Lab 03

L0301 ให้นักศึกษาเขียนโปรแกรมเพื่อทดลองการคำนวณทางคณิตศาสตร์ด้วย Operator พื้นฐานกับข้อมูลชนิดต่าง ๆ โปรแกรมที่นักศึกษาจะเขียนประกอบด้วยตัวแปร 6 ตัว ได้แก่ n1, n2, f1, f2, r1, และ r2 กำหนดให้ n1, n2, f1, และ f2 เป็นตัวแปรที่รับค่ามาจากผู้ใช้ผ่านคีย์บอร์ด, r1 เป็นตัวแปรเก็บผลลัพธ์ระหว่าง n1 และ n2, และ r2 เก็บผลการคำนวณระหว่าง f1 และ f2

กำหนดชนิดข้อมูลของ n1 และ n2 เป็นตัวแปรสำหรับเก็บข้อมูลเลขจำนวนเต็ม ส่วน f1, f2, r1, และ r2 เป็นตัวแปรเก็บข้อมูลเลขทศนิยม

กำหนดการแสดงผลของเลขทศนิยมทั้งหมดในโปรแกรมนี้ แสดงในรูปแบบเลขทศนิยม 2 ตำแหน่ง

ตัวอย่างผล

```
Enter n1: 7
Enter n2: 3
Enter f1: 6.3
Enter f2: 1.5

Results
7 + 3 = 10.00
7 - 3 = 4.00
7 * 3 = 21.00
7 / 3 = 2.00
7 % 3 = 1.00

6.30 + 1.50 = 7.80
6.30 - 1.50 = 4.80
6.30 * 1.50 = 9.45
6.30 / 1.50 = 4.20
```

Code ที่ใช้ พร้อมผลของตนเอง

Enter n1: 4
Enter n2: 5
Enter f1: 6
Enter f2: 7
Results
4 + 5 = 9.00
4 - 5 = -1.00
4 * 5 = 20.00
4 / 5 = 0.00
4 % 5 = 4.00
6.00 + 7.00 = 13.00
6.00 - 7.00 = -1.00
6.00 * 7.00 = 42.00
6.00 / 7.00 = 0.86

```c
#include <stdio.h>

int main(void) {
    int n1, n2;
    float f1, f2;
    float r1, r2;

    printf("Enter n1: ");
    scanf("%d", &n1);
    printf("Enter n2: ");
    scanf("%d", &n2);
    printf("Enter f1: ");
    scanf("%f", &f1);
    printf("Enter f2: ");
    scanf("%f", &f2);

    printf("\nResults\n");

    r1 = n1 + n2;
    printf("%d + %d = %.2f\n", n1, n2, r1);

    r1 = n1 - n2;
    printf("%d - %d = %.2f\n", n1, n2, r1);

    r1 = n1 * n2;
    printf("%d * %d = %.2f\n", n1, n2, r1);

    r1 = n1 / n2;               /* int หาร int -> ปัดเศษทิ้ง */
    printf("%d / %d = %.2f\n", n1, n2, r1);

    r1 = n1 % n2;
    printf("%d %% %d = %.2f\n", n1, n2, r1);

    r2 = f1 + f2;
    printf("%.2f + %.2f = %.2f\n", f1, f2, r2);

    r2 = f1 - f2;
    printf("%.2f - %.2f = %.2f\n", f1, f2, r2);

    r2 = f1 * f2;
    printf("%.2f * %.2f = %.2f\n", f1, f2, r2);

    r2 = f1 / f2;
    printf("%.2f / %.2f = %.2f\n", f1, f2, r2);

    return 0;
}
```

L0302 ให้นักศึกษาเขียนโปรแกรมใหม่ โดยดัดแปลงจากข้อ L0301 โดยตัวแปรในข้อ L0302 เหลือเพียงแค่ n1, n2, f1, และ r1 กำหนดให้ r1 เป็นตัวแปรเก็บผลคำนวณระหว่าง ตัวแปร 2 ตัว (n1 กับ n2 และ n1 กับ f1) การแสดงผลของเลขทศนิยมทั้งหมดในโปรแกรมนี้ แสดงในรูปแบบเลขทศนิยม 2 ตำแหน่ง

