

การทดลองที่ 5 การพัฒนาโปรแกรมด้วยภาษา C

การทดลองนี้คาดว่าผู้อ่านเคยเรียนการเขียนหรือพัฒนาโปรแกรมด้วยภาษา C มาบ้างแล้ว และมีความคุ้นเคยกับ IDE (Integrated Development Environment) จากพัฒนาโปรแกรมและการดีบั๊กโปรแกรมด้วยภาษา C/C++ ดังนั้น การทดลองมีวัตถุประสงค์เหล่านี้

- เพื่อให้เข้าใจการพัฒนาซอฟต์แวร์ด้วย IDE ชื่อ Code::Blocks บนระบบปฏิบัติการ Raspbian/Linux/Unix
- เพื่อให้เข้าใจความแตกต่างระหว่างการพัฒนาโปรแกรมภาษา C ด้วย Integrated Development Environment และ Makefile
- เพื่อให้สามารถสร้าง Makefile เพื่อความสะดวกและพัฒนาศักยภาพการทำงานเป็นนักพัฒนาอาชีพ

E.1 การพัฒนาโดยใช้ IDE

โปรแกรมหรือแอปพลิเคชัน IDE ย่อมาจาก Integrated Development Environment IDE ทำหน้าที่ช่วยเหลือโปรแกรมเมอร์ ทดสอบ และควบคุมซอร์สโค้ดให้เป็นปัจจุบัน ขั้นตอนการทดลองนี้เริ่มต้นโดย

1. ติดตั้ง Code::Blocks ผู้อ่านต้องพิมพ์คำสั่งเหล่านี้ลงบนโปรแกรม Terminal

```
$ sudo apt-get install codeblocks
```

คำสั่ง sudo นำหน้าคำสั่งใดๆ นี่จะเป็นการเรียกใช้งานคำสั่งนั้นด้วยสิทธิ์ระดับ superuser การติดตั้งจะดาวน์โหลดโปรแกรมผ่านทางเครือข่ายอินเทอร์เน็ต จำเป็นต้องใช้สิทธิ์ระดับสูงสุดนี้

2. พิมพ์คำสั่งนี้ เพื่อเริ่มต้นใช้งาน Code::Blocks

```
$ codeblocks
```

3. การใช้งาน Code::Blocks ครั้งแรกจะเป็นการติดตั้งค่า compiler plug-ins เป็น GNU GCC compiler.

4. หน้าต่างหลักจะปรากฏขึ้น หลังจากนั้น ผู้อ่านควรกด "Create a new project" เพื่อสร้างโปรเจกต์ใหม่ในหน้าต่าง "New from template"

