Chapter 6 – Making decisions

- ในบทที่แล้วเราได้ศึกษาเกี่ยวกับความสามารถของโปรแกรมในการทำซ้ำคำสั่ง
- บทนี้จะกล่าวเกี่ยวกับการเลือกทำคำสั่งหรือการตัดสินใจ ซึ่งการเลือกว่าจะหยุดลูปหรือไม่ก็เป็นการตัดสินใจอย่างหนึ่ง
- คำสั่งที่เกี่ยวข้องกับการตัดสินใจได้แก่ if และ switch
- เงื่อนไขที่ใช้ตัดสินใจจะเกี่ยวข้องกับ relational operators เช่น <, >, <=, >=, ==, !=
- นอกจากนี้การตัดสินใจยังมีเรื่องของ Boolean และ conditional operator เข้ามาเกี่ยวข้อง

6.1 คำสั่ง if

- รูปแบบการใช้งานคำสั่ง (เลือกที่จะทำหรือไม่ทำคำสั่ง)

```
if(condition)
{
     statement_1;
     statement_2;
     ...
}
```

- ตัวอย่าง

```
if( count > COUNT_LIMIT )
     printf ("Count limit exceeded\n");
```

- อีกตัวอย่างคือในกรณีที่เราต้องการหาค่าสัมบูรณ์ของจำนวนเต็ม (โดยไม่ใช้ฟังก์ชันทางคณิตศาสตร์) ในการนี้เราจะต้องคูณด้วยลบหนึ่ง ถ้าจำนวนเต็มนั้นมีค่าน้อยกว่า 0 เขียนได้ดังโปรแกรมที่ 6.1

Program 6.1 Calculating the Absolute Value of an Integer

```
// Program to calculate the absolute value of an integer
#include <stdio.h>
int main (void)
{
    int number;
    printf("Type in your number: ");
    scanf("%i", &number);
    if( number < 0 )
        number = -number;
    printf ("The absolute value is %i\n", number);
    return 0;
}
```

```
Type in your number: -100
The absolute value is 100
```

```
Type in your number: 2000
The absolute value is 2000
```

- สังเกตว่าเรารันโปรแกรมสองครั้งเพื่อทดสอบความถูกต้องของโปรแกรมในสองสถานการณ์ที่แตกต่างกัน
- โปรแกรมที่ 6.2 ใช้หาค่าเฉลี่ยของคะแนน และนับจำนวนคะแนนที่ตก ถ้าคะแนนที่ได้ต่ำกว่า 65

Program 6.2 Calculating the Average of a Set of Grades and Counting the Number of Failing Test Grades

```
/* Program to calculate the average of a set of grades and count
the number of failing test grades */
#include <stdio.h>
int main (void)
{
    int numberOfGrades, i, grade;
    int gradeTotal = 0;
    int failureCount = 0;
    float average;
    printf("How many grades will you be entering? ");
    scanf("%i", &numberOfGrades);
    for( i = 1; i <= numberOfGrades; ++i )
    {
        printf ("Enter grade #%i: ", i);
        scanf ("%i", &grade);
        gradeTotal = gradeTotal + grade;
        if( grade < 65 )
            ++failureCount;
    }
    average = (float) gradeTotal / numberOfGrades;
    printf("\nGrade average = %.2f\n", average);
    printf("Number of failures = %i\n", failureCount);
    return 0;
}
```

```
How many grades will you be entering? 7
Enter grade #1: 93
Enter grade #2: 63
Enter grade #3: 87
Enter grade #4: 65
Enter grade #5: 62
Enter grade #6: 88
Enter grade #7: 76
Grade average = 76.29
Number of failures = 2
```

- สังเกตการคำนวณค่าเฉลี่ย โดยใช้ type cast operator (float)
- สังเกตการใช้ precision modifier %.2f

6.2 คำสั่ง if-else

- รูปแบบการใช้งานคำสั่ง (เลือกทำอย่างใดอย่างหนึ่ง)

```
if(condition)
{
     statement_a1;
     statement_a2;
     ...
}
else
{
     statement_b1;
     statement_b2;
     ...
}
```

