3.1 คำสั่งเงื่อนไข

การเขียนโปรแกรมภาษาซี จะมีรูปแบบการใช้คำสั่งเงื่อนไขโดยจะเป็นการสร้างทางเลือกให้กับโปรแกรมแสดงผลต่างๆ กันออกไปตามที่กำหนดไว้ในแต่ละเงื่อนไข โดยใช้คำสั่ง if ซึ่งแปลตรงตัวว่า “ถ้า” โดยมีรูปแบบดังนี้

```
if(เงื่อนไข){
    คำสั่งเมื่อเงื่อนไขเป็นจริง
}
```

รูปแบบคำสั่งของ if คือ ถ้าเงื่อนไขเป็นจริงจะทำคำสั่งที่อยู่ระหว่างวงเล็บปีกกาให้ ไม่ว่าคำสั่งจะมีกี่ร้อยกี่พันบรรทัดก็ตาม โดยจะต้องใช้ตัวดำเนินการทางคณิตศาสตร์กลุ่มที่ใช้ในการเปรียบเทียบมาใช้ในการสร้างเงื่อนไข

==	เท่ากัน	!=	ไม่เท่ากัน
>	มากกว่า	<	น้อยกว่า
>=	มากกว่าหรือเท่ากับ	<=	น้อยกว่าหรือเท่ากับ

การสร้างเงื่อนไขนั้นจะใช้ค่าของตัวแปรมาเปรียบเทียบกับค่าอื่นๆ เช่น ตรวจสอบว่า ตัวแปร x มีค่าเท่ากับ 5 หรือไม่ ก็สร้างเงื่อนไขเป็น x==5 โดยยึดตามรูปแบบดังนี้

ค่าที่ตรวจสอบ **ตัวดำเนินการ** *ค่าเปรียบเทียบ*

** สิ่งที่ควรระวังที่มักจะผิดกันบ่อยๆ คือ ในเงื่อนไขจะใช้ == หมายถึงการเท่ากัน ห้ามใช้ = ตัวเดียวเด็ดขาด เพราะ = ตัวเดียว ในภาษาซีหมายถึงการกำหนดค่า

โปรแกรมที่ 3.1 ตรวจสอบว่าเลขที่รับมามีค่ามากกว่า 10 ใช่หรือไม่ ถ้าใช่ให้แสดงคำว่า GOOD

```
#include <stdio.h>
main() {
    int a;
    printf("input number = ");
    scanf("%d",&a);
    if(a>10){
        printf("GOOD");
    } ** ระวังอย่าลืมวงเล็บปิดนะ มือใหม่มักลืม
}     กันบ่อยๆ อย่าไปสับสนกับวงเล็บปิดของ main
```

ผลการรันโปรแกรม

```
input number = 15
GOOD
```

ผลการรันโปรแกรม

```
input number = 3
```

จากโปรแกรมที่ 3.1 ครั้งแรกลองกรอกตัวเลข 15 จะได้ผลลัพธ์เป็น GOOD เพราะ 15 มีค่ามากกว่า 10 เป็นจริงตามเงื่อนไข จึงแสดงคำสั่งในวงเล็บปีกกาหลังเงื่อนไขออกมา แต่ในการทดลองรันโปรแกรมครั้งที่สอง กรอกตัวเลขเป็น 3 จะพบว่าโปรแกรมไม่ได้แสดงผลลัพธ์อะไรออกมา เนื่องจาก 3 มีค่าน้อยกว่า 10 ซึ่งก็คือเงื่อนไขไม่เป็นจริง

การใช้คำสั่ง if จะมีการใช้คำสั่งต่อเนื่องกันคือคำสั่ง else โดยจะทำงานเมื่อโปรแกรมตรวจสอบพบว่าเงื่อนไขเป็นเท็จ ก็จะไปทำคำสั่งหลัง else แทน โดยมีรูปแบบดังนี้

```
if(เงื่อนไข){
   คำสั่งเมื่อเงื่อนไขเป็นจริง
}
else{
   คำสั่งเมื่อเงื่อนไขเป็นเท็จ
}
```