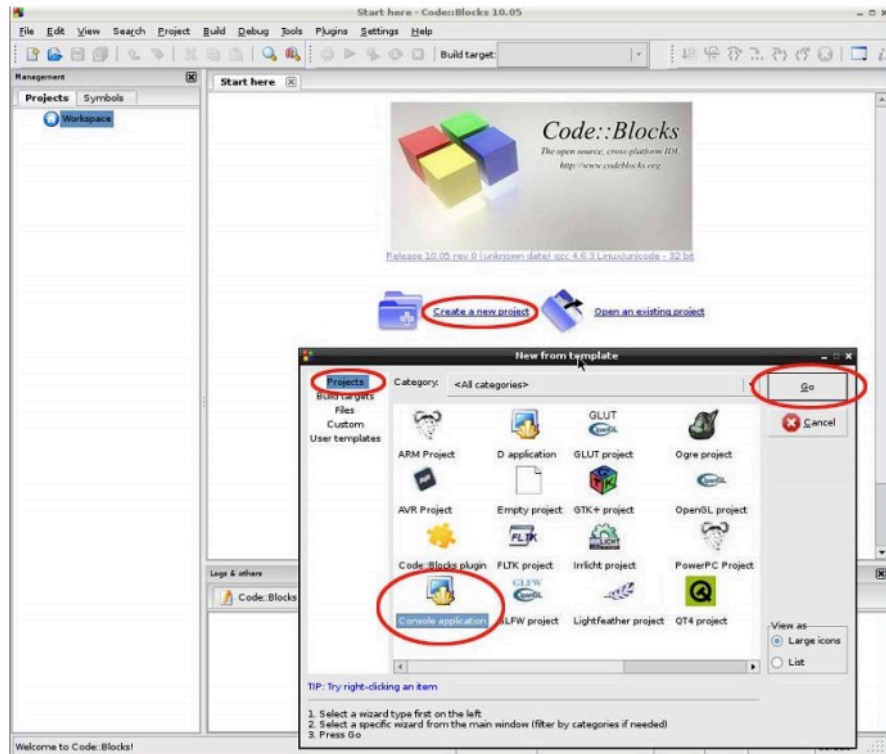


Figure E.1: หน้าต่างเลือกชนิดโปรเจกต์ที่จะพัฒนาเป็นชนิด "Console application"

5. เลือก 'New Projects' ในช่องด้านซ้าย แล้วเลือก "Console application" ในช่องด้านขวาเพื่อสร้างโปรแกรมในรูปแบบเท็กซ์โหมด (Text Mode) กดปุ่ม "Go" ตามรูปที่ E.1

6. กดปุ่ม Next> เพื่อดำเนินการต่อ

7. หน้าต่าง "Console application" จะปรากฏขึ้น กดเลือกภาษา "C" เพื่อพัฒนาโปรแกรมก่อนแล้วกดปุ่ม "Next>" ตามรูปที่ E.2)

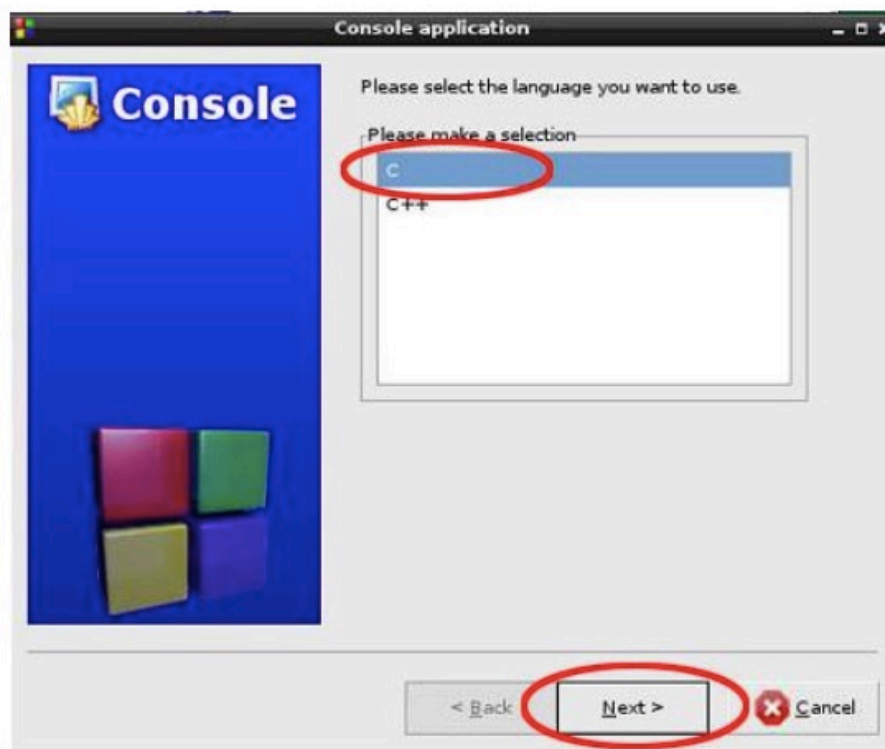


Figure E.2: หน้าต่างเลือกภาษาสำหรับโปรเจกต์ที่จะพัฒนา

8. กรอกชื่อโปรเจกต์ใหม่ชื่อ Lab5 ในช่อง Project title: และกรอกชื่อโฟลเดอร์ /home/pi/c/ ในช่อง Folder to create project in: โปรดสังเกตข้อความในช่อง Project filename: ว่าตรงกับ Lab5.cbpp ใช่หรือไม่
9. กดปุ่ม "Next>" เพื่อดำเนินการต่อและสุดท้ายจะเป็นขั้นตอนการเลือกคอนฟิกูเรชัน (Configuration) สำหรับคอมพิวเตอร์ในรูปที่ E.3 โดย Debug เหมาะสำหรับการเริ่มต้นและแก้ไขข้อผิดพลาดแล้วจึงกดปุ่ม "Finish" เมื่อเสร็จสิ้น