- ตัวอย่าง เขียนโปรแกมเพื่อใช้บอกว่าจำนวนเต็มที่ใส่โดยผู้ใช้เป็นจำนวนคู่หรือคี่ ถ้า (if)จำนวนเต็มสามารถหารลงตัวโดย 2 จำนวนนั้นเป็นจำนวนคู่ นอกจากนั้น (else) เป็นจำนวนคี่

Program 6.3 Determining if a Number Is Even or Odd (using *if*)

```
// Program to determine if a number is even or odd
#include <stdio.h>
int main (void)
{
     int number_to_test, remainder;
     printf ("Enter your number to be tested.: ");
     scanf ("%i", &number_to_test);
     remainder = number_to_test % 2;
     if( remainder == 0 )
          printf ("The number is even.\n");
     if( remainder != 0 )
          printf ("The number is odd.\n");
     return 0;
}
```

```
Enter your number to be tested: 2455
The number is odd.
```

```
Enter your number to be tested: 1210
The number is even.
```

Program 6.4 Revising the Program to Determine if a Number Is Even or Odd

```
// Program to determine if a number is even or odd (Ver. 2)
#include <stdio.h>
int main ()
{
    int number_to_test, remainder;
    printf("Enter your number to be tested: ");
    scanf("%i", &number_to_test);
    remainder = number_to_test % 2;
    if( remainder == 0 )
        printf("The number is even.\n");
    else
        printf("The number is odd.\n");
    return 0;
}
```

6.3 การตัดสินใจที่มีเงื่อนไขซับซ้อนขึ้น (Compound relational tests)

- จากตัวอย่างที่ผ่านมา เงื่อนไขในคำสั่ง if จะไม่ซับซ้อน หากเราต้องการใช้เงื่อนไขที่ซับซ้อนขึ้นก็ทำได้เช่นกัน ยกตัวอย่างเช่นในโปรแกมที่ 6.2 การคิดคะแนนสอบ หากเราต้องการนับจำนวนของคนที่ได้คะแนนอยู่ในช่วงระหว่าง 70 ถึง 79 ก็ทำได้เช่นกัน
- โดยภาษา C อนุญาตให้มีการรวมหลายๆเงื่อนไขเข้าด้วยกัน ผ่าน AND หรือ OR operator ซึ่งเขียนแทนได้ด้วยเครื่องหมาย && หรือ || ตัวอย่างการใช้งาน

```
if(grade >=70 && grade <=79)
    ++grade_C;
```

```
if(grade < 0 || grade > 100)
    printf("Error - index out of range\n");
```

- ในตัวอย่างถัดไปเราจะใช้เงื่อนไขที่ซับซ้อนเหล่านี้ในการตรวจสอบว่าปี เป็น leap year หรือไม่ โดย leap year คือ ปีที่หารด้วย 4 ลงตัว ส่วนปีที่หารด้วย 100 ลงตัวนั้นไม่ใช่ leap year ยกเว้นปีที่หารด้วย 400 ลงตัว
- อัลกอรึทึมคือ เราจะหาเศษจากการหารค่าปีด้วย 4, 100, และ 400 แล้วเก็บไว้ในตัวแปร จากนั้นนำมาตรวจสอบว่าตรงตามเงื่อนไขหรือไม่
- สามารถเขียนเป็นเงื่อนไขได้ดังนี้

```
if ( (rem_4 == 0 && rem_100 != 0) || rem_400 == 0 )
    printf ("It's a leap year.\n");
```

Program 6.5 Determining if a Year Is a Leap Year

```
// Program to determines if a year is a leap year
#include <stdio.h>
int main(void)
{
    int year, rem_4, rem_100, rem_400;
    printf("Enter the year to be tested: ");
    scanf("%i", &year);
    rem_4 = year % 4;
    rem_100 = year % 100;
    rem_400 = year % 400;
```

```
    if( (rem_4 == 0 && rem_100 != 0) || rem_400 == 0 )
        printf ("It's a leap year.\n");
    else
        printf ("Nope, it's not a leap year.\n");
    return 0;
}
```

```
Enter the year to be tested: 1955
Nope, it's not a leap year.
```

```
Enter the year to be tested: 2000
It's a leap year.
```

```
Enter the year to be tested: 1800
Nope, it's not a leap year.
```

