บทที่ 4 โครงสร้างคำสั่งควบคุม

(Statements)

ในบทที่ผ่านมา เราให้ความสนใจกับค่าคงที่ ตัวแปร และตัวดำเนินการซึ่งอยู่ในรูปของนิพจน์ต่าง ๆ สำหรับในบทนี้เราจะใช้อธิบายโครงสร้างควบคุมการทำงานภาในโปรแกรมด้วยภาษา C

จากที่เราทราบว่าในการทำชุดคำสั่งใดๆ จะประกอบด้วยขั้นตอนหลักๆ 3 ขั้นตอนดังรูป

รูปที่ 4.1 ขั้นตอนหลักในการประมวลผล

ขั้นตอนการนำข้อมูลเข้า การประมวลผลและแสดงผลลัพธ์ ขั้นตอนทั้ง 3 จะแสดงไว้เป็นลำดับ ซึ่งแสดงถึง แนวคิดทางตรรกะ (logical thinking) ซึ่งจะกลายเป็นขั้นตอนวิธี (procedure) ในการทำงานของชุดคำสั่งใด ๆ

ในลำดับของแนวคิดทางตรรกะ ซึ่งแสดงถึงโครงสร้างควบคุมการทำงานมีอยู่ 3 แบบคือ

- แบบเรียงลำดับ (sequential)
- แบบทางเลือก (selection)
- แบบวนซ้ำ (repetitive)

ในที่นี้จะกล่าวถึงรายละเอียดวิธีโครงสร้างควบคุมแบบทางเลือก และโครงสร้างควบคุมแบบวนซ้ำ

4.1 โครงสร้างแบบทางเลือก

การทดสอบแบบทางเลือก เกิดจากการตั้งเงื่อนไขต่าง ๆ และทำการทดสอบเงื่อนไข ซึ่งแต่ละเงื่อนไขจะมีทางเลือกให้ดำเนินการแตกต่างกัน เช่น

ช่วงคะแนน	**เกรดที่ได้รับ**
>= 90	A
80-89	B
70-79	C
60-69	D
<60	F

เช่นถ้าคะแนนสอบเป็น 75 จะได้เกรดเป็น C ซึ่งเกิดจากการทดสอบพบว่าคะแนนที่ได้ตกอยู่ในช่วง 70 –79 เป็นจริง

ทั้งนี้ ผลลัพธ์จากการทดสอบเกิดจากการตัดสินว่านิพจน์เงื่อนไข(conditional expression) เป็นจริง หรือเท็จ โดยผลลัพธ์ที่ได้จะให้ค่า 1 เมื่อมีค่าเป็น “จริง” และ ให้ค่า 0 เมื่อมีค่าเป็น “เท็จ”

การทดสอบโครงสร้างควบคุมแบบทางเลือกในภาษา C จะใช้คำสั่ง if และ switch ดังนี้

4.1.1 คำสั่ง if

รูปแบบ

```
if ( <conditional expression>)
        <statement-list(s)>;
```

การทำงานจะทำตามลำดับดังนี้

- คอมไพเลอร์ภาษา C จะทำการทดสอบ นิพจน์เงื่อนไข (conditional expression) ซึ่งจะได้ผลลัพธ์เป็นจริง หรือ เท็จ
- หากผลลัพธ์เป็นจริง โปรแกรมจะทำงานในประโยคคำสั่ง หรือกลุ่มของประโยคคำสั่งต่อมาของ if (statement-list(s))
- หากผลลัพธ์เป็นเท็จ โปรแกรมจะไม่สนใจในประโยคคำสั่ง หรือกลุ่มของประโยคคำสั่ง ต่อมาของ if แต่จะไปทำงานในคำสั่งถัดไปจากกลุ่มคำสั่ง if