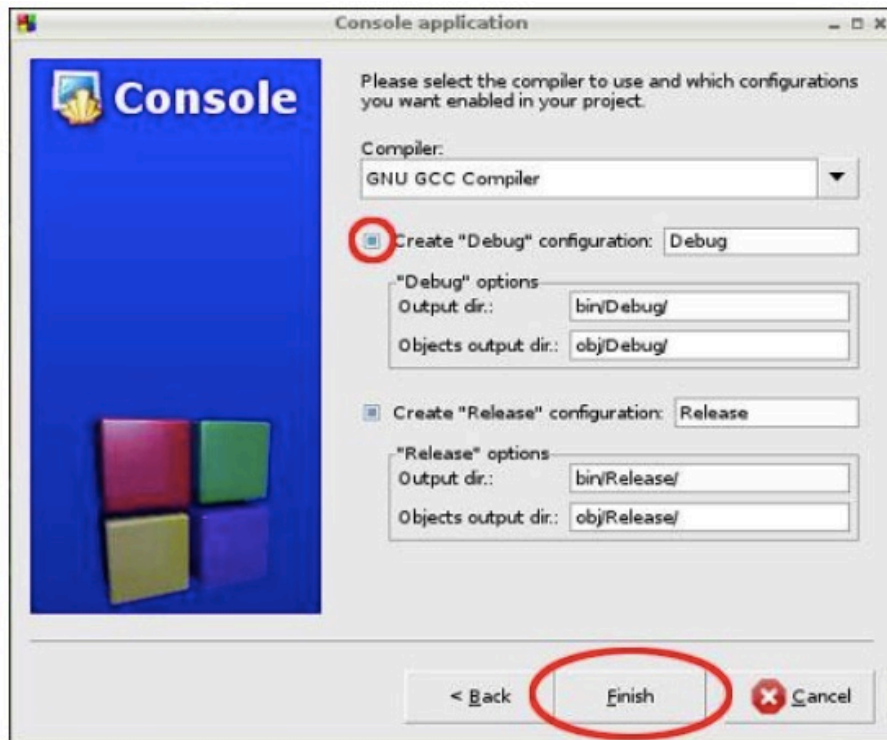


Figure E.3: การเลือกคอนฟิก Debug สำหรับคอมไพเลอร์ GNU GCC ในโปรเจกต์ Lab5

10. เพิ่มไฟล์เปล่าด้วยเมนูต่อไปนี้ File->New->Empty file ตามรูปที่ E.4

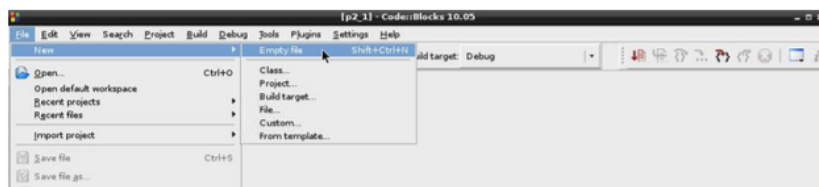


Figure E.4: การเพิ่มไฟล์เปล่าให้กับโปรเจกต์ Lab5 ที่สร้างขึ้น

11. ป้อนโปรแกรมนีกลงในหน้าต่าง main.c

```
#include <stdio.h>
int main(void)
{
    int a;
    printf("Please input an integer: ");
    scanf("%d", &a);
    printf("You entered the number: %d\n", a);
    return 0;
}
```

12. คอมไพล์และ Build โปรแกรม จนไม่มีข้อผิดพลาด โดยสังเกตจากหน้าต่างด้านล่างสุด

13. รันโปรแกรมเพื่อทดสอบการทำงาน

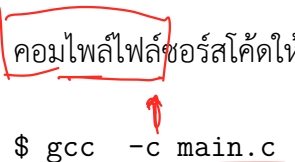
E.2 การดีบั๊ก (Debugging) โดยใช้ IDE

การดีบั๊กโปรแกรม คือ การตรวจสอบการทำงานของโปรแกรมอย่างละเอียด code::Blocks รองรับการดีบั๊ก ผ่านเมนู Debug ผู้อ่านสามารถเริ่มต้นโดย

