



#3 Control Flow

Statement & Block

Statement

คือ การนำ Expression หรือ Function ต่างๆ มารวมกัน โดยที่จะจบด้วย ; (Semicolon)

Block

คือ การนำ Statement หลายๆ Statement มารวมกันโดยจะอยู่ในเครื่องหมาย {} (Braces)

1. Statements and Blocks

- expression ถูกเรียกว่า statement เมื่อถูกปิดท้ายด้วยเครื่องหมาย semi-colon ;
- x = 0 ถูกเรียกว่า x = 0;
- เครื่องหมาย {} ใช้รวมส่วนของ declarations และ statements เข้าด้วยกันเป็น block
- ฟังก์ชัน, if, else, while, for มักใช้ block ในการรวม statement หลายค่าสั่งเข้าด้วยกัน
- Null statement คือ statement ที่ปราศจากส่วนของ expression โดยเหลือเพียงเครื่องหมาย semi-colon ; เพ่านั้น

Conditional

If..else

เป็นคำสั่งในการเช็คเงื่อนไขโดยจะมีรูปแบบการเขียนที่ที่แตกต่างกันไป

If

รูปแบบการใช้งานของ if ก็คือ

```
if (expression) statement;
```

```
if (expression) { statement 1; statement 2; . . . }
```

ตัวอย่างการใช้งาน

```
#include <stdio.h> int main() { int x = 20; if (x > 0) printf("%d is positive number!\n", x); }
```

โดยถ้า expression เป็นจริงโปรแกรมจะเข้าไปทำ statement ใน if แต่ถ้า expression ไม่เป็นจริงก็จะทำการข้ามไปเลย

Else

จะเป็นคำสั่งที่ใช้คู่กับ if โดยเป็นการเช็คว่าถ้าเงื่อนไขไม่ตรงตามใน if ก็จะลงมาทำใน block ของ else

```
if (expression) statement; else statement;
```

```
if (expression) { statement 1; statement 2; . . . } else { statement 3; statement 4; . . . }
```

ตัวอย่างการใช้งาน

```
#include <stdio.h> int main() { int x = 20; if (x > 0) printf("%d is positive number!\n", x); else printf("%d is negative number!\n", x); }
```

```
#include <stdio.h> int main() { int ages = 5; if (ages < 0) { printf("Invalid ages, please try again!\n"); } else { printf("Your ages is: %d\n", ages); } }
```

อย่าใช้แบบนี้

```
int foo(int x) { if (x > 5) return 1; else return 0; } int foo(int x) {  
return x > 5; }
```

Else if

else if จะเป็นคำสั่งที่ใช้สำหรับเช็คเงื่อนไขเพิ่มเติมก่อนที่จะลงไปหา else โดย else if จะมีค่าตัวก็ได้ต่างกับ if กับ else ที่มีได้แค่ตัวเดียว

Flow การเช็คเงื่อนไข if -> else if -> else

```
if (expression) statement; else if (expression) statement; else if (expression) statement; else statement;
```

```
if (expression) { statement 1; statement 2; . . . } else if (expression){ statement 3; statement 4; . . . } else if (expression){ statement 5; statement 6; . . . } else{ statement 7; statement 8; . . . }
```

ตัวอย่างการใช้งาน

```
#include <stdio.h> int main() { int ages = 5; if (ages < 0) { printf("Invalid ages, please try again!\n"); } else if(ages >= 18) { printf("You can pass!\n"); } else { printf("You can't pass!\n"); } }
```

```
#include <stdio.h> int main() { int ages; printf("Enter your ages: "); scanf("%d", &ages); if (ages >= 0) { if (ages >= 18) { printf("You can pass."); } else { printf("You can't pass"); } } else { printf("Invalid ages!"); } }
```

switch..case

จะใช้ในการเช็คเงื่อนไขเหมือนกันแต่การเช็คนั้นก็จะมี syntax ที่แตกต่างกันไป โดย if..else นั้นจะเป็นการเช็คโดยใช้ Expression แต่ switch..case จะเป็นการเช็คแบบการใช้ค่าคงที่

```
switch (expression) { case VALUE_1: // statements case VALUE_2: // statements  
break; case VALUE_3: // statements break; default: // statements }
```

```
#include <stdio.h> int main() { int c, n_vowels = 0; int n_non_vowels = 0; while ((c=getchar()) != '\n') { switch (tolower(c)) { case 'a': case 'e': case 'i': case 'o': case 'u': n_vowels++; break; default: n_non_vowels++; break; } } printf("จำนวนสระ: %d, ไม่มี สระ ชั่ว : %d\n", n_vowels, n_non_vowels); }
```