โปรแกรมที่ 3.2 ถ้าค่ามากกว่า 10ให้แสดงคำว่า GOOD ถ้าไม่ใช่ให้แสดงคำว่า BAD

```
#include <stdio.h>
main() {
    int a;
    printf("input number = ");
    scanf("%d",&a);
    if(a>10){
        printf("GOOD");
    }
    else{
        printf("BAD");
    }
}
```

ผลการรันโปรแกรม

```
input number = 15
GOOD
```

ผลการรันโปรแกรม

```
input number = 3
BAD
```

จากโปรแกรมที่ 3.2 เมื่อกรอกตัวเลข 15 จะได้ผลลัพธ์เป็น GOOD เพราะ 15 มีค่ามากกว่า 10 เป็นจริงตามเงื่อนไข แต่เมื่อกรอกตัวเลขเป็น 3 จะพบว่าได้ผลลัพธ์เป็น BAD เนื่องจาก 3 มีค่าน้อยกว่า 10 ซึ่งก็คือเงื่อนไขไม่เป็นจริง จึงมาทำคำสั่งหลัง else แทน

เพิ่มเติม

การใช้ { } ในคำสั่ง if

โดยปกติแล้วรูปแบบคำสั่ง if จะมีการใช้วงเล็บ { } หนึ่งคู่หลังเงื่อนไข ซึ่งจะทำคำสั่งที่อยู่ระหว่างวงเล็บ { } เมื่อเงื่อนไขเป็นจริง แต่ถ้าคำสั่งที่ให้ทำมีเพียงคำสั่งเดียว สามารถลดรูป โดยการไม่ใส่วงเล็บ { } ก็ได้

```
if(a>10)   printf("GOOD");
else       printf("BAD");
```

นอกจากนี้การสร้างเงื่อนไขยังสามารถเปรียบเทียบค่าจากสูตรคำนวณก็ได้ เช่น สร้างเงื่อนไข x+5>10 โปรแกรมจะทำการนำค่าของตัวแปร x ไปบวก 5 ก่อนแล้วจึงนำมาเปรียบเทียบว่ามากกว่า 10 หรือไม่

สูตรคำนวณ ตัวดำเนินการ *ค่าเปรียบเทียบ*

ตัวอย่างที่ 3.1

เขียนโปรแกรมรับค่าเงินเดือนกับรายจ่าย ถ้าเงินเดือนมากกว่า 3 เท่าของรายจ่าย ให้แสดงคำว่า GOOD ถ้าไม่ใช่ให้แสดงคำว่า BAD

```
#include <stdio.h>
main(){
   int a,b;
   printf("input salary = ");
   scanf("%d",&a);
   printf("input expend = ");
   scanf("%d",&b);
   if(a>b*3){
       printf("GOOD");
    }
    else{
       printf("BAD");
    }
}
```

ผลการรันโปรแกรม

```
input salary = 10000
input expend = 2500
GOOD
```

ผลการรันโปรแกรม

```
input salary = 5000
input expend = 2500
BAD
```

**จากตัวอย่างนี้แสดงให้เห็นว่า เงื่อนไขสามารถใช้ตัวแปรมาเปรียบเทียบกัน หรือสูตรคำนวณมาเปรียบเทียบก็ได้

ตัวอย่างที่ 3.2

จงเขียนโปรแกรมรับค่าตัวเลขจำนวนเต็ม 1 ตัว ถ้าเป็นเลขคู่ให้แสดงคำว่า Even Number ถ้าเป็นเลขคี่ให้แสดงคำว่า Odd Number

```
#include <stdio.h>

main(){

   int num;

   printf("input number = ");

   scanf("%d",&num);

   if(num%2==0){

       printf("Even Number");

    }

    else{

       printf("Odd Number");

    }

}
```

ผลการรันโปรแกรม

```
Input number = 20
Even Number
```

ผลการรันโปรแกรม

```
Input number = 21
Odd Number
```

****คุณสมบัติของเลขคู่ คือ หารด้วย2ลงตัว ดังนั้นจึงนำ % มาช่วยในการตรวจสอบเงื่อนไข**

จากตัวอย่างที่ 3.2 จะเป็นการสร้างเงื่อนไขเพื่อหาว่าตัวเลขที่รับมาเป็นเลขคู่หรือเลขคี่ ซึ่งคุณสมบัติของเลขคู่คือ หารด้วย 2 ลงตัว ก็คือ ตัวเลขดังกล่าวถ้าหารด้วย 2 จะได้เศษเป็น 0 ดังนั้นเราจึงนำตัวดำเนินการ % (mod) มาช่วยในการตรวจสอบเศษจากการหาร