ตัวอย่างที่ 4.1 โปรแกรมทดสอบคะแนนที่รับเข้า จากผู้ใช้ โดยกำหนดให้อยู่ในช่วง 001-100 หากจำนวนที่ป้อนเข้ามามีค่ามากกว่าหรือเท่ากับ 60 จะพิมพ์ข้อความว่า “PASS” แต่จะไม่พิมพ์ข้อความใดๆ ในกรณีอื่น

```
#include <stdio.h>
main()
{
        float score_in;
        printf ("Enter score  001-100 :   \n");
        scanf("%f",&score_in);
        printf("\tyour score = %5.2f  \n ",score_in);
        if ( score_in >= 60)
                printf("\tPASS\n");
        return 0;
} /* End main */
```

ผลลัพธ์ที่ได้ จากการรันโปรแกรมครั้งที่หนึ่ง ทดสอบโดยใส่ค่า 59.9

```
Enter score  001-100 :
59.9
        your score = 59.90
```

ผลลัพธ์ที่ได้ จากการรันโปรแกรมครั้งที่สอง ทดสอบโดยใส่ค่า 60.0

```
Enter score  001-100 :
60.0
        your score = 60.00
        PASS
```

ตัวอย่างที่ 4.2 จากตัวอย่างที่ 4.1 ทำการทดสอบเพิ่มว่าหากคะแนนที่ป้อนมีค่ามากกว่า 90 เขาจะได้รับเกรด A ส่วนคะแนนนอกเงื่อนไขดังกล่าวจะไม่พิมพ์ข้อความใดๆ

```
#include <stdio.h>
main()
{
        float score_in;
        printf ("Enter score  :   \n");
        scanf("%f",&score_in);
        printf("\tyour score = %5.2f  \n ",score_in);
        if ( score_in >= 60)
                printf("\t\"PASS\"\n");
        if ( score_in >= 90)
                printf("\t\"You got A\"\n");
        return 0;
} /* End main */
```

ผลลัพธ์ที่ได้ จากกการรันโปรแกรมครั้งที่หนึ่ง ทดสอบโดยใส่ค่า 59.9

```
Enter score  001-100 :
59.9
        your score = 59.90
```

ผลลัพธ์ที่ได้ จากกการรันโปรแกรมครั้งที่สอง ทดสอบโดยใส่ค่า 60.0

```
Enter score 001-100 :
60.0
        your score = 60.00
        "PASS"
```

ผลลัพธ์ที่ได้ จากกการรัน โปรแกรมครั้งที่สาม ทดสอบโดยใส่ค่า 90.01

```
Enter score 001-100 :
90.01
        your score = 90.01
        "PASS"
        "You got A"
```

4.1.2 คำสั่ง if - else

ในการทดสอบเงื่อนไขบางกรณี กำหนดให้ทำงานอย่างหนึ่งเมื่อเงื่อนไขเป็นจริงและทำงานอย่างอื่นเมื่อเงื่อนไขเป็นเท็จ ในกรณีเช่นนี้สามารถใช้ if ชนิดมี else ซึ่งมีรูปแบบ ดังนี้

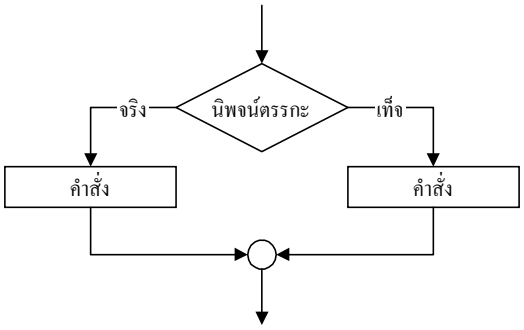
รูปแบบ

```
if ( <conditional expression>)
        <statement-list1(s)>;
else
        <statement-list2(s)>;
```

การทำงานจะทำตามลำดับดังนี้

- คอมไพเลอร์ภาษา C จะทำการทดสอบ นิพจน์เงื่อนไข (conditional expression) ซึ่งจะได้ผลลัพธ์เป็นจริง หรือ เท็จ
- หากผลลัพธ์เป็นจริง โปรแกรมจะทำงานในประโยคคำสั่งที่ 1 หรือกลุ่มของประโยคคำสั่งที่ 1 ซึ่งต่อจาก if (statement-list1(s));
- หากผลลัพธ์เป็นเท็จ โปรแกรมจะทำงานในประโยคคำสั่งที่ 2 หรือกลุ่มของประโยคคำสั่งที่ 2 ซึ่งต่อจาก else (statement-list2(s));