ตัวอย่างผล

```
Enter n1: 7
Enter n2: 3
Enter f1: 2.5

Results
7 / 3 = 2.00
7 / 2.50 = 2.80
```

Code ที่ใช้ พร้อมผลของตนเอง

Enter n1: 4
Enter n2: 5
Enter f1: 6
Results
4 / 5 = 0.00
4 / 6.00 = 0.67

```c
#include <stdio.h>

int main(void) {
    int n1, n2;
    float f1;
    float r1;

    printf("Enter n1: ");
    scanf("%d", &n1);
    printf("Enter n2: ");
    scanf("%d", &n2);
    printf("Enter f1: ");
    scanf("%f", &f1);

    printf("\nResults\n");

    r1 = n1 / n2;                /* int / int -> เป็นจำนวนเต็ม เศษถูกตัดทิ้ง */
    printf("%d / %d = %.2f\n", n1, n2, r1);

    r1 = n1 / f1;                /* int / float -> คอมไพเลอร์เลื่อนขั้น n1 เป็น float ก่อน
คำนวณ */
    printf("%d / %.2f = %.2f\n", n1, f1, r1);

    return 0;
}
```

ตอบคำถามข้อ L0302

- **ทำอย่างจึงจะทำให้ n1 และ n2 เกิดการคำนวณแบบเลขทศนิยม จนได้ผลที่เป็นเลขทศนิยมอย่างถูกต้อง (เหมือน n1 และ f1)**

ต้อง **cast ตัวแปรใดตัวหนึ่ง (n1 หรือ n2) ให้เป็นชนิดข้อมูลทศนิยม (float) ก่อนทำการหาร** เช่น c

r1 = (float) n1 / n2;

เหตุผล: ปกติเมื่อ int หารกับ int (n1 / n2) ภาษา C จะทำ **integer division** คือคำนวณแบบจำนวนเต็มล้วน ๆ แล้วตัดเศษทศนิยมทิ้งไปเลย (เช่น 7/3 ได้ 2 ไม่ใช่ 2.33) แม้จะเอาผลลัพธ์ไปเก็บในตัวแปร float (r1) ก็ตาม เพราะการปัดเศษทิ้งเกิดขึ้น "ตอนคำนวณ" ก่อนที่จะ assign ค่าไปเก็บแล้ว

ในขณะที่ n1/f1 ได้ผลถูกต้องเพราะ f1 เป็น float อยู่แล้ว เมื่อ int หารกับ float ภาษา C จะทำ **implicit type promotion** คือเลื่อนชั้นตัวแปร int ให้กลายเป็น float ก่อนโดยอัตโนมัติ แล้วจึงคำนวณแบบทศนิยมทั้งนิพจน์

ดังนั้นถ้าอยากให้ n1/n2 ได้ผลทศนิยมแบบถูกต้องเหมือนกัน ต้อง **บังคับ (cast) ให้อย่างน้อยตัวใดตัวหนึ่งกลายเป็น float ก่อน** ตัวหารทั้งนิพจน์จะถูกเลื่อนชั้นเป็นทศนิยมไปด้วย เช่น (float)n1 / n2 หรือ n1 / (float)n2 ก็ได้ผลเหมือนกัน

L0303 ให้นักศึกษาเขียนโปรแกรมคำนวณสมการ 3 สมการ จากนั้นแสดงวิธีคิดของสมการแต่ละข้อจนได้ผลลัพธ์ สมการทั้ง 3 เป็นดังนี้

- **x = 3 + 4 * n / 2 – 18**
- **y = (3 + 4) * n / (18 * 2)**
- **z = 15 / 2 + 3 – (14 * n)**

กำหนดให้ n เป็นตัวแปรเลขจำนวนเต็มที่รับข้อมูลจากผู้ใช้โปรแกรม

ตัวอย่างผล

```
Enter n: 30

Result x = 45
Result y = 5
Result z = -410
```

ผลที่ได้ พร้อมแสดงวิธีทำของแต่ละสมการในแต่ละขั้นตอน

x = 3 + 4 * n / 2 - 18

= 3 + 16 / 2 - 18 (4 * 4)

= 3 + 8 - 18 (16 / 2)

= 11 - 18 (3 + 8)