จุดสำคัญของบทนี้คือการสร้างเงื่อนไข นักเรียนจะต้องสามารถวิเคราะห์โจทย์เพื่อสร้างเงื่อนไขให้ได้ และต้องระวังเรื่องการใช้ตัวดำเนินการเปรียบเทียบให้ถูกต้องตามความหมายเช่น มีค่าไม่เกินใช้ <= มีค่าไม่ถึงใช้ < มีค่าตั้งแต่ >= มีค่าเกินกว่าใช้ > เป็นต้น

ตัวอย่างที่ 3.3

จงเขียนโปรแกรมคำนวณภาษีจากเงินเดือน โดยรับค่าเงินเดือนจากคีย์บอร์ด ถ้าเงินเดือนตั้งแต่ 20,000 บาทขึ้นไปจะเสียภาษี 3% ถ้าไม่ถึงไม่เสียภาษี

```
#include <stdio.h>

main(){

   float salary;

   printf("input salary = ");

   scanf("%f",&salary);

   if(salary>=20000){

        printf("tax=%.2f",salary*3/100);

    }

    else{

        printf("tax=0");

    }

}
```

ผลการรันโปรแกรม

Input salary = 20000 ↵

tax=600.00

ผลการรันโปรแกรม

Input salary = 10000 ↵

tax=0

**จากตัวอย่างนี้จะเห็นว่าคำสั่งในเงื่อนไขก็สามารถใช้เป็นสูตรคำนวณได้เช่นกัน

นักเรียนจะเห็นว่าจากคำสั่งเล็กๆ น้อยๆ หรือรูปแบบคำสั่งต่างๆ ที่ผ่านมานั้นสามารถนำมาใช้แก้ปัญหาร่วมกับการเขียนโปรแกรมรูปแบบใหม่ๆ ได้เสมอ ดังนั้นหากยังไม่แม่นเรื่องพื้นฐาน ลองกลับไปทบทวนในบทที่แล้วดูสักหนึ่งรอบนะ

สื่อมัลติมีเดียที่ 3.1

หัวข้อ

คำสั่งเงื่อนไข

Link

http://www.sa.ac.th/c/3-1.mp4

3.2 คำสั่งเงื่อนไขต่อเนื่อง

ในหัวข้อที่แล้ว เราได้ใช้คำสั่ง if ในการเขียนโปรแกรม โดยการสร้างเงื่อนไข ถ้าพบว่าเงื่อนไขเป็นจริง โปรแกรมก็จะทำคำสั่งที่อยู่ในวงเล็บปีกกาหลังเงื่อนไข แต่ถ้าพบว่าเงื่อนไขเป็นเท็จก็จะไปทำคำสั่งในวงเล็บปีกกาหลัง else แทน แต่ถ้าเกิดเราเจอกรณีที่เงื่อนไขในการตรวจสอบมีมากกว่าสองเงื่อนไขขึ้นมา เราจะไม่สามารถใช้แค่คำสั่ง if และ else ได้ จะต้องใช้คำสั่ง else if มาช่วย โดยมีรูปแบบการใช้งานดังนี้

```
if(เงื่อนไข 1){
   คำสั่งเมื่อเงื่อนไข 1 เป็นจริง
}
else if(เงื่อนไข 2){
   คำสั่งเมื่อเงื่อนไข 2 เป็นจริง
}
.
.
else if(เงื่อนไข n){
   คำสั่งเมื่อเงื่อนไข n เป็นจริง
}
```