1. กด Debug บนเมนูแถวบนสุด เลือก Active Debuggers ซึ่งค่าปัจจุบัน (Target's Default) คือ GDB/CDB Debugger
2. ตั้งเบรกพอยท์ (Break Point) ตรงประโยคที่ต้องการศึกษา โดยเลื่อนเคอร์เซอร์ (Cursor) ไปยังบรรทัดนั้น แล้วกดปุ่ม F5 โปรแกรมจะตั้งจุดตัดประโยคจะมีวงกลมสีแดงปรากฏขึ้น และเมื่อกด F5 อีกครั้งวงกลมสีแดงจะหายไป เรียกว่า การท็อกเกิล (Toggle) เบรกพอยท์ กด F5 อีกครั้งเพื่อสร้างวงกลมสีแดง เพียงจุดเดียวเท่านั้น
3. กด F8 บนคีย์บอร์ดเพื่อรันโปรแกรม โปรแกรมจะรันไปจนหยุดตรงประโยคที่มีวงกลมสีแดงนั้น โปรแกรมจะแสดงสัญลักษณ์สามเหลี่ยมสีเหลืองซ้อนทับกันอยู่ หลังจากนั้น กด F8 เพื่อดำเนินการต่อ
4. เลื่อนเคอร์เซอร์ไปยังประโยคที่มีวงกลมสีแดง กดปุ่ม F5 บนคีย์บอร์ดเพื่อปลดวงกลมสีแดงออก
5. เริ่มต้นใหม่เพื่อศึกษาการทำงานของ F4 (Run to cursor) โดยเลื่อนเคอร์เซอร์ไปวางบนประโยค ที่สนใจ กดปุ่ม F4 และสังเกตว่า สามเหลี่ยมสีเหลืองจะปรากฏหน้าประโยค เพื่อระบุว่าเครื่องรันมาถึงประโยคนี้แล้ว
6. กด F8 เพื่อรันต่อไป จนถึงสิ้นสุดการทำงานของโปรแกรม

E.3 การพัฒนาโดยใช้ประโยคคำสั่งที่ละขั้นตอน

ผู้อ่านควรเข้าใจคำสั่งพื้นฐานในการแปลโปรแกรมภาษาแอสเซมบลีที่สร้างขึ้นใน Code::Blocks ก่อนหน้านี้ ตามขั้นตอนต่อไปนี้

1. เปิดโปรแกรม Terminal หน้าต่างใหม่ แล้วย้ายโฟลเดอร์ไปยัง `/home/pi/c/Lab5` โดยใช้คำสั่ง `cd`
2. คอมไพล์ไฟล์ซอร์สโค้ดให้เป็นไฟล์อ็อบเจกต์ โดยเรียกใช้คอมไพเลอร์ชื่อ gcc ดังนี้


```
$ gcc -c main.c
```

ไฟล์ผลลัพธ์ ชื่อ `main.o` จะปรากฏขึ้น ผู้อ่านต้องตรวจสอบโดยใช้คำสั่ง `ls -la` เพื่อตรวจสอบวันที่และขนาดของไฟล์

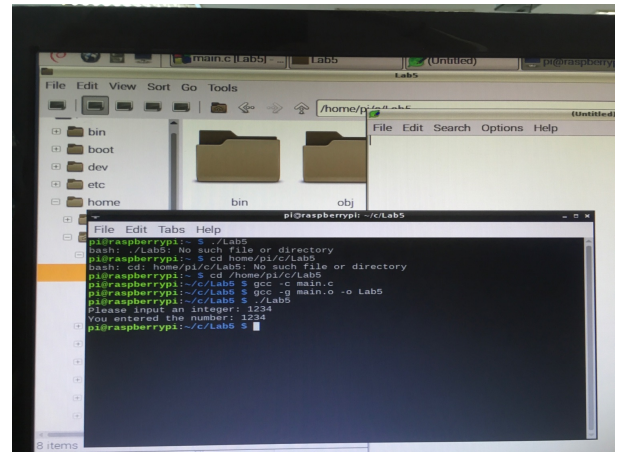
3. ทำการลิงค์และแปลงไฟล์อ็อบเจกต์เป็นไฟล์โปรแกรมโดย

```
$ gcc -g main.o -o Lab5
```

