล 366
ส่วนฟังก์ชันสำหรับแก้ไขข้อมูล 368
ส่วนฟังก์ชันสำหรับลบข้อมูล 373
ส่วนฟังก์ชันสำหรับค้นหาข้อมูล 377
ส่วนฟังก์ชันสำหรับแสดงข้อมูลทั้งหมด 380
สรุปเนื้อหาบทที่ 15 383

บทที่ 16 แนะนำการประยุกต์ใช้ภาษา C กับ Arduino 385
รู้จักกับ Arduino 385
การใช้งานโปรแกรม Tinkercad 387
ขั้นตอนการสมัครสมาชิก 387
โครงสร้างภาษา C Arduino เบื้องต้น 389
สรุปเนื้อหาบทที่ 16 399

บทที่ 3

ตัวแปรและชนิดของข้อมูล

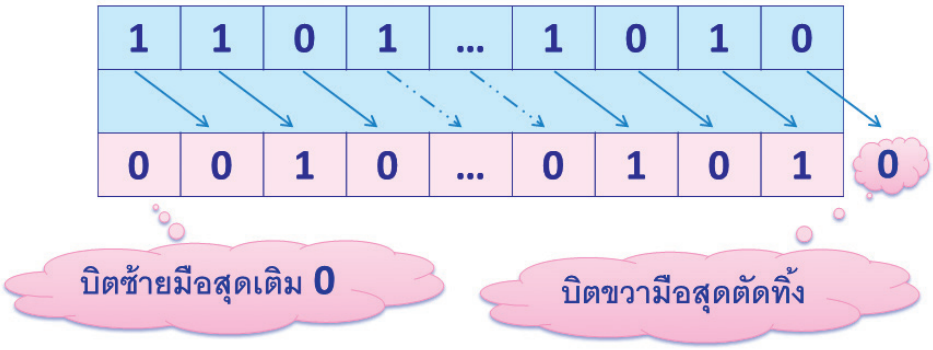
ในการพัฒนาโปรแกรมด้วยภาษา C สิ่งที่ผู้อ่านควรเริ่มต้นเรียนรู้ก็คือ การทำความเข้าใจพื้นฐานเกี่ยวกับตัวแปรและชนิดของข้อมูล รวมถึงความสัมพันธ์และความแตกต่างระหว่างข้อมูลในแต่ละชนิด ในบทนี้ผู้อ่านจะได้เรียนรู้สิ่งเหล่านี้อย่างละเอียด

ตัวแปร (Variable)

ตัวแปร (Variable) คือ สัญลักษณ์ที่ใช้แทนค่าข้อมูลในการเขียนโปรแกรม ซึ่งข้อมูลนั้นจะเป็นตัวหนังสือหรือตัวเลขก็ได้ โดยเบื้องหลังการทำงานของตัวแปรจะเป็นการจองพื้นที่ของหน่วยความจำสำหรับเก็บข้อมูลตามรูปแบบชนิดของข้อมูล

เพื่อให้เข้าใจมากยิ่งขึ้น ผู้อ่านสามารถศึกษาได้จากตัวอย่างต่อไปนี้

เมื่อมีการประกาศตัวแปร คอมไพเลอร์จะมีการจองพื้นที่ของหน่วยความจำสำหรับเก็บข้อมูลดังนี้



ตารางแสดงความสัมพันธ์การหารตัวเลข Shift Right

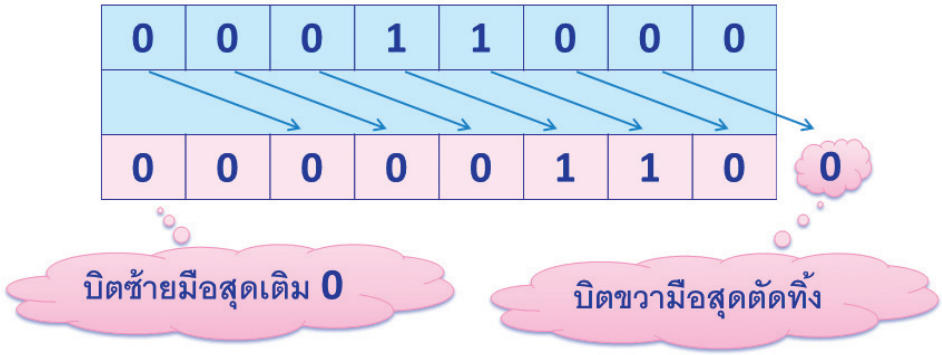
จำนวนการ Shift	ตัวเลขที่ใช้หาร	ตัวดำเนินการ Shift
1	2	>> 1
2	4	>> 2
3	8	>> 3
...	$2^{...}$	>> ...
y	2^{y}	>> y

เพื่อเพิ่มความเข้าใจ ผู้อ่านสามารถศึกษาได้จากตัวอย่างการทำงานของตัวดำเนินการ Bitwise Shift Right ดังนี้