Result x = -7

y = (3 + 4) * n / (18 * 2)

= 7 * 4 / 36 ((3+4)=7, (18*2)=36)

= 28 / 36

Result y = 0

z = 15 / 2 + 3 - (14 * n)

= 7 + 3 - 56 (15/2=7, 14*4=56)

= 10 - 56

Result z = -46

```c
#include <stdio.h>

int main(void) {
    int n;
    int x, y, z;
    int step1, step2, step3;

    printf("Enter n: ");
    scanf("%d", &n);
    printf("\n");

    /* ---------- x = 3 + 4 * n / 2 - 18 ---------- */
    printf("x = 3 + 4 * n / 2 - 18\n");
    step1 = 4 * n;
    printf("  = 3 + %d / 2 - 18        (4 * %d)\n", step1, n);
    step2 = step1 / 2;
    printf("  = 3 + %d - 18           (%d / 2)\n", step2, step1);
    step3 = 3 + step2;
    printf("  = %d - 18               (3 + %d)\n", step3, step2);
    x = step3 - 18;
    printf("Result x = %d\n\n", x);

    /* ---------- y = (3 + 4) * n / (18 * 2) ---------- */
    printf("y = (3 + 4) * n / (18 * 2)\n");
    int a = 3 + 4;
    int b = 18 * 2;
    printf("  = %d * %d / %d          ((3+4)=%d, (18*2)=%d)\n", a, n, b, a, b);
    int c = a * n;
    printf("  = %d / %d\n", c, b);
    y = c / b;
    printf("Result y = %d\n\n", y);

    /* ---------- z = 15 / 2 + 3 - (14 * n) ---------- */
    printf("z = 15 / 2 + 3 - (14 * n)\n");
    int d = 15 / 2;
    int f = 14 * n;
    printf("  = %d + 3 - %d          (15/2=%d, 14*%d=%d)\n", d, f, d, n, f);
    int e = d + 3;
    printf("  = %d - %d\n", e, f);
    z = e - f;
    printf("Result z = %d\n", z);

    return 0;
}
```

L0304 ให้นักศึกษาเขียนโปรแกรมคำนวณสมการ 4 สมการ จากนั้นให้แปลงสมการต่อไปนี้ให้อยู่ในรูปที่ใช้ Shorthand Operator โดยให้ผลเหมือนกับสมการเดิมก่อนแปลง ถ้าสมการใดแปลงไม่ได้ ให้เขียน Comment กำกับไว้ว่าทำไม่ได้

- x = x – 3/2
- x = 3x + 4x – x
- x = 12 * 3 + x
- x = 14 * (7 * 2 – 4) / 3 * x

กำหนดให้ x เป็นตัวแปรเลขจำนวนเต็มที่รับค่ามาจากผู้ใช้งาน

ตัวอย่างผลลัพธ์

```
Enter x: 7

x after eq1 = 6
x after eq2 = 36
x after eq3 = 72
x after eq4 = 3312
```

Code ที่ใช้

```c
#include <stdio.h>

int main(void) {
    int x;

    printf("Enter x: ");
    scanf("%d", &x);

    /* eq1: x = x - 3/2  ->  x อยู่ฝั่งซ้ายตัวเดียว แปลงได้ตรง ๆ */
    x -= 3 / 2;
    printf("x after eq1 = %d\n", x);

    /* eq2: x = 3x + 4x - x = (3+4-1)*x = 6x
       ต้องยุบพจน์ x ทั้งสามให้เหลือค่าสัมประสิทธิ์เดียวก่อน จึงจะใช้ shorthand ได้ */
    x *= (3 + 4 - 1);
    printf("x after eq2 = %d\n", x);

    /* eq3: x = 12*3 + x  ->  การบวกสลับที่ได้ (x + 12*3) จึงแปลงเป็น += ได้ */
    x += 12 * 3;
    printf("x after eq3 = %d\n", x);

    /* eq4: x = 14*(7*2-4)/3 * x  ->  การคูณสลับที่ได้ (x * ผลลัพธ์คงที่) จึงแปลงเป็น *= ได้ */
    x *= 14 * (7 * 2 - 4) / 3;
    printf("x after eq4 = %d\n", x);

    return 0;
}
```