- ภาษา C อนุญาตให้มีการคำนวณในบรรทัดเดียวกันได้ ซึ่งทำให้เงื่อนไขข้างต้นสามารถเขียนได้ดังนี้

```
if ( ( year % 4 == 0 && year % 100 != 0 ) || year % 400 == 0 )
```

6.4 การใช้ if ซ้อนกัน

- if สามารถใช้ซ้อนกันได้ดังตัวอย่าง

```
if( gameIsOver == 0 )
    if( playerToMove == YOU)
        printf("Your Move\n");
```

- ซึ่งมีความหมายเดียวกับ

```
if ( gameIsOver == 0 && playerToMove == YOU )
    printf ("Your Move\n");
```

- เราสามารถใช้ if ซ้อนกับ if-else ได้ เช่นกัน

```
if ( gameIsOver == 0 )
    if ( playerToMove == YOU )
        printf ("Your Move\n");
    else
        printf ("My Move\n");
```

- สังเกตว่า else ในคำสั่ง จะจับคู่กับ if ตัวสุดท้ายที่ไม่มีคู่ else อยู่

```
if ( gameIsOver == 0 )
    if ( playerToMove == YOU )
        printf ("Your Move\n");
    else
        printf ("My Move\n");
else
    printf ("The game is over\n");
```

6.5 การใช้ if-else if

- การใช้ if-else ที่ผ่านมานั้น เป็นการเลือกทำระหว่างสองกรณี เช่นเลขคู่หรือเลขคี่ leap year หรือไม่ใช่ leap year
- แต่ในความเป็นจริงเราอาจต้องการให้มีการเลือกทำระหว่างสามกรณีขึ้นไป เช่น ให้แสดงผลเป็น -1 ถ้าค่าที่ใส่ให้มีค่าน้อยกว่า 0 แสดงผลเป็น 0 ถ้าค่าเป็น 0 และแสดงผลเป็น 1 ถ้าค่ามากกว่า 0 (sign function)

- ในการนี้เราจะใช้ if-else if

```
if(condition_a)
{
    statement_a1;
    statement_a2;
    ...
}
else if(condition_b)
{
    statement_b1;
    statement_b2;
    ...
}
else
{
    statement_c1;
    statement_c2;
    ...
}
```

- ในกรณีที่ทั้งเงื่อนไข condition_a และ condition_b เป็นจริงทั้งคู่ โปรแกรมจะทำเฉพาะ statement ใน condition_a และออกจากคำสั่ง if

Program 6.6 Implementing the Sign Function

```
// Program to implement the sign function
#include <stdio.h>
int main (void)
{
    int number, sign;
    printf("Please type in a number: ");
    scanf("%i", &number);
    if( number < 0 )
        sign = -1;
    else if( number == 0 )
        sign = 0;
    else // Must be positive
        sign = 1;
    printf ("Sign = %i\n", sign);
    return 0;
}
```

```
Please type in a number: 1121
Sign = 1
```

```
Please type in a number: -158
Sign = -1
```

```
Please type in a number: 0
Sign = 0
```

Program 6.7 Categorizing a Single Character Entered at the Terminal

```
/* Program to categorize a single character that is entered at
the terminal */
#include <stdio.h>
int main (void)
{
    char c;
    printf ("Enter a single character:\n");
    scanf("%c", &c);
    if( (c >= 'a' && c <= 'z') || (c >= 'A' && c <= 'Z') )
        printf ("It's an alphabetic character.\n");
    else if ( c >= '0' && c <= '9' )
        printf ("It's a digit.\n");
    else
        printf ("It's a special character.\n");
    return 0;
}
```

```
Enter a single character:
8
It's a digit.
```

```
Enter a single character:
B
It's an alphabetic character.
```

```
Enter a single character:
&
It's a special character.
```

Program 6.8 Evaluating Simple Expressions

```
/* Program to evaluate simple expressions of the form
number operator number */
#include <stdio.h>
int main (void)
{
    float value1, value2;
    char operator;
    printf("Type in your expression.\n");
    scanf("%f %c %f", &value1, &operator, &value2);
    if( operator == '+' )
        printf("%.2f\n", value1 + value2);
    else if( operator == '-' )
        printf("%.2f\n", value1 - value2);
    else if( operator == '*' )
        printf("%.2f\n", value1 * value2);
    else if( operator == '/' )
        printf("%.2f\n", value1 / value2);
    return 0;
}
```

```
Type in your expression.
123.5 + 59.3
182.80
```

```
Type in your expression.
198.7 / 26
7.64
```

```
Type in your expression.
89.3 * 2.5
223.25
```