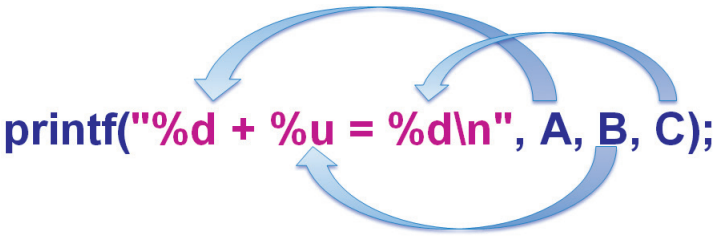
ตัวอย่างที่ 4.17 การหาค่าผลลัพธ์ของ 24 >> 2 โดยสมมติว่าใช้เลขจำนวน 8 บิต

ขั้นตอนที่ 1 แปลงเลขฐานสิบเป็นเลขฐานสองได้เป็น 0001 1000

ขั้นตอนที่ 2 เลื่อนบิตไปทางขวา 2 บิต และใส่ 0 ที่บิตซ้ายมือ

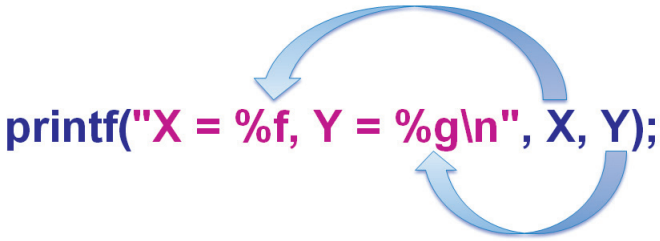


แสดงการ Shift Right 2 บิต



บรรทัดที่ 17 แสดงผลข้อความ X = 14.000000, Y = 4.66667 โดยแทนค่าตัวแปร X และ Y ที่รหัสรูปแบบแสดงผล ในคำสั่งนี้คือ %f และ %g ซึ่งเป็นรหัสแสดงผลทศนิยม และการแสดงผลค่าตัวแปรนี้จะแสดงตามลำดับของรหัสรูปแบบแสดงผลคือ

%f ตัวแรก	แทนที่ด้วยค่าตัวแปร X
%g ตัวที่สอง	แทนที่ด้วยค่าตัวแปร Y



การจัดรูปแบบการแสดงผลข้อมูลของคำสั่ง printf()

ในการแสดงผลการทำงานของโปรแกรม บ่อยครั้งที่เราต้องการแสดงอักขระพิเศษทางจอภาพ ซึ่งภาษา C ได้กำหนดรูปแบบการแสดงผลด้วยรหัสควบคุมหรืออักขระพิเศษต่างๆ ดังนี้

ซึ่งคำสั่ง for ในภาษา C มีรูปแบบคำสั่งดังนี้

```
for (initial; condition; change)
{
        statements;
}
```

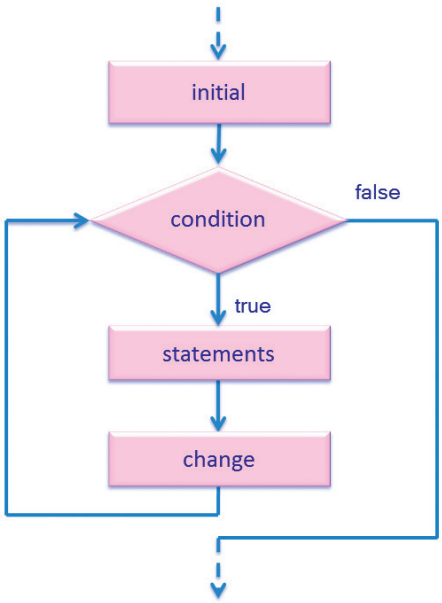
โดยที่ initial เป็นส่วนกำหนดค่าเริ่มต้นของตัวแปรที่ใช้กำหนดเงื่อนไข

condition เป็นส่วนกำหนดเงื่อนไขการวนลูป

change เป็นส่วนการเปลี่ยนแปลงของตัวแปรในแต่ละรอบ

statements เป็นชุดคำสั่งที่ต้องทำซ้ำเมื่อเงื่อนไขเป็นจริง

จากรูปแบบคำสั่ง for เราสามารถเขียนแผนภูมิการทำงาน (Flowchart) ได้ดังนี้



ตัวอย่างที่ 7.1 การใช้คำสั่ง for

```
for (count = 1; count <= n; count++)
        sum += count;

for (count = 0; count <= 100; count = count + 2)
{
        sum += count;
}
```

ตัวอย่างที่ 7.8 โปรแกรมการใช้คำสั่งวนลูปพิมพ์สามเหลี่ยมตัวเลขและสามเหลี่ยม * ซึ่งมีโค้ดดังนี้

```
#include <stdio.h>
#include <conio.h>

int main()
{
    int iRow;
    int iLimit;
    int iCol;

    printf("Input number integer between 1 and 9 : ");
    scanf_s("%d", &iLimit);

    for (iRow = 1; iRow <= iLimit; iRow++)
    {
        printf("\n");
        for (iCol = 1; iCol <= iLimit; iCol++)
        {
            if (iRow > iCol)
            {
                printf("*");
            }
            else
            {
                printf("%d", iCol);
            }
        }
    }
    _getch();
    return 0;
}
```