L0305 ให้นักศึกษาเขียนโปรแกรมเขียนโปรแกรมคำนวณสมการ 3 สมการคล้ายกับข้อ L0303 โดยสมการแต่ละสมการมีการเปลี่ยนแปลงดังนี้

- **x = 3 + 4 * ++n / 2 – 18**
- **y = (3 + 4) * n++ / (18 * 2)**
- **z = 15 / 2 + 3 – (14 * --n)**

กำหนดให้ n เป็นตัวแปรเลขจำนวนเต็มที่รับข้อมูลจากผู้ใช้โปรแกรม

ตัวอย่างผลลัพธ์

```
Enter n: 30

Result x = 47
Result y = 6
Result z = -424
```

ผลที่ได้ พร้อมแสดงวิธีทำของแต่ละสมการในแต่ละขั้นตอน

Enter n: 4

x = 3 + 4 * ++n / 2 - 18

++n -> n = 5

= 3 + 20 / 2 - 18 = 3 + 10 - 18 = 13 - 18

Result x = -5 (n = 5)

y = (3 + 4) * n++ / (18 * 2)

n++ -> n = 5 n = 6

= 7 * 5 / 36 = 35 / 36

Result y = 0

z = 15 / 2 + 3 - (14 * --n)

--n -> n = 5

= 7 + 3 - 70 = 10 - 70

Result z = -60

```c
#include <stdio.h>

int main(void) {
    int n;
    int x, y, z;

    printf("Enter n: ");
    scanf("%d", &n);
    printf("\n");

    /* ---------- x = 3 + 4 * ++n / 2 - 18 ---------- */
    printf("x = 3 + 4 * ++n / 2 - 18\n");
    int n1 = ++n;                           /* pre-increment: เพิ่มค่า n ก่อนนำไปใช้ */
    printf("  ++n -> n = %d\n", n1);
    int step1 = 4 * n1;
    int step2 = step1 / 2;
    int step3 = 3 + step2;
    x = step3 - 18;
    printf("  = 3 + %d / 2 - 18 = 3 + %d - 18 = %d - 18\n", step1, step2, step3);
    printf("Result x = %d   (n = %d)\n\n", x, n);

    /* ---------- y = (3 + 4) * n++ / (18 * 2) ---------- */
    printf("y = (3 + 4) * n++ / (18 * 2)\n");
    int n_used = n;                         /* post-increment: ใช้ค่าเดิมของ n ก่อน */
    n++;                                    /* แล้วค่อยเพิ่ม n ทีหลัง */
    printf("  n++ -> ใช้ n = %d ก่อน แล้วค่อยเพิ่มเป็น n = %d\n", n_used, n);
    int a = 3 + 4;
    int b = 18 * 2;
    int c = a * n_used;
    y = c / b;
    printf("  = %d * %d / %d = %d / %d\n", a, n_used, b, c, b);
    printf("Result y = %d\n\n", y);

    /* ---------- z = 15 / 2 + 3 - (14 * --n) ---------- */
    printf("z = 15 / 2 + 3 - (14 * --n)\n");
    int n2 = --n;                           /* pre-decrement: ลดค่า n ก่อนนำไปใช้ */
    printf("  --n -> n = %d\n", n2);
    int d = 15 / 2;
    int e = d + 3;
    int f = 14 * n2;
    z = e - f;
    printf("  = %d + 3 - %d = %d - %d\n", d, f, e, f);
    printf("Result z = %d\n", z);

    return 0;
}
```

L0306 ให้นักศึกษาเขียนโปรแกรมคำนวณพื้นที่วงกลม ปริมาตรทรงกลม และปริมาตรทรงกระบอกจากค่ารัศมีและค่าความสูงที่ผู้ใช้เป็นคนป้อนสู่โปรแกรม ให้ใช้คำสั่งสำเร็จรูปทางคณิตศาสตร์ในการคำนวณ กำหนดค่า pi มีค่า 3.1416

สูตรคำนวณ

- พื้นที่วงกลม = pi * r^2
- ปริมาตรทรงกลม = 4/3 * pi * r^3
- ปริมาตรทรงกระบอก = pi * r^2 * h