สามารถเขียนผังงานได้ดังนี้



รูปที่ 4.1 แสดงผังงานการทำงานของคำสั่ง if-else

ตัวอย่างที่ 4.3 โปรแกรมทดสอบจำนวนที่รับเข้าจากผู้ใช้ โดยกำหนดให้อยู่ในช่วง 00-99 และทำการทดสอบว่าจำนวนดังกล่าวมากกว่าหรือเท่ากับ 50 หรือน้อยกว่า 50

```
#include <stdio.h>
main()
{
        int x;
        printf ("Enter the number 00-99  :  ");
        scanf("%d",&x);
        printf("The value of X is %d\n",x);
        if ( x >= 50)
                printf("The value of X is higher than or equal to 50\n");
        else
                printf("The value of X is lower than 50\n");
        return 0;
} /* End main */
```

ผลลัพธ์ที่ได้ จากการรันโปรแกรมครั้งที่หนึ่ง ทดสอบโดยใส่ค่า 49

```
Enter the number 00-99  :  49
The value of X is 49
The value of X is lower than 50
```

ผลลัพธ์ที่ได้ จากการรันโปรแกรมครั้งที่สอง ทดสอบโดยใส่ค่า 50

```
Enter the number 00-99  :  50
The value of X is 50
The value of X is higher than or equal to 50
```

จากตัวอย่างที่ยกมาข้างต้นจะเห็นว่าประโยคคำสั่งที่ตามหลัง if หรือ else เป็นประโยคโดด ๆ แต่เราสามารถเขียนกลุ่มคำสั่งตามหลัง if หรือ else ได้ กลุ่มคำสั่งดังกล่าวต้องเขียนไว้ภายใต้บล็อค (บล็อคแต่ละบล็อคจะขึ้นต้นด้วยเครื่องหมาย ‘{‘ และปิดด้วยเครื่องหมาย ‘}’)

ตัวอย่างที่ 4.4 จากโปรแกรมที่ 4.3 หากจำนวนที่รับเข้ามามากกว่าหรือเท่ากับ 50 ให้หาค่ากำลังสามของจำนวนนั้น นอกนั้นให้หาค่ากำลังสอง

```
#include <stdio.h>
main()
{
        int x;
        long y;
        printf ("Enter the number 00-99  :  ");
        scanf("%d",&x);
        printf("The value of X is %d\n",x);
        if ( x >= 50)
        {
                printf("The value of X is higher than or equal 50\n");
                y = x*x;
                printf("The value of  X * X *X =  %ld\n", y*x);
        } /* End if */
        else
        {
                printf("The value of X is lower than 50\n");
                printf("The value of X * X  =  %d\n",x*x);
        } /* End else */

        return 0;
} /* End main */
```

ผลลัพธ์ที่ได้ จากกการรันโปรแกรมครั้งที่หนึ่ง ทดสอบโดยใส่ค่า 49

```
Enter the number 00-99  :  49
The value of X is 49
The value of X is lower than 50
The value of X * X  =  2401
```

ผลลัพธ์ที่ได้ จากกการรันโปรแกรมครั้งที่สอง ทดสอบโดยใส่ค่า 50

```
Enter the number 00-99  :  50
The value of X is 50
The value of X is higher than or equal 50
The value of  X * X *X =  125000
```

4.1.3 คำสั่ง If ซ้อน

จากตัวอย่างที่ 4.2 พบว่าคะแนนที่ไม่ผ่านเกณฑ์คือน้อยกว่า 60 นั้นก็ยังต้องผ่านการทดสอบว่าได้ A หรือไม่ ซึ่งผู้เขียนโปรแกรมไม่ควรทำเช่นนั้นเพราะไม่ให้ประโยชน์ใด ๆ เราควรกำหนดว่าเฉพาะผู้สอบตามเกณฑ์เท่านั้น จึงจะมีสิทธิ์เข้าทำการทดสอบว่าได้ A หรือไม่ การทำงานที่เป็นการทดสอบแบบลำดับต่อเนื่องนี้ สามารถเขียนโดยใช้ If แบบซ้อน