ผลการทำงานของโปรแกรมแสดงได้ดังนี้

```
Input number integer between 1 and 9 : 7

1234567
*234567
**34567
***4567
****567
*****67
******7
```

บทที่ 8

การเขียน Flowchart

Flowchart เป็นผังงานที่ใช้แสดงแนวคิดหรือขั้นตอนการทำงานของโปรแกรม ยังรวมไปถึงการเป็นเครื่องมือที่ช่วยให้มองเห็นภาพรวมของโปรแกรม ที่ทำให้เราเขียนโปรแกรมได้ง่ายยิ่งขึ้น เนื่องจากเราสามารถมองเห็นแนวคิดและทิศทางการทำงานของโปรแกรมแล้วนั่นเอง

หลักการเขียน Flowchart

จากคำกล่าวที่ว่า Flowchart เป็นผังงานที่ใช้แสดงแนวคิดหรือขั้นตอนการทำงานของโปรแกรม โดยใช้สัญลักษณ์แทนคำอธิบาย ไม่ว่าจะเป็นการใช้กรอบสี่เหลี่ยมเป็นสัญลักษณ์แทนการประมวลผล หรือจะเป็นการใช้ลูกศรแทนทิศทางการทำงานของโปรแกรม ซึ่งสามารถสรุปสัญลักษณ์การทำงานที่ควรทราบได้ดังนี้

สัญลักษณ์	การใช้งาน
	เริ่มต้นและสิ้นสุดการทำงานของโปรแกรม
	ประมวลผลการทำงานของโปรแกรม
	รับหรือแสดงผลโดยไม่กำหนดชนิดของอุปกรณ์
	การทำงานย่อย

จากตัวอย่างตัวแปรอาร์เรย์ intEx2 มีขนาด 3 แถว 5 คอลัมน์ สามารถเก็บข้อมูลได้ 15 ค่า โดยที่

- ตัวแปรแถวที่ 1 คอลัมน์ที่ 1 มีค่าเท่ากับ 3 ซึ่งสามารถเขียนได้เป็น intEx2[0][0] = 3
- ตัวแปรแถวที่ 1 คอลัมน์ที่ 2 มีค่าเท่ากับ 7 ซึ่งสามารถเขียนได้เป็น intEx2[0][1] = 7
- ตัวแปรแถวที่ 1 คอลัมน์ที่ 5 มีค่าเท่ากับ 1 ซึ่งสามารถเขียนได้เป็น intEx2[0][4] = 1
- ตัวแปรแถวที่ 3 คอลัมน์ที่ 1 มีค่าเท่ากับ 4 ซึ่งสามารถเขียนได้เป็น intEx2[2][0] = 4
- ตัวแปรแถวที่ 3 คอลัมน์ที่ 2 มีค่าเท่ากับ 2 ซึ่งสามารถเขียนได้เป็น intEx2[2][1] = 2
- ตัวแปรแถวที่ 3 คอลัมน์ที่ 5 มีค่าเท่ากับ 3 ซึ่งสามารถเขียนได้เป็น intEx2[2][4] = 3

ตัวแปรอาร์เรย์ 3 มิติ (Three Dimensional Array)

ตัวแปรอาร์เรย์ 3 มิติ เปรียบได้กับการนำตัวแปรมาเรียงต่อกันหลายๆ ตัวในลักษณะของกล่องข้อมูล หรือเรียกอีกอย่างว่า อาร์เรย์ของอาร์เรย์ 2 มิติ