ตัวอย่างผลลัพธ์

```
Enter r: 3.5
Enter h: 12

Circle area = 38.48

Sphere volume = 179.59

Cylinder volume = 461.82
```

Code ที่ใช้

```c
#include <stdio.h>
#include <math.h>

int main(void) {
    float r, h;
    const float pi = 3.1416f;
    float circleArea, sphereVolume, cylinderVolume;

    printf("Enter r: ");
    scanf("%f", &r);
    printf("Enter h: ");
    scanf("%f", &h);

    circleArea     = pi * pow(r, 2);
    sphereVolume   = (4.0 / 3.0) * pi * pow(r, 3);
    cylinderVolume = pi * pow(r, 2) * h;

    printf("\nCircle area = %.2f\n", circleArea);
    printf("Sphere volume = %.2f\n", sphereVolume);
    printf("Cylinder volume = %.2f\n", cylinderVolume);

    return 0;
}
```

L0307 ให้นักศึกษาเขียนโปรแกรมคำนวณค่า RMS ของข้อมูลจำนวน 4 ค่า ตามสูตรที่กำหนดให้

สูตรคำนวณ RMS

$$X_{rms} = \sqrt{\frac{x_1^2 + x_2^2 + x_3^2 \ldots + x_n^2}{n}}$$

เมื่อ x_n คือข้อมูลแต่ละค่าตั้งแต่ค่าที่ 1 ไปถึงค่าที่ n และ n คือจำนวนของข้อมูลที่มีในการคำนวณค่า x_{rms} กำหนดให้ใช้คำสั่งสำเร็จรูปทางคณิตศาสตร์ในการคำนวณ

ตัวอย่างผลลัพธ์

```
Enter x1: 3
Enter x2: 4
Enter x3: 5
Enter x4: 6

xrms = 4.637
```

Code ที่ใช้

```c
#include <stdio.h>
#include <math.h>

int main(void) {
    float x1, x2, x3, x4;
    float xrms;
    int n = 4;

    printf("Enter x1: ");
    scanf("%f", &x1);
    printf("Enter x2: ");
    scanf("%f", &x2);
    printf("Enter x3: ");
    scanf("%f", &x3);
    printf("Enter x4: ");
    scanf("%f", &x4);

    xrms = sqrt((pow(x1, 2) + pow(x2, 2) + pow(x3, 2) + pow(x4, 2)) / n);

    printf("\nxrms = %.5f\n", xrms);

    return 0;
}
```

L0308 ให้นักศึกษาเขียนโปรแกรมคำนวณพื้นที่วงกลม จากค่ารัศมีที่ผู้ใช้เป็นคนป้อนสู่โปรแกรม ให้ใช้คำสั่งสำเร็จรูปทางคณิตศาสตร์ในการคำนวณ กำหนดค่า pi มีค่า 3.1416

สูตรคำนวณ

- พื้นที่วงกลม = pi * r^2

จากนั้นให้นักศึกษาตรวจสอบว่าพื้นที่ของวงกลมมีค่าน้อยกว่า 100 หรือไม่ ถ้าพื้นที่วงกลมน้อยกว่า 100 แสดงผล Big circle 0 ถ้าไม่ใช่ แสดงผล Big circle 1 ตามตัวอย่าง **(ห้ามใช้ if-else)**

ตัวอย่างผลลัพธ์

```
Enter r: 2

Circle area = 12.57
Big circle 0
```

```
Enter r: 7

Circle area = 153.94
Big circle 1
```

Code ที่ใช้

```c
#include <stdio.h>
#include <math.h>

int main(void) {
    float r;
    const float pi = 3.1416f;
    float circleArea;
    int bigCircle;

    printf("Enter r: ");
    scanf("%f", &r);

    circleArea = pi * pow(r, 2);
    printf("\nCircle area = %.2f\n", circleArea);

    /* ห้ามใช้ if-else: นิพจน์เปรียบเทียบ (circleArea >= 100) ให้ผลเป็น 1 (true) หรือ 0 (false)
       อยู่แล้ว จึงพิมพ์ค่านั้นออกมาโดยตรงแทนการเช็คเงื่อนไขด้วย if-else */
    bigCircle = (circleArea >= 100);
    printf("Big circle %d\n", bigCircle);

    return 0;
}
```