- โปรแกรมที่ 6.8 ยังมีจุดที่ไม่สมบูรณ์ คือเมื่อผู้ใช้ใส่เครื่องหมายนอกเหนือจาก +,-,*,/ โปรแกรมจะไม่เข้าเงื่อนไข if else อันไหนเลยและจะจบโปรแกรมโดยไม่บอกผู้ใช้ทราบ นอกจากนี้หากผู้ใช้ ใส่ตัวหารเป็น 0 โปรแกรมจะ error
- โปรแกรมต่อไปจะแก้ไขข้อบกพร่องเหล่านี้

Program 6.8 A Revising the Program to Evaluate Simple Expressions

```
/* Program to evaluate simple expressions of the form
value operator value */
#include <stdio.h>
int main (void)
{
    float value1, value2;
    char operator;
    printf("Type in your expression.\n");
    scanf("%f %c %f", &value1, &operator, &value2);
    if( operator == '+' )
        printf ("%.2f\n", value1 + value2);
    else if( operator == '-' )
        printf ("%.2f\n", value1 - value2);
    else if( operator == '*' )
        printf ("%.2f\n", value1 * value2);
    else if( operator == '/' )
        if( value2 == 0 )
            printf("Division by zero.\n");
        else
            printf("%.2f\n", value1 / value2);
    else
        printf ("Unknown operator.\n");
    return 0;
}
```

```
Type in your expression.
123.5 + 59.3
182.80
```

```
Type in your expression.
198.7 / 0
Division by zero.
```

```
Type in your expression.
125 $ 28
Unknown operator.
```

6.6 คำสั่ง switch

- นอกเหนือจากการใช้ if-else if เพื่อจำแนกเงื่อนไขออกเป็นหลายกรณีแล้ว ภาษา C ยังมีคำสั่งเพิ่มเติมที่ใช้ได้เช่นเดียวกันคือ switch

```
switch ( variable )
{
    case value1:
        program statement
        program statement
        ...
        break;
    case value2:
        program statement
        program statement
        ...
        break;
    ...
    case valuen:
        program statement
        program statement
        ...
        break;
    default:
        program statement
        program statement
        ...
        break;
}
```

- คำสั่งจะเปรียบเทียบค่าใน variable กับค่าของ value1, value2, … , valuen ซึ่งอาจเป็นค่าคงที่หรือตัวแปร หากเท่ากับกรณีใดแล้ว คำสั่งในกรณีนั้นจะถูกกระทำ สังเกตว่าในแต่ละกรณีมีหลายคำสั่งได้โดยไม่ต้องใช้เครื่องหมายวงเล็บคร่อมเหมือนคำสั่งอื่นๆ

- คำสั่ง break ใช้เพื่อออกจาก switch หากไม่ใส่ จะทำให้โปรแกรมทำการตรวจสอบกรณีถัดไปอีก
- default คือกรณีที่ variable ไม่เข้าเงื่อนไขใดๆ เลย ซึ่งคล้ายกับ else ในกรณีของ if-else
- เราสามารถเขียนคำสั่ง if-else if แทน switch ได้

```
if( expression == value1 )
{
    program statement
    program statement
    ...
}
else if( expression == value2 )
{
    program statement
    program statement
    ...
}
...
else if( expression == valuen )
{
    program statement
    program statement
    ...
}
else
{
    program statement
    program statement
    ...
}
```

Program 6.9 Revising the Program to Evaluate Simple Expressions

```
/* Program to evaluate simple expressions of the form
value operator value */
#include <stdio.h>
int main(void)
{
    float value1, value2;
    char operator;
    printf("Type in your expression.\n");
    scanf("%f %c %f", &value1, &operator, &value2);
    switch(operator)
    {
        case '+':
            printf("%.2f\n", value1 + value2);
            break;
        case '-':
            printf("%.2f\n", value1 - value2);
            break;
        case '*':
            printf("%.2f\n", value1 * value2);
            break;
        case '/':
            if ( value2 == 0 )
                printf ("Division by zero.\n");
            else
                printf ("%.2f\n", value1 / value2);
            break;
        default:
            printf ("Unknown operator.\n");
            break;
    }
    return 0;
}
```

```
Type in your expression.
178.99 - 326.8
-147.81
```