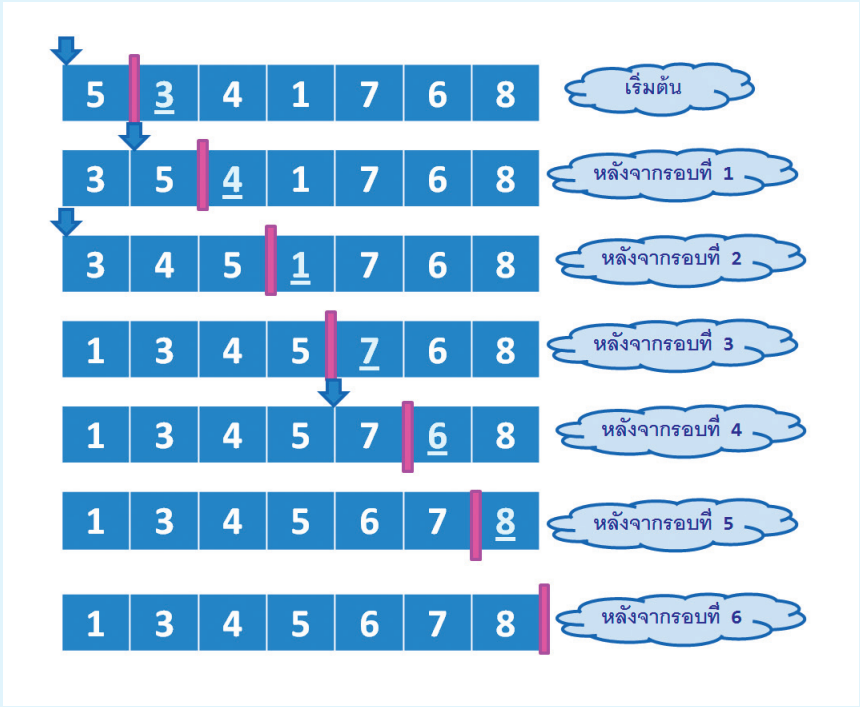
ในที่นี้ผู้เขียนจะขออธิบายโดยการจำลองตัวอย่างตัวแปรอาร์เรย์ 3 มิติ ชื่อตัวแปร intEx3 เป็นตัวแปรชนิดจำนวนเต็มที่เก็บข้อมูลจำนวนเต็มได้ 24 ค่า ดังรูป

จากตัวอย่างตัวแปรอาร์เรย์ intEx3 มีขนาด 2 บล็อก 3 แถว 4 คอลัมน์ สามารถเก็บข้อมูลได้ 24 ค่า โดยที่

- ตัวแปรบล็อกที่ 1 แถวที่ 1 คอลัมน์ที่ 1 มีค่าเท่ากับ 9 เขียนได้เป็น intEx3[0][0][0] = 9
- ตัวแปรบล็อกที่ 1 แถวที่ 1 คอลัมน์ที่ 2 มีค่าเท่ากับ 2 เขียนได้เป็น intEx3[0][0][1] = 2
- ตัวแปรบล็อกที่ 1 แถวที่ 1 คอลัมน์ที่ 3 มีค่าเท่ากับ 1 เขียนได้เป็น intEx3[0][0][2] = 1
- ตัวแปรบล็อกที่ 1 แถวที่ 2 คอลัมน์ที่ 1 มีค่าเท่ากับ 8 เขียนได้เป็น intEx3[0][1][0] = 8
- ตัวแปรบล็อกที่ 1 แถวที่ 2 คอลัมน์ที่ 2 มีค่าเท่ากับ 5 เขียนได้เป็น intEx3[0][1][1] = 5
- ตัวแปรบล็อกที่ 1 แถวที่ 3 คอลัมน์ที่ 4 มีค่าเท่ากับ 7 เขียนได้เป็น intEx3[0][2][3] = 7

Note

หลักการเรียงข้อมูล Insertion Sort จากน้อยไปมาก มีขั้นตอนดังนี้



แนวคิดคือ จะเริ่มวนลูปตั้งแต่สมาชิกใน Array ตัวที่ 2 จนถึงตัวสุดท้ายของ Array โดยในการวนลูปแต่ละครั้งจะหาตำแหน่งที่เหมาะสมกับค่าสมาชิกใน Array นั้นๆ

1. จากภาพเริ่มต้นครั้งแรกวนลูปในตำแหน่ง Array ตัวที่ 2 คือ ค่า 3 จากนั้นหาตำแหน่งที่เหมาะสมคือ 3 น้อยกว่า 5 ดังนั้น ให้ 3 อยู่หน้า 5

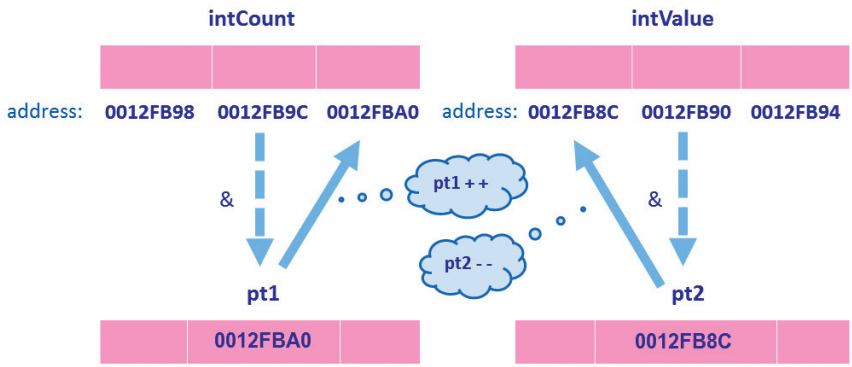
2. เริ่มวนลูปไปตำแหน่งที่ Array ตัวที่ 3 คือ ค่า 4 จากนั้นหาตำแหน่งที่เหมาะสมคือ 4 มากกว่า 3 แต่น้อยกว่า 5 ดังนั้น 4 อยู่ระหว่าง 3 กับ 5