รูปแบบ

```
if ( <conditional expression1>)
     <statement-list1(s)>;
else
         if ( <conditional expression2>)
            <statement-list2(s)>;
         else
                 if ( <conditional expression3>)
                      <statement-list3(s)>;
                 else
                          if ( <conditional expression4>)
                             <statement-list4(s)>;
                          else
                             <statement-list5(s)>;
```

การทำงานจะทำตามลำดับดังนี้

- คอมไพเลอร์ภาษา C จะทำการทดสอบ นิพจน์เงื่อนไขที่ 1 (conditional expression1) ซึ่งจะได้ผลลัพธ์เป็นจริง หรือ เท็จ
- หากผลลัพธ์เป็นจริง โปรแกรมจะทำงานในประโยคคำสั่งที่ 1 หรือกลุ่มของประโยคคำสั่งที่ 1 (statement-list1(s)) และเสร็จสิ้นการทำงานของคำสั่ง if
- หากผลลัพธ์เป็นเท็จ จะทำการทดสอบ นิพจน์เงื่อนไขที่ 2 (conditional expression2) ซึ่งจะได้ผลลัพธ์เป็นจริง หรือ เท็จ
- หากผลลัพธ์เป็นจริง โปรแกรมจะทำงานในประโยคคำสั่งที่ 2 หรือกลุ่มของประโยคคำสั่งที่ 2 (statement-list2(s)) และเสร็จสิ้นการทำงานของคำสั่ง if
- การทดสอบลำดับถัดไป หากผลลัพธ์เป็นเท็จ จะทำการทดสอบในลักษณะข้างต้นและทำเช่นนี้ไปเรื่อยๆ จนจบชุดคำสั่ง if

ตัวอย่างที่4.5 โปรแกรมแสดงการคำนวณหาคะแนนเฉลี่ยและเกรดที่ได้รับ โดยใช้คำสั่ง if ซ้อน ดังเงื่อนไขต่อไปนี้

ช่วงคะแนน	**เกรดที่ได้รับ**
>= 90	A
80-90	B
70-80	C
60-70	D
<60	F

```
#include <stdio.h>

main()
{
        float score_in1,score_in2,score_in3,average;
        printf ("Enter all scores ");
        scanf("%f %f %f",&score_in1,&score_in2,&score_in3);
        average =( score_in1+score_in2+score_in3)/3;
        if ( average >=90)
                printf("Your average score is \t%5.2f\t ,so your grade =
                A\n",average);
        else
                if ( average >=80)
                        printf("Your average score is \t%5.2f\t ,so your
                        grade = B\n",average);
                else
                        if ( average >=70)
                                printf("Your average score is \t%5.2f\t ,so
                                your grade = C\n",average);
                        else
                                if ( average >=60)
                                        printf("Your average score is
                                        \t%5.2f\t ,so your grade =
                                        D\n",average);
                                else
                                        printf("Your average score is
                                        \t%5.2f\t ,so your grade =
                                        F\n",average);
        return 0;
}/* End main */
```

ผลที่ได้รับ

```
Enter all scores 65.3 78.5 82.3
Your average score is   75.37    ,so your grade = C
```

4.1.4 switch

การทดสอบแบบทางเลือกข้างต้น เป็นการทดสอบในลักษณะทางเลือกสองทาง แต่เราสามารถเขียนการทดสอบแบบหลายทางเลือกได้ โดยใช้ switch

รูปแบบ

```
switch (<expression>)
{
        <case case-label-1: statement-list1(s);>
<case case-label-2: statement-list2(s);>
<case case-label-3: statement-list3(s);>
…
<case case-label-n: statement-listn(s);>
<default : statement-list0(s);>
}
```