รูปแบบของ else if จะต้องเริ่มด้วย if ก่อนเสมอ โดยโปรแกรมจะทำการตรวจสอบเงื่อนไขที่ 1 ถ้าผมว่าเงื่อนไข 1 เป็นจริงก็จะทำคำสั่งในวงเล็บปีกกาหลังเงื่อนไขที่ 1 แล้วจบการทำงานของกลุ่มคำสั่ง if แต่ถ้าพบว่าเงื่อนไข 1 เป็นเท็จก็จะไปตรวจสอบเงื่อนไข 2 หลัง else if ต่อไปทันที เป็นอย่างนี้ไปเรื่อยๆ จนกว่าจะพบเงื่อนไขที่เป็นจริง หรือหมดเงื่อนไข ซึ่งสามารถกำหนดจำนวนเงื่อนไขได้ไม่จำกัด

โปรแกรมที่ 3.3 **ถ้าค่ามากกว่า 10 ให้แสดงคำว่า GOOD** ถ้าน้อยกว่า 10 ให้แสดงคำว่า BAD ถ้าเท่ากับ 10 ให้แสดงคำว่า OK

```
#include <stdio.h>
main() {
    int a;
    printf("input number = ");
    scanf("%d",&a);
    if(a>10){
        printf("GOOD");
    }
    else if(a<10){
        printf("BAD");
    }
    else if(a==10){
        printf("OK");
    }
}
```

ผลการรันโปรแกรม

```
input number = 15
GOOD
```

ผลการรันโปรแกรม

```
input number = 3
BAD
```

ผลการรันโปรแกรม

```
input number = 10
OK
```

จากโปรแกรมที่ 3.3 ในครั้งแรก เรากรอกตัวเลขไป 15 เมื่อโปรแกรมตรวจสอบเงื่อนไขที่ 1 พบว่า 15 มากกว่า 10 จริง ก็จะแสดงคำว่า GOOD แล้วจบคำสั่ง if แค่นั้นโดยไม่สนใจเงื่อนไขที่เหลือ ครั้งที่สอง กรอกตัวเลข 3 เมื่อตรวจสอบเงื่อนที่ 1 พบว่า 3 มากกว่า 10 เป็นเท็จ จึงข้ามไปตรวจสอบเงื่อนไขที่ 2 หลัง else if พบว่า 3 น้อยกว่า 10 เป็นจริง จึงแสดงคำว่า BAD ออกมาแล้วจบคำสั่ง if ครั้งที่สาม ทดลองกรอกตัวเลข 10 จากการตรวจสอบเงื่อนไขที่ 1 และ 2 พบว่าเป็นเท็จจึงข้ามมาตรวจสอบเงื่อนไขที่ 3 ซึ่ง 10 มีค่าเท่ากับ 10 จึงแสดงคำว่า OK ออกมา

ข้อแตกต่างระหว่างการใช้ if อย่างเดียวกับการใช้ else if ให้นักเรียนลองสังเกตจากตัวอย่างนี้

```
if(a>10){
    printf("GOOD");
}
else if(a<10){
    printf("BAD");
}
else if(a==10){
    printf("OK");
}
```

```
if(a>10){
    printf("GOOD");
}
if(a<10){
    printf("BAD");
}
if(a==10){
    printf("OK");
}
```

ถ้าเรากรอกตัวเลขลงไปเป็น 15 ผลลัพธ์ที่ได้จากโค้ดโปรแกรมในกรอบซ้ายมือและขวามือจะได้ผลลัพธ์ออกมาเหมือนกันคือ GOOD แต่สิ่งที่แตกต่างกันคือระยะเวลาประมวลผล ในกรอบซ้ายมือ เมื่อโปรแกรมตรวจสอบเงื่อนไขที่ 1 พบว่าเป็นจริงแล้วจะทำคำสั่งที่อยู่ในวงเล็บปีกกาแล้วจบการทำงาน if ทันที แต่ในกรอบขวามือ เมื่อโปรแกรมตรวจสอบเงื่อนไขที่ 1 พบว่าเป็นจริงแล้วทำคำสั่งแล้ว โปรแกรมก็จะไปตรวจสอบเงื่อนไขของ if ตัวที่ 2 และ 3 ต่อไปจนครบทั้งๆ ที่ไม่จำเป็นเลย เนื่องจากเราได้ผลลัพธ์ที่ต้องการแล้ว ผิดกับ else if ที่ทำให้เงื่อนไขทั้งหมดกลายเป็นกลุ่มคำสั่งเดียว มีการตรวจสอบเงื่อนไขต่อเนื่องกัน เมื่อพบเงื่อนไขที่เป็นจริงแล้วก็ไม่จำเป็นจะต้องตรวจสอบเงื่อนไขต่อไป แต่โค้ดในกรอบขวามือก็ไม่ใช่ว่าผิด สามารถใช้ได้หากเจอกรณีที่โปรแกรมต้องการแสดงผลหลายแบบ ขึ้นอยู่กับว่าต้องการผลลัพธ์แบบไหน