3. เริ่มวนลูปไปตำแหน่งที่ Array ตัวที่ 4 คือ ค่า 1 จากนั้นหาตำแหน่งที่เหมาะสมคือ 1 น้อยกว่า 3 ซึ่ง 3 อยู่ในตำแหน่งแรก ดังนั้น 1 จะไปอยู่ในตำแหน่งแรกข้างหน้า 3

4. เริ่มวนลูปไปตำแหน่งที่ Array ตัวที่ 5 คือ ค่า 7 จากนั้นหาตำแหน่งที่เหมาะสมคือ 7 มีค่ามากกว่า 5 ที่เป็นค่ามากสุดในส่วนของข้อมูลที่เรียงแล้ว ดังนั้น 7 จะอยู่หลัง 5 เหมือนเดิม

5. เริ่มวนลูปไปตำแหน่งที่ Array ตัวที่ 6 คือ ค่า 6 จากนั้นหาตำแหน่งที่เหมาะสมคือ 6 มากกว่า 5 แต่น้อยกว่า 7 ดังนั้น 6 อยู่ระหว่าง 5 กับ 7

6. เริ่มวนลูปไปตำแหน่งที่ Array ตัวที่ 7 คือ ค่า 8 จากนั้นหาตำแหน่งที่เหมาะสมคือ 8 มีค่ามากกว่า 7 ที่เป็นค่ามากสุดในส่วนของข้อมูลที่เรียงแล้ว ดังนั้น 8 จะอยู่หลัง 7 เหมือนเดิม



อธิบายการดำเนินการกับตัวแปรพอยน์เตอร์ pt1 และ pt2

บรรทัดที่ 20 แสดงผลค่าตัวแปรพอยน์เตอร์ pt1 ซึ่งเป็นตำแหน่งที่อยู่ของตัวแปร intCount

บรรทัดที่ 21 แสดงผลค่าตัวแปรพอยน์เตอร์ pt2 ซึ่งเป็นตำแหน่งที่อยู่ของตัวแปร intValue

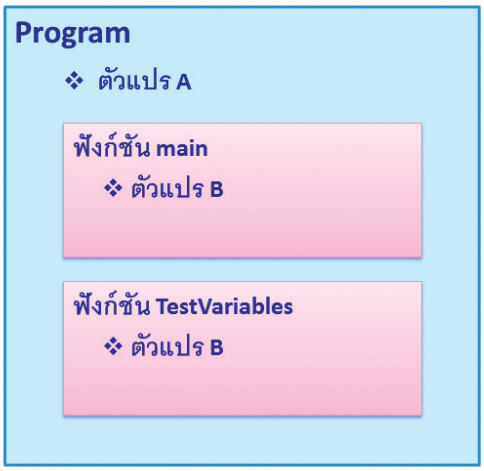
การใช้งานพอยน์เตอร์กับอาร์เรย์

ความสามารถอีกอย่างหนึ่งที่น่าสนใจของพอยน์เตอร์ที่ผู้อ่านจะได้เรียนรู้ก็คือ การใช้พอยน์เตอร์อ้างถึงตำแหน่งที่อยู่ของตัวแปรอาร์เรย์ ตัวแปรอาร์เรย์ในที่นี้รวมไปถึงข้อมูลชุดข้อความซึ่งเป็นตัวแปรอาร์เรย์ของอักขระด้วย

พอยน์เตอร์ที่ใช้อ้างอิงตำแหน่งที่อยู่ของตัวแปรอาร์เรย์ โดยปกติจะเป็นการอ้างถึงตำแหน่งที่อยู่ตำแหน่งแรกของตัวแปรอาร์เรย์เท่านั้น (อินเด็กซ์เป็น 0) เนื่องจากตำแหน่งอื่นสามารถเข้าถึงได้ในลำดับถัดๆ ไปต่อจากตำแหน่งแรกของตัวแปรอาร์เรย์นั่นเอง

เพื่อให้ผู้อ่านเข้าใจการใช้งานพอยน์เตอร์กับตัวแปรอาร์เรย์ ให้ผู้อ่านศึกษาจากการทำงานของตัวอย่างต่อไปนี้

ตัวอย่างที่ 11.9 โปรแกรมแสดงขอบเขตการทำงานของตัวแปร โดยการประกาศการใช้งานตัวแปรของโปรแกรมนี้ แสดงได้ดังรูป



และเขียนโค้ดโปรแกรมดังนี้

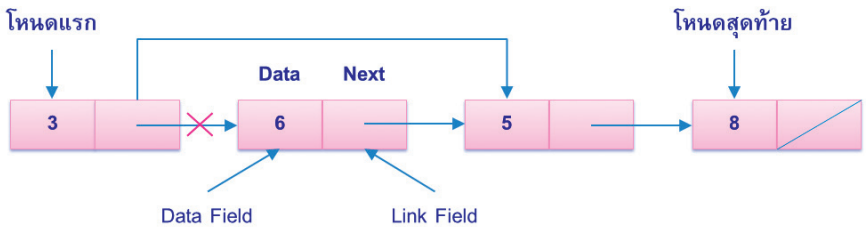
```
1   #include <stdio.h>
2   #include <conio.h>
3
4   void TestVariables (int B);
5
6   int A = 3;
7
8   int main()
9   {
10      int B = 5;
11
12      printf("A = %d, B = %d\n\n", A, B);
13      TestVariables(B);
14      printf("A = %d, B = %d", A, B);
15      _getch();
16     return 0;
17  }
18
19  void TestVariables (int B)
20  {
21     A = A + 5;
22     B = B + 5;
23     printf("A = %d, B = %d\n\n", A, B);
24  }
```