หมายเหตุ default นั้นอาจใช้หรือไม่ก็ได้

การทำงานจะทำการทดสอบนิพจน์ในคำสั่ง switch และนำผลลัพธ์ที่ได้จากการทดสอบมาเปรียบเทียบกับ label ในแต่ละ case หากไม่พบจะไปทำงานที่ default หลังการทำงานในคำสั่ง case แต่ละตัว โปรแกรมจะไปทดสอบคำสั่ง case ตัวถัดไป ซึ่งไม่ใช่ความประสงค์ของผู้เขียนโปรแกรม ดังนั้นเพื่อให้เสร็จสิ้นการทำงานในคำสั่ง case เราจึงมักนิยมใส่คำสั่ง break ตามหลังแต่ละ case ดังนี้

```
switch (<expression>)
{
        <case case-label-1: statement-list1(s);>
        break;
<case case-label-2: statement-list2(s);>
break;
<case case-label-3: statement-list3(s);>
break;
…
<case case-label-n: statement-listn(s);>
break;
<default : statement-list0(s);>
}
```

ตัวอย่างที่ 4.6 โปรแกรมเพื่อทำการทายว่าเลขโดด(0-9) ที่อยู่ในใจของผู้ใช้คือจำนวนใด โดยมีเงื่อนไขให้ผู้ใช้ป้อนข้อมูลดังนี้

- ป้อน 0 หรือ 1 ในกรณีต่อไปนี้
  - 1 หากจำนวนที่นึกอยู่เป็นจำนวนคู่
  - 0 หากจำนวนที่นึกอยู่เป็นจำนวนคี่
- ป้อนค่าเศษที่ได้จากการหารจำนวนดังกล่าวด้วย 5 ถ้าหารลงตัวให้ระบุเศษเป็น 0

วิธีการคิด

เลขโดด	**จำนวนคู่/คี่**	**เศษของจำนวนที่หารด้วย 5**
0	1	0
1	0	1
2	1	2
3	0	3
4	1	4
5	0	0
6	1	1
7	0	2
8	1	3
9	0	4

```
#include <stdio.h>
main()
{
        int digit, oddoreven, modby5;
        printf ("Type 0 if even and 1 if odd.\n");
        scanf("%d", &oddoreven);
        printf("What is the remainder when the digit is divided by 5.\n");
        scanf("%d", &modby5);
        switch (oddoreven)
        {
                case 0 : switch (modby5)
                {
                        case 0 : digit = 0; break;
                        case 1 : digit = 1; break;
                        case 2 : digit = 7; break;
                        case 3 : digit = 3; break;
                        case 4 : digit = 9; break;
                } /* End case 0 */
                break;
                case 1 : switch (modby5)
                {
                        case 0 : digit = 0; break;
                        case 1 : digit = 6; break;
                        case 2 : digit = 2; break;
                        case 3 : digit = 8; break;
                        case 4 : digit = 4; break;
                } /* End case 1 */
                break;
        } /* End switch */

        printf ("\tYour digit is %d\n", digit);
        return 0;
}/* End main */
```

ผลลัพธ์ที่ได้ จากกการรันโปรแกรมครั้งที่หนึ่ง ทดสอบโดยใส่ค่า 0 และ 2

```
Type 0 if even and 1 if odd.
0
What is the remainder when the digit is divided by 5.
2
        Your digit is 7
```

ผลลัพธ์ที่ได้ จากกการรันโปรแกรมครั้งที่สอง ทดสอบโดยใส่ค่า 1 และ 2

```
Type 0 if even and 1 if odd.
1
What is the remainder when the digit is divided by 5.
2
        Your digit is 2
```

4.2 โครงสร้างควบคุมแบบวนซ้ำ

ถ้าบางส่วนของชุดคำสั่งในโปรแกรมมีการทำงานซ้ำมากกว่า 1 ครั้ง เราจะใช้โครงสร้างควบคุมแบบวนซ้ำ ซึ่งภาษา C มีรูปแบบการเขียนคำสั่งควบคุมแบบวนซ้ำได้ในหลายลักษณะดังนี้

4.2.1 วงวน for

รูปแบบ for (<initial condition>; <test condition>; <loop operation>)