เพิ่มเติม

การใช้คำสั่ง else if ในกรณีที่เงื่อนไขสุดท้ายคือเงื่อนไขที่ครอบคลุมถึงกรณีที่เหลือทั้งหมด สามารถลดรูปได้ด้วยการใช้ else อย่างเดียว

```
if(a>10){
    printf("GOOD");
}
else if(a<10){
    printf("BAD");
}
else{
    printf("OK");
}
```

ตัวอย่างที่ 3.4

จงเขียนโปรแกรมตัดเกรดโดยรับค่าคะแนนจากคีย์บอร์ด ถ้าได้คะแนนตั้งแต่ 80 ขึ้นไปได้เกรด G ถ้าได้คะแนนตั้งแต่ 50 แต่ไม่ถึง 80 ได้เกรด P ถ้าได้คะแนนไม่ถึง 50 ได้เกรด F

```
#include <stdio.h>
main(){
   int sc;
   printf("input score = ");
   scanf("%d",&sc);
   if(sc>=80){
        printf("grade G");
    }
    else if(sc>=50){
        printf("grade P");
    }
    else if(sc<50){
        printf("grade F");
    }
}
```

ผลการรันโปรแกรม

```
input score = 90
grade G
```

ผลการรันโปรแกรม

```
input score = 75
grade P
```

ผลการรันโปรแกรม

```
input score = 40
grade F
```

จากตัวอย่างที่ 3.4 นักเรียนอาจสงสัยว่าตรงเงื่อนไขที่ 2 ถ้าได้คะแนนตั้งแต่ 50 แต่ไม่ถึง 80 ได้เกรด P มีลักษณะเป็นช่วง ทำไมจึงกำหนดเงื่อนไขแค่ sc>=50 เท่านั้น ตรงนี้ให้นักเรียนนึกถึงลักษณะการทำงานของ else if ซึ่งจะทำงานจากบนลงล่าง โดยถ้าตรวจสอบเงื่อนไขใดเป็นจริงแล้ว จะทำคำสั่งที่อยู่ในวงเล็บปีกกาหลังเงื่อนไขแล้วจบการทำงานในกลุ่ม if นั้นทันที โดยไม่สนใจเงื่อนไขที่เหลืออยู่ ดังนั้นถ้าผู้ใช้กรอกคะแนนตั้งแต่ 80 ขึ้นไปจะเป็นจริงที่เงื่อนไขแรก sc>=80 ไปแล้วและแสดงผลลัพธ์เป็น grade G ออกมา ดังนั้นการที่โปรแกรมจะไปตรวจสอบถึงเงื่อนไขที่ 2 ได้ นั่นแสดงว่า คะแนนที่ตรวจสอบจะเหลือเฉพาะคะแนนที่ต่ำกว่า 80 เท่านั้น

จะเห็นได้ว่าการทำงานของ else if สามารถกำหนดค่าเป็นช่วงได้ถ้าค่าที่ตรวจสอบมีความต่อเนื่องกันตามลำดับ นักเรียนควรทำความเข้าใจกับรูปแบบการทำงานจะทำให้ประยุกต์เขียนโปรแกรมได้หลากหลายยิ่งขึ้น

สื่อมัลติมีเดียที่ 3.2

หัวข้อ

คำสั่งเงื่อนไขต่อเนื่อง

Link

http://www.sa.ac.th/c/3-2.mp4

3.3 คำสั่งเงื่อนไขเชิงซ้อน

การเขียนคำสั่งเงื่อนไข ในบางครั้งจะมีกรณีที่ต้องตรวจสอบเงื่อนไขครั้งเดียวแต่มีเงื่อนไขมากกว่าหนึ่งเงื่อนไขเพื่อให้ได้ผลลัพธ์เดียว เราสามารถใช้รูปแบบของ if ซ้อน if แบบนี้ได้

```
if(เงื่อนไข 1){
   if(เงื่อนไข 2){
      คำสั่งเมื่อเป็นจริง
   }
}
```