จากโค้ดตัวอย่าง เป็นการเพิ่มโหนดข้อมูลในตำแหน่งเริ่มต้นของ Linked List และลบโหนดข้อมูลที่ 2 ออกจาก Linked List ซึ่งการลบโหนดดังกล่าวเป็นการกำหนดให้ Link Field ของโหนดแรกชี้ไปยังตำแหน่งที่อยู่ของโหนดที่ 3 นั่นเอง โดยผลการทำงานของโปรแกรมแสดงได้ดังนี้

```
Enter number : 8
Enter number : 5
Enter number : 6
Enter number : 3
Enter number : 0

Output : 3
Output : 5
Output : 8
```

จากโค้ดตัวอย่าง สามารถอธิบายด้วยภาพได้ดังนี้



อธิบายการทำงานของโปรแกรม

บรรทัดที่ 5 ประกาศตัวแปร linkNext เป็นตัวแปรพอยน์เตอร์ของโครงสร้างข้อมูล struct ชื่อ listData

บรรทัดที่ 6-10 ประกาศโครงสร้างข้อมูลแบบ struct ชื่อ listData ประกอบด้วยสมาชิกดังนี้

- ข้อมูล เป็นตัวแปรชนิดจำนวนเต็มชื่อ Data
- Link Field เป็นตัวแปรพอยน์เตอร์ชื่อ next

บรรทัดที่ 14 ประกาศตัวแปรพอยน์เตอร์สำหรับเก็บตำแหน่งที่อยู่ของข้อมูล

บรรทัดที่ 16 กำหนดค่าให้ตัวแปรพอยน์เตอร์ link เท่ากับ NULL

บรรทัดที่ 18-19 รับค่าข้อมูลจำนวนเต็มทางแป้นพิมพ์

บรรทัดที่ 21-30 วนลูปจนกว่าค่าข้อมูลจำนวนเต็มที่รับเข้ามาจะมีค่าเป็น 0

บรรทัดที่ 23 กำหนดให้ตัวแปร temp เป็นโหนดใหม่สำหรับเก็บข้อมูล

บรรทัดที่ 24 กำหนดข้อมูลให้กับโหนด temp

บรรทัดที่ 25 กำหนดตำแหน่งที่อยู่ของโหนดแรกให้กับ Link Field ของโหนด temp ทำให้โหนด temp เป็นโหนดแรกของ Linked List

การเขียนข้อมูลลงไฟล์ด้วยฟังก์ชัน fprintf()

จากบทที่ 5 ผู้อ่านคงรู้จักฟังก์ชัน printf() เป็นอย่างดีแล้ว ซึ่งเป็นฟังก์ชันสำหรับแสดงข้อมูลทางจอภาพ แต่ถ้าผู้อ่านต้องการแสดงหรือบันทึกข้อมูลลงไฟล์จะต้องใช้ฟังก์ชัน fprintf() ซึ่งตัวอักษร f ที่อยู่หน้าคำว่า printf เป็นการบ่งบอกถึงการแสดงหรือบันทึกลงไฟล์นั่นเอง ซึ่งมีรูปแบบการใช้งานดังนี้

```
fprintf(filePointerName,"string format", varName);
```

โดยที่ filePointerName เป็นชื่อตัวแปรไฟล์พอยน์เตอร์ที่เราสร้างขึ้น เพื่อชี้ไปยังไฟล์ที่ต้องการเขียนข้อมูล