↳ debug กับ GDB

4. เรียกโปรแกรม Lab5 โดยพิมพ์

```
$ ./Lab5
```



5. เปรียบเทียบผลลัพธ์ที่ปรากฏขึ้นว่าตรงกับผลการรันใน IDE หรือไม่ อย่างไร

E.4 โครงสร้างของ Makefile

นอกเหนือจากการพัฒนาโปรแกรมด้วย IDE แล้ว การพัฒนาด้วย Makefile จะช่วยให้นักพัฒนามือสมัครเล่นและมืออาชีพดำเนินการได้ถูกต้องและรวดเร็ว ไฟล์ชื่อ Makefile เป็นไฟล์อักษรหรือเท็กซ์ไฟล์ (text file) ง่ายๆ ที่อธิบายความสัมพันธ์ระหว่าง ไฟล์ซอร์สโค้ดต่างๆ ไฟล์อ็อบเจกต์ และไฟล์โปรแกรม แต่ละบรรทัดจะมีโครงสร้างดังนี้

```
target : prerequisites ...
```

```
<tab>recipe
```

```
<tab>      ...
```

```
<tab>...
```

- target หมายถึง ชื่อไฟล์ที่จะถูกสร้างขึ้น โดยอาศัยไฟล์ต่างๆ จากส่วนที่เรียกว่า prerequisites นอกจากชื่อไฟล์แล้ว คำสั่งง่ายเช่น 'clean' สามารถใช้เป็น target ได้ ซึ่งนิยมใช้สำหรับลบไฟล์ต่างๆ ที่ไม่ต้องการ
- recipe หมายถึง คำสั่งหรือการกระทำที่จะใช้รายชื่อไฟล์ใน prerequisites นั้นมาสร้างไฟล์ target ได้สำเร็จ โดยแต่ละบรรทัดจะต้องเริ่มต้นด้วยปุ่ม tab เสมอ

E.5 การพัฒนาโดยใช้ Makefile

ตัวอย่างนี้เป็นกรสร้าง Makefile เพื่อคอมไพล์โปรแกรมเดิมที่เรามีอยู่ ผู้อ่านจะได้เข้าใจกลไกการทำงานที่ง่ายที่สุดก่อน หลังจากนั้นผู้อ่านสามารถศึกษาเพิ่มเติมด้วยตนเองได้จากเว็บไซต์หรือตัวอย่างโปรแกรม Open Source ที่ซับซ้อนขึ้นเรื่อยๆ ต่อไป

1. จงสร้าง makefile ในโฟลเดอร์ /home/pi/c/Lab5 โดยกรอกข้อความเหล่านี้ในไฟล์ใหม่

Lab5: main.o

```
gcc -g main.o -o Lab5
```

main.o: main.c

```
gcc -c main.c
```

clean :

```
rm *.o
```

- เมื่อกรอกเสร็จแล้ว ให้ทำการบันทึก หรือ save โดยตั้งชื่อไฟล์ว่า Makefile โดยไม่มีนามสกุล หลัง จากนั้น รันคำสั่งต่อไปนี้ใน Terminal ใน /home/pi/c/Lab5

- พิมพ์คำสั่งนี้ใน Terminal

```
$ make clean
```

เป็นการเรียกใช้คำสั่ง rm *.o ผ่านทาง Makefile เพื่อลบ (Remove) ไฟล์ที่มีนามสกุล .o ทั้งหมด

- พิมพ์คำสั่งนี้ใน Terminal

```
$ make Lab5
```

เป็นการเรียกใช้คำสั่ง gcc -c main.c และ gcc -g main.c -o Lab5 เพื่อสร้างไฟล์คำสั่ง Lab5 ที่จะทำงานตามซอร์สโค้ด main.c ที่กรอกไป โดยไฟล์ Lab5 ที่เกิดขึ้นใหม่จะมีโครงสร้างรูปแบบ ELF