<statement-list(s);>

การทำงานจะทำตามลำดับดังนี้

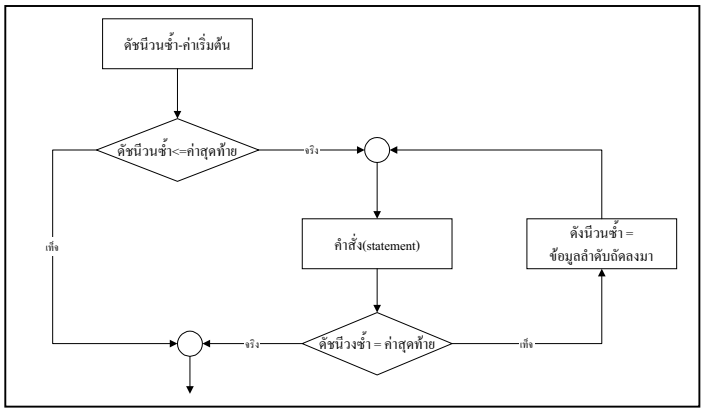
(a) ทำงานในนิพจน์ <initial condition> เพื่อกำหนดค่าเริ่มต้นของดัชนีในวงวน

(b) การทดสอบ <test condition> เพื่อกำหนดเงื่อนไขการทำงาน คือ

- ถ้าเป็นเท็จ จะเสร็จการทำงานในวงวน for
- ถ้าเป็นจริง จะทำงานในประโยคคำสั่ง หรือกลุ่มของประโยคคำสั่งในชุดของวงวน for

(c) ทำงานในนิพจน์ <loop operation> เพื่อเพิ่มหรือลดค่าของดัชนีของวงวน for และไปทำข้อ (b)

สามารถเขียนผังงานได้ดังนี้



รูปที่ 4.2 แสดงผังงานการทำงานของคำสั่ง for

ตัวอย่างที่ 4.7 โปรแกรมพิมพ์กรอบสี่เหลี่ยมผืนผ้า ขนาดกว้าง 20 ตัวอักษร สูง 12 ตัวอักษร และให้พิมพ์ข้อความ example ไว้ในบรรทัดที่สอง โดยให้พิมพ์กรอบด้วยอักขระ “*”

```
#include <stdio.h>

main()
{
        int count;
        printf("\t");
        for (count = 1; count<=20;count++)
        {
                printf("*");
        } /* End for */
        printf("\n");
        printf("\t*    EXAMPLE     *\n");
        for (count = 1; count<=9;count++)
        {
                printf("\t*                *\n");
        } /* End for */
        printf("\t");
        for (count = 1; count<=20;count++)
        {
                printf("*");
        }/* End for */
        printf("\n");
        return 0;
} /* End main */
```

ผลลัพธ์ที่ได้

```
**********************
*      EXAMPLE       *
*                    *
*                    *
*                    *
*                    *
*                    *
*                    *
*                    *
*                    *
*                    *
**********************
```

ตัวอย่างที่ 4.8 โปรแกรมพิมพ์ตารางผลคูณของตัวแปร i และ j โดย i และ j มีค่า ตั้งแต่ 1 ถึง 12

```
#include <stdio.h>
main()
{
        int i,j;
        for (i=1; i<=12;i++)
        {
                printf("\n");
                for (j =1; j<=12;j++)
                {
                        printf("\t%d",i*j);
                } /* End for j */
        } /* End for i */
        return 0;
} /* End main */
```

ผลลัพธ์ที่ได้

```
 1    2    3    4    5    6    7    8    9   10   11   12
 2    4    6    8   10   12   14   16   18   20   22   24
 3    6    9   12   15   18   21   24   27   30   33   36
 4    8   12   16   20   24   28   32   36   40   44   48
 5   10   15   20   25   30   35   40   45   50   55   60
 6   12   18   24   30   36   42   48   54   60   66   72
 7   14   21   28   35   42   49   56   63   70   77   84
 8   16   24   32   40   48   56   64   72   80   88   96
 9   18   27   36   45   54   63   72   81   90   99  108
10   20   30   40   50   60   70   80   90  100  110  120
11   22   33   44   55   66   77   88   99  110  121  132
12   24   36   48   60   72   84   96  108  120  132  144
```