string format เป็นรหัสรูปแบบของข้อความในภาษา C

varName เป็นตัวแปรเก็บข้อมูลที่อ่านมา ซึ่งต้องมีชนิดข้อมูลตรงกับ string format

ตารางรหัสรูปแบบข้อมูล Input และ Output

รหัสรูปแบบ	ความหมาย
%c	ใช้แทนค่าตัวแปรตัวอักษรตัวเดียว (char)
%d	ใช้แทนค่าตัวแปรจำนวนเต็ม (int)
%ld	ใช้แทนค่าตัวแปรจำนวนเต็มยาว (long int)
%hd	ใช้แทนค่าตัวแปรจำนวนเต็มสั้น (short int)
%u	ใช้แทนค่าตัวแปรที่เก็บค่าเป็นจำนวนเต็มบวก (unsigned int)
%hu	ใช้แทนค่าตัวแปรจำนวนเต็มสั้นแบบไม่คิดเครื่องหมาย (unsigned short int)
%lu	ใช้แทนค่าตัวแปรจำนวนเต็มยาวแบบไม่คิดเครื่องหมาย (unsigned long int)
%o	ใช้แทนค่าตัวแปรในรูปฐานแปด (int)
%ho	ใช้แทนค่าตัวแปรในรูปฐานแปด (short)
%lo	ใช้แทนค่าตัวแปรในรูปฐานแปด (long)
%x หรือ %X	ใช้แทนค่าตัวแปรในรูปฐานสิบหก (int)
%hx หรือ %hX	ใช้แทนค่าตัวแปรในรูปฐานสิบหก (short)
%lx หรือ %lX	ใช้แทนค่าตัวแปรในรูปฐานสิบหก (long)
%f	ใช้แทนค่าตัวแปรจำนวนจริงที่เป็นทศนิยม (float)
%lf	ใช้แทนค่าตัวแปรจำนวนจริงที่เป็นทศนิยม (double)

ในบทนี้ผู้เขียนจะยกตัวอย่างกรณีศึกษา ระบบบันทึกข้อมูลนักศึกษาในส่วนของการ “เขียนโปรแกรม” โดยระบบนี้สามารถบันทึก แก้ไข ลบ และค้นหาข้อมูล เพื่อให้ผู้อ่านได้ศึกษาและนำไปประยุกต์ใช้กับระบบที่ผู้อ่านได้ออกแบบและพัฒนาต่อไป

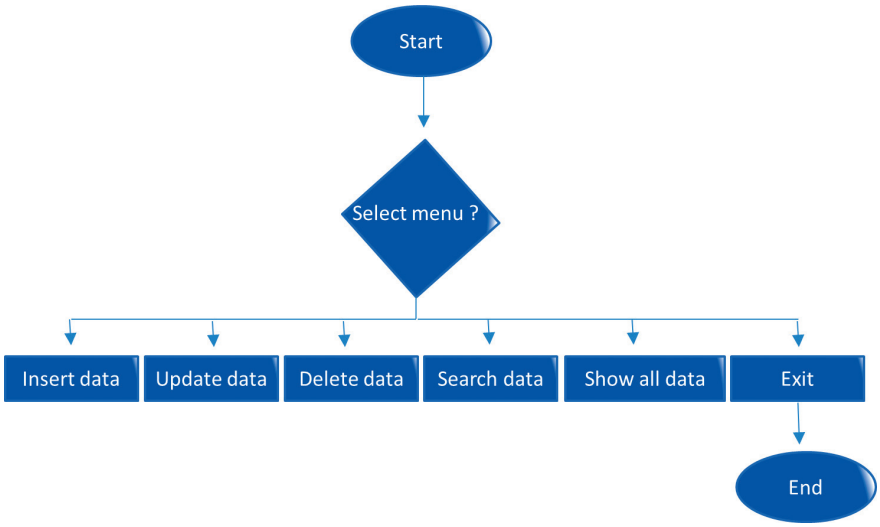
รายละเอียดระบบบันทึกข้อมูลนักศึกษา

ระบบบันทึกข้อมูลนักศึกษาเป็นระบบบันทึกข้อมูลเบื้องต้นของนักศึกษา โดยข้อมูลที่บันทึกจะประกอบด้วยรหัสนักศึกษา, คำนำหน้าชื่อ, ชื่อ, นามสกุล, ที่อยู่ และเบอร์ติดต่อ

Field Name	Description
ID	รหัสนักศึกษา
title_name	คำนำหน้าชื่อ
first_name	ชื่อ
last_name	นามสกุล
address	ทีอยู่
tel	เบอร์ติดต่อ

ระบบนี้จะประกอบด้วยส่วนต่างๆ ดังนี้

- **ส่วนฟังก์ชันหลักและเมนูต่างๆ** (ผู้อ่านอาจเขียนส่วนเมนูนี้แยกออกเป็นฟังก์ชันหนึ่งก็ได้)
- **ส่วนเพิ่มข้อมูล** ฟังก์ชัน ProcessInsertData()
- **ส่วนแก้ไขข้อมูล** ฟังก์ชัน ProcessUpdateData()
- **ส่วนลบข้อมูล** ฟังก์ชัน ProcessDeleteData()
- **ส่วนค้นหาข้อมูล** ฟังก์ชัน ProcessSearchData()
- **ส่วนแสดงข้อมูลทั้งหมด** ฟังก์ชัน ProcessShowAllData()



ส่วนฟังก์ชันหลักและเมนูต่างๆ

ในส่วนนี้จะเป็นส่วนของฟังก์ชันหลัก main() และส่วนของเมนูต่างๆ ที่มีอยู่ในระบบ โดยมีโค้ดการทำงานของโปรแกรมดังนี้

```
1   #define _CRT_SECURE_NO_WARNINGS
2   #include<stdio.h>
3   #include<conio.h>
4   #include<ctype.h>
5   #include <stdlib.h>
6   #include <string.h>
7
8   int ProcessInsertData(void);
9   int ProcessUpdateData(void);
10  int ProcessDeleteData(void);
11  int ProcessSearchData(void);
12  int ProcessShowAllData(void);
13
14  struct PERSON
15  {
16      char ID[11];
17      char title_name[10];
18      char first_name[30];
19      char last_name[30];
20      char address[100];
21      char tel[22];
22  }student;
```