- พิมพ์คำสั่งนี้ใน Terminal

```
$ ls -la
```

เพื่ออ่านค่าเวลาที่ไฟล์ Lab5 ที่เพิ่งถูกสร้าง โปรดสังเกตสีของชื่อไฟล์ต่างๆ ว่ามีสีอะไรบ้าง และบ่งบอกอะไรตามสีนั้นๆ

สีน้ำเงิน - Folder

สีเขียว - file ใดๆ

สีเทา - ไฟล์ปกติ (อะไรก็ได้)

- พิมพ์คำสั่งนี้ใน Terminal

```
$ ./Lab5
```

เป็นการเรียกใช้คำสั่ง Lab5 ให้ซีพียูปฏิบัติตาม

E.6 กิจกรรมท้ายการทดลอง

1. จงพัฒนาโปรแกรมภาษา C ให้สามารถอ่านไฟล์ Makefile เพื่อแสดงตัวอักษรในไฟล์ทีละตัวและค่ารหัส ASCII ฐานสิบหกของตัวอักษรนั้นบนหน้าจอ แล้วปิดไฟล์เมื่อเสร็จสิ้น
2. จงพัฒนาโปรแกรมภาษา C เพื่อสั่งพิมพ์เลขอนุกรม Fibonacci โดยรับค่าเลขเป้าหมาย n ซึ่งเกิดจาก $n = (n-1) + (n-2)$ และรายละเอียดเพิ่มเติมได้จาก [wikipedia](https://en.wikipedia.org/wiki/Fibonacci_sequence) ตัวอย่างต่อไปนี้ n=5 และพิมพ์ผลลัพธ์ดังนี้

1 1 2 3 5

1.

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

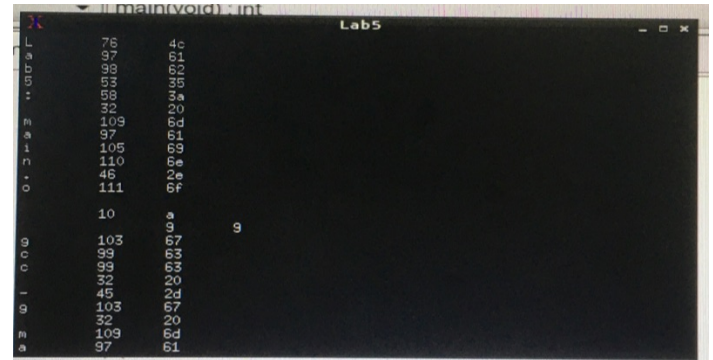
int main(void)
{
    FILE * fp;
    int c;

    fp = fopen ("Makefile", "r");

    while(1) {
        c = fgetc(fp);
        if( feof(fp) ) {
            break;
        }
        printf("%c \t %d \t %x\n", c, c, c);
    }

    fclose(fp);

    return 0;
}
```



2.

```
#include <stdio.h>

int fib(int);

int main() {
    int n = 0;
    printf("Enter number of Fibonacci : ");
    scanf("%d", &n);
    for (int i = 1; i <= n; i++) {
        printf("%d ", fib(i));
    }

    return 0;
}

int fib(int a) {
    if (a <= 2)
        return 1;
    else
        return fib(a-1) + fib(a-2);
}
```

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL 2: Code
Microsoft Windows [Version 10.0.18363.628]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\User>cd "c:\Users\User\Desktop\" && gcc HW1.c -o HW1 && "c:\Users\User\Desktop\HW1

c:\Users\User\Desktop>cd "c:\Users\User\Desktop\" && gcc HW2.c -o HW2 && "c:\Users\User\Desktop\HW2
c:/mingw/bin/../lib/gcc/mingw32/6.3.0/../../../../libmingw32.a(main.o):(.text.startup+0xa0): undefined reference to 'WinMain@16'
collect2.exe: error: ld returned 1 exit status

c:\Users\User\Desktop>cd "c:\Users\User\Desktop\" && gcc HW2.c -o HW2 && "c:\Users\User\Desktop\HW2
Enter number of Fibonacci : 5
1 1 2 3 5
```