การใช้ if รูปแบบนี้จริงๆ แล้วก็ไม่ต่างจาก if ทั่วไป ลักษณะการทำงานของ if เมื่อเงื่อนไขเป็นจริงจะทำคำสั่งทุกอย่างที่อยู่ระหว่างวงเล็บปีกกาหลังเงื่อนไข ในกรณีนี้เมื่อตรวจสอบเงื่อนไขที่ 1 (ส่วนที่เป็นสีแดง) พบว่าเป็นจริงแล้วก็จะทำคำสั่งทั้งหมดที่อยู่ในวงเล็บปีกกาหลังเงื่อนไข (ส่วนที่เป็นสีน้ำเงิน) ซึ่งคำสั่งที่อยู่ในวงเล็บปีกกาเป็น if อีกตัวหนึ่ง ก็ใช้หลักการเดียวกับ if ทั่วไปคือถ้าเงื่อนไขที่ 2 เป็นจริงก็ทำคำสั่งที่อยู่ระหว่างวงเล็บปีกกาหลังเงื่อนไข 2 ต่อไป

ถ้านักเรียนเข้าใจหลักการใช้ if ซ้อน if แบบนี้แล้ว สามารถประยุกต์ต่อไปได้ โดยจะมีเงื่อนไขซ้อนกันกี่ชั้นก็ได้ไม่จำกัด แต่ต้องระวังเรื่องการใช้วงเล็บด้วยนะ

โปรแกรมที่ 3.4 ถ้าค่ามากกว่า 10 แต่ไม่เกิน 20 ให้แสดงคำว่า GOOD

```
#include <stdio.h>
main() {
    int a;
    printf("input number = ");
    scanf("%d",&a);
    if(a>10){
      if(a<=20){
        printf("GOOD");
      }
    }
}
```

ผลการรันโปรแกรม

```
input number = 30
```

ผลการรันโปรแกรม

```
input number = 15
GOOD
```

จากโปรแกรมที่ 3.4 กรณีแรกเมื่อเรากรอกตัวเลข 30 ตรวจสอบเงื่อนไขที่ 1 พบว่า 30 มากกว่า 10 จริง ก็จะเข้าไปตรวจเงื่อนไขที่ 2 ต่อ พบว่า 30 ไม่น้อยกว่าหรือเท่ากับ 20 เป็นเท็จจึงไม่แสดงผลลัพธ์อะไรออกมา กรณีที่ 2 เมื่อเรากรอกตัวเลข 15 ตรวจสอบเงื่อนไขที่ 1 พบว่า 15 มากกว่า 10 จริง จึงเข้าไปตรวจสอบเงื่อนไขที่ 2 ต่อพบว่า 15 น้อยกว่าหรือเท่ากับ 20 เป็นจริงเช่นกัน ดังนั้นจึงแสดงคำว่า GOOD ออกมา

รูปแบบคำสั่งเงื่อนไขเชิงซ้อน นอกจากจะใช้รูปแบบ if ซ้อน if แล้ว ยังมีอีกรูปแบบหนึ่งที่เป็นที่นิยมใช้กัน คือการใช้ตัวเชื่อมในการเชื่อมเงื่อนไขใน if เดียว ตัวเชื่อมที่ใช้ในภาษาซีมี 2 แบบดังนี้

&&	*ตัวเชื่อม "และ (and)"*
||	*ตัวเชื่อม "หรือ (or)"*

รูปแบบการใช้ตัวเชื่อมในคำสั่งเงื่อนไขเป็นดังนี้

```
if(เงื่อนไข1 ตัวเชื่อม เงื่อนไข2){
   คำสั่งเมื่อเงื่อนไขเป็นจริง
}
```