การทำงานนี้จะทำการบันลงล่างโดยการถ้า expression เราตรงกับ case ได้ก็จะ ทำไปจนเจอ break; แต่ถ้าไม่ตรงเงื่อนไขอะไรเลยก็จะลงไปทำใน default

ตัวอย่างการใช้งาน

```
#include <stdio.h> int main() { char c; switch (c = getchar()) { case 'a': case 'A': printf("Turn left\\n"); break; case 'd': case 'D': printf("Turn right\\n"); break; default: printf("Invalid input!\\n"); } }
```

Loop

While Loop

เป็นการทำซ้ำโดยจะทำงานกว่า expression จะเป็นเท็จ

```
while (expression) { . . . }
```

```
#include <stdio.h> int main() { int i = 0; while (i < 10) { printf("%d ", i); i++; } printf("\\n"); }
```

For Loop

เป็นการทำซ้ำโดยที่จะเริ่มทำ initializationStatement และจะทำ testExpression เพื่อ เช็คว่าเงื่อนไขนั้นเป็นจริงหรือไม่ถ้าหากไม่เป็นจริงจะจบ Loop ถ้าเป็นจริงก็จะไปทำ updateStatement จนกว่าจะเป็นเท็จ

```
for (initializationStatement; testExpression; updateStatement) { . . . }
```

```
#include <stdio.h> int main() { for (int i = 0; i < 10; i++) { printf("%d ", i); } printf("\\n"); }
```

```
#include <stdio.h> int main() { for (;;) { printf("hello\n"); } }
```

```
#include <stdio.h> int main() { char str[20], c; int i, str_size = sizeof(str); for (i=0; i < str_size-1 && (c=getchar()) != '\n' && c != EOF; i++) str[i] = c; str[i] = '\0'; printf("%s\n", str); } // in:  
12345678901234567890123 // out: 1234567890123456789
```

Do...While Loop

```
do { . . . } while (expression);
```

```
#include <stdio.h> int main() { int i = 0; do { printf("%d ", i); i++; } while (i < 10); printf("\n"); }
```

```
#include <stdio.h> int main() { printf("do-while loop\n"); int n = 0; do { printf("%d\n", n); n--; } while (n > 0); printf("\nwhile loop\n"); int m = 0; while (m > 0) { printf("%d\n", m); m--; } }
```

Continue

ใช้เพื่อข้ามลูปรอบนั้นๆไป โดยถ้าเป็น while หรือ do...while ก็จะข้ามไปเริ่มทำใหม่ แต่ถ้าเป็น for จะไปท่า update statement ก่อนจึงข้ามไปทำการรอบ

- continue จะบังคับการทำงานของลูปให้เกิดการวนซ้ำขึ้นมาใหม่ โดยที่ ในกรณีของ while กับ do-while การทำงานจะกระโดดไปที่การตรวจสอบเงื่อนไข (expression) ส่วนในกรณีของ for นั้น การทำงานจะกระโดดไปที่ expr3 และถึงไปตรวจสอบเงื่อนไข (expr2)

continue จะบังคับการ ค ทำ งานของลูปไป ล ห้เกิดการวน ก ช า ขึ้นมาใหม ช น โดยที่ ในกรณีของ while กับก ด do-while การทำ งานจะกระโดดไปที่การตรวจสอบ เท ท ี่ เงื่อนไข ว า (expression) ส่วนในกรณีของ for นั้นน การทำ งานจะกระโดดไปที่ expr3 ท า แล้วถึงไปตรวจสอบ เท ท ี่ เงื่อนไข ว า (expr2)

```
int i = 0; while (i < 5) { i++; if (i == 3) continue; printf("%d ", i); //  
1 2 4 5 } printf("\n");
```

```
for (int i = 1; i < 5; i++) { if (i == 3) continue; printf("%d ", i); // 1  
2 4 } printf("\n");
```

Break

ໃຫ້ສໍາអວັບກາຮຈບລຸປ່ນໆ

```
int i = 1; while (i < 5) { if (i == 3) break; printf("%d\n", i); i++; }
```

Extra

```
#include <stdio.h> int main() { int count = 0, value = 0; for (i = 0; i <= 10; i++) { count++; if (i % 2 == 0) continue; value += i; if (value > 15) break; } printf("number of loops: %d\n", count); printf("value after loops: %d\n", value); printf("i after loops: %d\n", i); }
```