บทที่ 16

แนะนำการประยุกต์ใช้ภาษา C กับ Arduino

เป็นที่ทราบกันว่าภาษา C ได้ถูกนำมาใช้เพื่อการเขียนโปรแกรมควบคุมการทำงานของระบบอุปกรณ์ที่เชื่อมต่อกับไมโครคอนโทรลเลอร์ (Microcontroller) ซึ่งก็คือ อุปกรณ์อิเล็กทรอนิกส์ชนิดหนึ่งที่สามารถควบคุมการทำงานด้วยระบบคอมพิวเตอร์ขนาดเล็ก โดยไมโครคอนโทรลเลอร์ MCS-51 ถือได้ว่าเป็นตระกูลที่เก่าแก่และได้รับความนิยมมาอย่างต่อเนื่อง แต่ด้วยความก้าวหน้าของการพัฒนาเทคโนโลยีทางด้านอิเล็กทรอนิกส์ที่เป็นไปอย่างรวดเร็ว Arduino จึงถูกพัฒนาให้เป็นไมโครคอนโทรลเลอร์ที่มีความสามารถในการติดต่อสื่อสาร และควบคุมการรับส่งสัญญาณทางไฟฟ้าตามเงื่อนไขต่างๆ ด้วยการเขียนโปรแกรมภาษา C ซึ่งผู้อ่านจะได้เรียนรู้ในบทนี้

รู้จักกับ Arduino

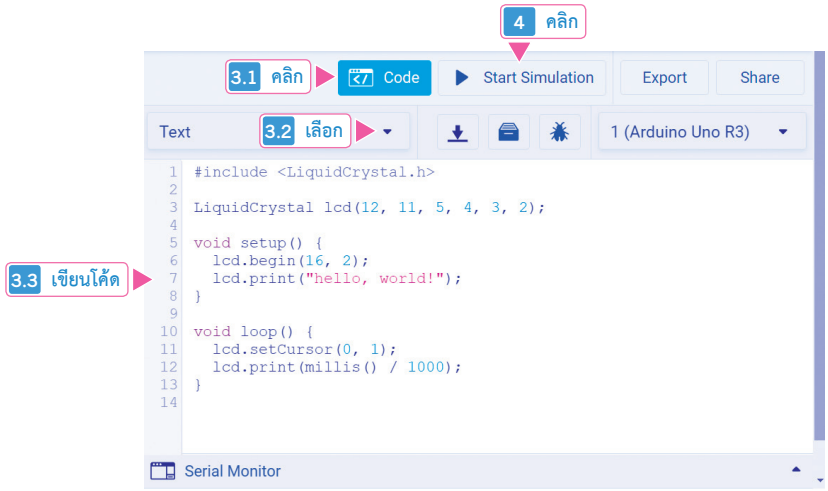
Arduino เป็นบอร์ดไมโครคอนโทรลเลอร์ตระกูล AVR ที่มีการพัฒนาแบบ Open Source คือมีการเปิดเผยข้อมูลทั้งด้าน Hardware และ Software ถูกออกแบบมาให้ใช้งานได้ง่ายและมีราคาถูก โดยมีส่วนการทำงานใน 2 ส่วนหลักคือ

1. ฮาร์ดแวร์ (Hardware)
2. ซอฟต์แวร์ (Software)

จากรายการ Component List ของตัวอย่างโปรแกรมที่ 16.3 อธิบายได้ว่า วงจรบอร์ด Arduino ประกอบด้วยอุปกรณ์ดังนี้

- U1 : บอร์ด Arduino Uno R3 จำนวน 1 บอร์ด
- U2 : จอ LCD ขนาด 16 x 2 จำนวน 1 จอ ใช้แสดงผลเป็นตัวอักษร (Character) ตามช่องขนาด 16 x 2 ซึ่งหมายถึงใน 1 บรรทัดแสดงผลได้ 16 ตัวอักษร และมีทั้งหมด 2 บรรทัด
- Rpot1 : ตัวต้านทาน Potentiometer จำนวน 1 ตัว
- R1 : ตัวต้านทาน Resistor ขนาด 220 โอห์ม จำนวน 1 ตัว

3. คลิกปุ่ม Code เพื่อเข้าสู่ส่วนของการเขียนโปรแกรม และเขียนโค้ดโปรแกรมดังนี้



4. คลิกปุ่ม Start Simulation เพื่อรันโปรแกรม โดยผลการทำงานคือ โปรแกรมจะแสดงข้อความทางจอ Character LCD โดยแสดงผลลัพธ์ของโปรแกรมดังนี้