รูปแบบของคำสั่ง if โดยใช้ตัวเชื่อมก็ยังเป็นเช่นเดิมคือ ถ้าเงื่อนไขเป็นจริงจะทำคำสั่งที่อยู่ระหว่างวงเล็บปีกกาหลังเงื่อนไขให้ แต่ในกรณีนี้จะต้องดูจากเงื่อนไขทั้งหมดในวงเล็บ ซึ่งการใช้ && กับ || เป็นตัวเชื่อมนั้นจะได้ผลต่างกัน ถ้าใช้ && เป็นตัวเชื่อม เงื่อนไขต้องเป็นจริงทั้งหมดจึงจะถือว่าเป็นจริงแล้วทำคำสั่งหลังเงื่อนไข แต่ถ้าใช้ || เป็นตัวเชื่อม ขอให้เงื่อนไขใดเงื่อนไขหนึ่งเป็นจริงก็จะถือว่าเป็นจริงแล้วทำคำสั่งหลังเงื่อนไขให้ ซึ่งลักษณะการใช้ตัวเชื่อมแบบนี้จะมีวิธีคิดเหมือนกับเรื่องประพจน์ในวิชาตรรกศาสตร์ ดังนี้

p	q	p ∧ q	p ∨ q
T	T	T	T
T	F	F	T
F	T	F	T
F	F	F	F

ตัวอย่าง

T T
```
if(เงื่อนไข1 && เงื่อนไข2)
```
เงื่อนไขเป็นจริง

T F
```
if(เงื่อนไข1 && เงื่อนไข2)
```
เงื่อนไขเป็นเท็จ

T F
```
if(เงื่อนไข1 || เงื่อนไข2)
```
เงื่อนไขเป็นจริง

โปรแกรมที่ 3.5 ถ้าค่ามากกว่า 10 แต่ไม่เกิน 20 ให้แสดงคำว่า GOOD

```
#include <stdio.h>
main() {
    int a;
    printf("input number = ");
    scanf("%d",&a);
    if(a>10 && a<=20){
        printf("GOOD");
    }
}
```

ผลการรันโปรแกรม

```
input number = 30
```

ผลการรันโปรแกรม

```
input number = 15
GOOD
```

จากโปรแกรมที่ 3.5 โจทย์กำหนดว่า ค่าต้องมากกว่า 10 แต่ไม่เกิน 20 ดังนั้นตัวเชื่อมจึงต้องใช้ && กรณีแรกเมื่อเรากรอกตัวเลข 30 ตรวจสอบเงื่อนไขที่ 1 พบว่า 30 มากกว่า 10 จริง แต่เมื่อตรวจเงื่อนไขที่ 2 ต่อ พบว่า 30 ไม่น้อยกว่าหรือเท่ากับ 20 เป็นเท็จ เมื่อใช้ตัวเชื่อมเป็นและจึงไม่แสดงผลลัพธ์อะไรออกมา กรณีที่ 2 เมื่อเรากรอกตัวเลข 15 ตรวจสอบเงื่อนไขที่ 1 พบว่า 15 มากกว่า 10 จริง และตรวจสอบเงื่อนไขที่ 2 ต่อพบว่า 15 น้อยกว่าหรือเท่ากับ 20 เป็นจริงเช่นกัน ดังนั้นจึงแสดงคำว่า GOOD ออกมา

** การที่เราจะใช้ตัวเชื่อม && หรือ || ก็ขึ้นอยู่กับการวิเคราะห์เงื่อนไขของนักเรียน ถ้าเงื่อนไขนั้นจะต้องเป็นจริงทั้งหมดถึงจะแสดงผลลัพธ์ก็ให้ใช้ && เป็นตัวเชื่อม แต่ถ้าเงื่อนไขไม่ต้องเป็นจริงทั้งหมดก็ได้ ก็ให้ใช้ตัวเชื่อมเป็น ||

การใช้ตัวเชื่อมในคำสั่งเงื่อนไข จริงๆ แล้วสามารถเชื่อมเงื่อนไขได้ไม่จำกัด และสามารถใช้ทั้ง && และ || เป็นตัวเชื่อมผสมกันก็ได้ ดังนี้

```
if(เงื่อนไข1 ตัวเชื่อม เงื่อนไข2 .. ตัวเชื่อม เงื่อนไขn){
    คำสั่งเมื่อเงื่อนไขเป็นจริง
}
```