- หากมีหลายกรณีที่ต้องทำคำสั่งเดียวกัน เราสามารถเขียนได้ดังนี้

```
switch (operator)
{
    ...
    case '*':
    case 'x':
        printf ("%.2f\n", value1 * value2);
        break;
    ...
}
```

6.7 ตัวแปรชนิด Boolean

- โปรแกรมต่อไปนี้ใช้หาค่าเลขจำนวนเฉพาะที่มีค่าต่ำกว่า 50

Program 6.10 Generating a Table of Prime Numbers

```
// Program to generate a table of prime numbers
#include <stdio.h>
int main (void)
{
    int p, d;
    _Bool isPrime;

    for( p = 2; p <= 50; ++p )
    {
        isPrime = 1;
        for( d = 2; d < p; ++d )
            if( p % d == 0 )
                isPrime = 0;
        if( isPrime )
            printf ("%i ", p);
    }

    printf ("\n");
    return 0;
}
```

```
2 3 5 7 11 13 17 19 23 29 31 37 41 43 47
```

- ในโปรแกรมนี้ isPrime ทำหน้าที่เป็น flag เพื่อบอกว่าตัวเลขนั้นเป็นจำนวนเฉพาะหรือไม่ โดยตัว isPrime จะมีค่าเป็น 0 (FALSE) หรือ 1 (TRUE) เท่านั้น
- นอกจากนี้ค่าคงที่อื่นๆ ที่ไม่ใช่ 0 ภาษา C จะถือว่าเป็น TRUE หมด อย่างเช่น

```
if ( 100 )
    printf ("This will always be printed.\n");
```

- หากเราต้องการกลับ TRUE เป็น FALSE หรือ FALSE เป็น TRUE เราจะใช้ negation operator ! เช่น

```
myMove = ! myMove;
```

- หรือ `!(x < y)` ซึ่งมีความหมายเดียวกับ _____
- ในภาษา C มีไลบรารีสำหรับทำงานกับ Boolean โดยเฉพาะ คือ <stdbool.h> โปรแกรมต่อไปนี้เป็นการแก้ไขโปรแกรม 6.10 โดยใช้ไลบรารีนี้เข้ามาเกี่ยวข้อง

Program 6.10 Program to Generate a Table of Prime Numbers

```
// Program to generate a table of prime numbers
#include <stdio.h>
#include <stdbool.h>
int main (void)
{
    int p, d;
    bool isPrime;
    for( p = 2; p <= 50; ++p )
    {
        isPrime = true;
        for( d = 2; d < p; ++d )
            if( p % d == 0 )
                isPrime = false;
        if(isPrime )
            printf ("%i ", p);
    }
```

```
    printf("\n");
    return 0;
}
```

6.8 Conditional operator

- Conditional operator มีรูปแบบดังนี้

```
condition ? expression1 : expression2;
```

- เมื่อ condition คือเงื่อนไขที่ให้ผลลัพธ์เป็น TRUE หรือ FALSE
- หากผลลัพธ์จากเงื่อนไขเป็น TRUE, expression1 จะถูกกระทำ แต่หากเงื่อนไขให้ผลลัพธ์เป็น FALSE, expression2 จะถูกกระทำ
- ซึ่งสามารถเขียนในรูปแบบของ if-else ได้ดังนี้

```
if(condition)
    expression1;
else
    expression2;
```

- conditional operator นี้นิยมใช้เพื่อกำหนดค่าให้กับตัวแปร โดยมีเงื่อนไขมากำกับ ยกตัวอย่างเช่น หากมี integer s และ x และต้องการให้ s เป็น -1 หาก x น้อยกว่าศูนย์ นอกเหนือจากนั้น ให้ s เท่ากับ x^2 เราเขียนได้ดังนี้

```
s = ( x < 0 ) ? -1 : x * x;
```