4.2.2 วงวน while

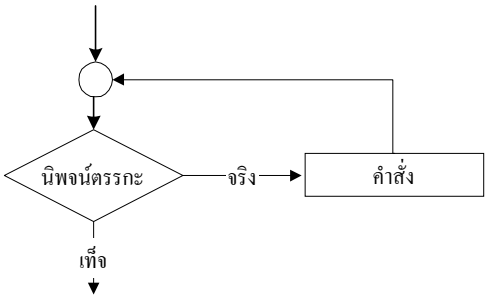
ผู้ใช้อาจสังเกตเห็นว่าการเขียนเงื่อนไขในวงวน for ค่อนข้างที่จะมีลักษณะเฉพาะเจาะจง คือ ต้องทราบจำนวนครั้งที่จะทำซ้ำ แต่ในบางครั้งอาจต้องการเขียนเงื่อนไขเพื่อให้ทำงานในวงวนที่มีลักษณะยืดหยุ่น เช่น จะทำงานในวงวน ถ้าหากค่าที่รับเข้าไม่ใช่ศูนย์ หรือไม่ทราบจำนวนครั้งที่จะทำงานในวงวน ในกรณีนี้ควรใช้วงวนwhile ทำงานแทนวงวน for

รูปแบบ while (<conditional expression>)

<statement-list(s);>

การทำงานจะทำงานในวงวน while ตราบเท่าที่นิพจน์ (conditional expression) เป็นจริง

สามารเขียนผังงานได้ดังนี้



รูปที่ 4.3 แสดงผังงานการทำงานของคำสั่ง while

ตัวอย่างที่ 4.9 ในการพิจารณา 1+2+3+…+ n ต้องการทราบว่า พจน์ที่ทำให้ผลรวมดังกล่าวมีค่าเกินร้อย เป็นพจน์แรกคือค่าอะไร และค่าผลรวมนั้นคืออะไร

```
#include <stdio.h>
main()
{
        int  sum,n;
        sum = n = 0;
        while (sum <= 100)
        {
                n++;
                sum += n;
        } /* End while */
        printf("\n\t n  : \t%6d\n\tSum : \t%6d\n",n,sum);
        return 0;
} /* End main */
```

ผลลัพธ์ที่ได้

```
 n   :       14
Sum  :      105
```

แบบฝึกหัด

1. จงแปลงองศาฟาเรนไฮด์ให้เป็นองศาเซลเซียส โดยใช้สูตร (C/5) = (f-32)/9 ทั้งนี้ C หมายถึงองศาเซลเซียส และF หมายถึงองศาฟาเรนไฮด์ และกำหนดรูปแบบผลลัพธ์ดังนี้

FARENHITE	CELCIUS
100	XX.X
80	XX.X
60	XX.X
40	XX.X
20	XX.X
0	XX>X

2. ให้เขียนโปรแกรมว่าปีคศ. ที่สนใจเป็นปีอธิกวาร (leap year) หรือไม่ ซึ่งปีอธิกวารจะหารด้วย 4 ลงตัว แต่หารด้วย 100 ไม่ลงตัว หรือเป็นปีคศ. ซึ่งหารลงตัวด้วย 400
3. จอห์น และ จิม ลงสมัครรับเลือกตั้งเป็นประธานนักเรียน นักเรียนที่มีสิทธิลงคะแนนมีจำนวน 500 คนซึ่ง จะนับคะแนนที่แต่ละคนได้รับจากแผงแป้นอักขระ จะตัดสินว่าผู้ใดชนะก็ต่อเมื่อเปอร์เซนต์ของผู้ออกเสียงให้แก่ผู้สมัครคนนั้นมากกว่าอีกคนหนึ่ง
4. จงหาผลบวกของ 1/1+1/2+1/3+...+1/n โดยรับค่า n จากผู้ใช้
5. จงหาว่าจำนวนเต็มบวกที่รับเข้ามาคือจำนวนเฉพาะ