** โดยโปรแกรมจะทำการตรวจสอบเงื่อนไขจากคู่ที่เชื่อมด้วย && ก่อน จากซ้ายไปขวา แล้วตามด้วย ||

ตัวอย่างที่ 3.5

จงเขียนโปรแกรมตรวจสอบว่า ปี พ.ศ.ที่สนใจเป็นปีอธิกสุรทินหรือไม่ ซึ่งปีอธิกสุรทิน คือปี พ.ศ.ที่หารด้วย 4 ลงตัว แต่หารด้วย 100 ไม่ลงตัว หรือหารด้วย 400 ลงตัว

```
#include <stdio.h>
main(){
   int a;
   printf("input B.E. = ");
   scanf("%d",&a);
   if(a%4==0 && a%100!=0 || a%400==0){
        printf("YES");
    }
    else{
        printf("NO");
    }
}
```

ผลการรันโปรแกรม

input B.E. = 1800 ↵

NO

ผลการรันโปรแกรม

input B.E. = 2000 ↵

YES

จากตัวอย่างที่ 3.5 เมื่อเรากรอกปี 1800 ตรวจสอบเงื่อนไขพบว่า หารด้วย 4 ลงตัว และหารด้วย 100 ลงตัว เงื่อนไขคู่แรกจึงเป็นเท็จ นำไปตรวจสอบกับเงื่อนไขที่ 3 ซึ่งเชื่อมกันด้วย || พบว่า 1800 หารด้วย 400 ไม่ลงตัว ดังนั้นจึงได้ผลออกมาเป็นเท็จ จึงแสดงผลเป็น NO ในกรณีที่ 2 กรอกปี 2000 ตรวจสอบเงื่อนไขคู่แรกพบว่า หาร 4 ลงตัว และหาร 100 ลงตัวจึงได้ผลเป็นเท็จ แต่เมื่อตรวจสอบเงื่อนไขสุดท้ายพบว่า 2000 หาร 400 ลงตัวเป็นจริง ซึ่งเงื่อนไขเชื่อมกันด้วย || ดังนั้นจึงได้ผลเป็นจริง แล้วแสดงผลลัพธ์เป็น YES ออกมา

** การที่นักเรียนจะใช้คำสั่ง if ในโปรแกรมนั้น นักเรียนจำเป็นจะต้องวิเคราะห์เองว่าควรจะใช้รูปแบบไหน เพราะโจทย์จะไม่บอกเราหรอกว่า ข้อนี้ควรใช้อะไร ดังนั้นขอให้นักเรียนฝึกฝนบ่อยๆ จึงจะช่วยให้เราสามารถตัดสินใจในการเขียนโปรแกรมได้ดีขึ้น

ตัวอย่างที่ 3.6

จงเขียนโปรแกรมแปลงปีพ.ศ.และค.ศ. โดยให้เลือกว่าจะแปลงแบบไหน

1.แปลงปีพ.ศ.เป็นปีค.ศ. 2.แปลงปีค.ศ.เป็นปีพ.ศ.

```
#include <stdio.h>
main(){
  int ch,num;
  printf("1.B.E. to A.D.    2.A.D. to B.E.\n");
  printf("enter choice = ");
  scanf("%d",&ch);
  if(ch==1){
      printf("input B.E. = ");
      scanf("%d",&num);
      printf("A.D. = %d",num-543);
  }
  else if(ch==2){
      printf("input A.D. = ");
      scanf("%d",&num);
      printf("B.E. = %d",num+543);
  }
}
```

ผลการรันโปรแกรม

```
1.B.E. to A.D.    2.A.D. to B.E.
enter choice = 1
input B.E. = 2558
A.D. = 2015
```

ผลการรันโปรแกรม

```
1.B.E. to A.D.    2.A.D. to B.E.
enter choice = 2
input A.D. = 2015
B.E. = 2558
```

สื่อมัลติมีเดียที่ 3.3

หัวข้อ

คำสั่งเงื่อนไขเชิงซ้อน

Link

http://www.sa.ac.th/c/3-3.mp4