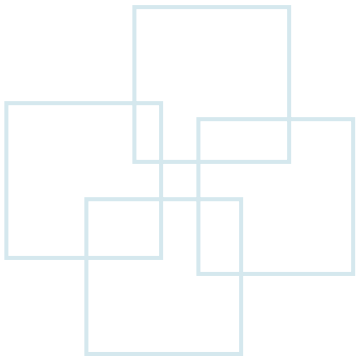


Chapter 7

Control Statements - Loop





Outline

- Structured program
- Repetition essentials
- Repetition statement
 - for
 - while
 - do ... while
- break and continue statements



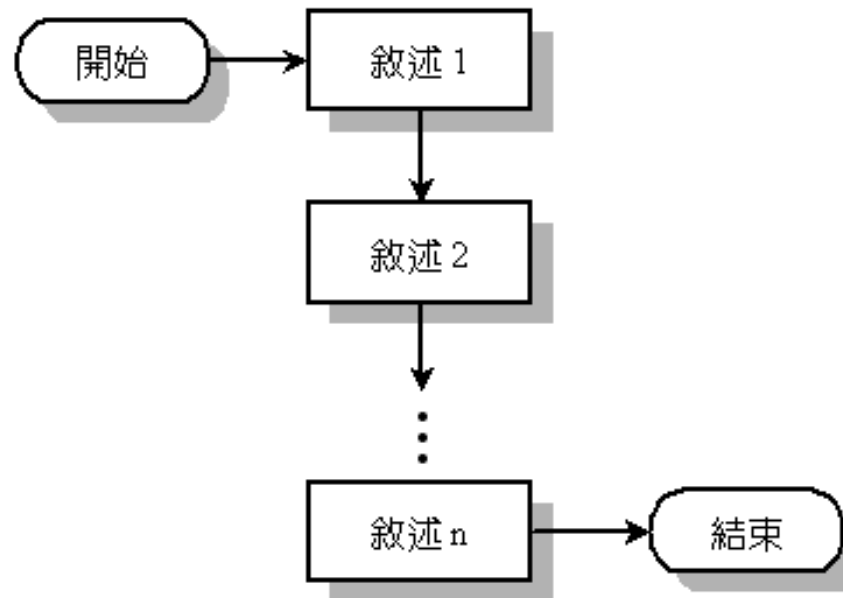
Structured Program

- Sequence structure
- Selection structure
- Iteration structure



Sequence Structure

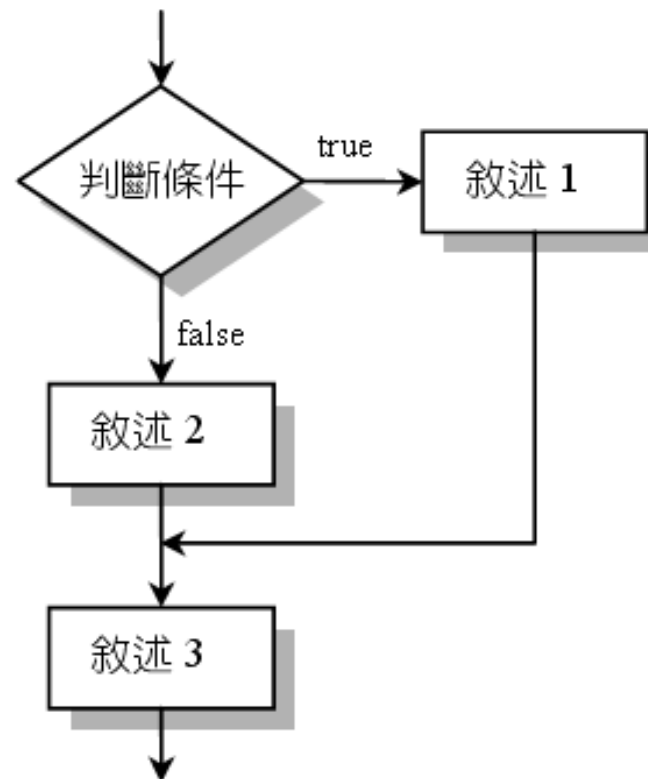
- Top-down execution.
- One-by-one statement execution





Selection Structure

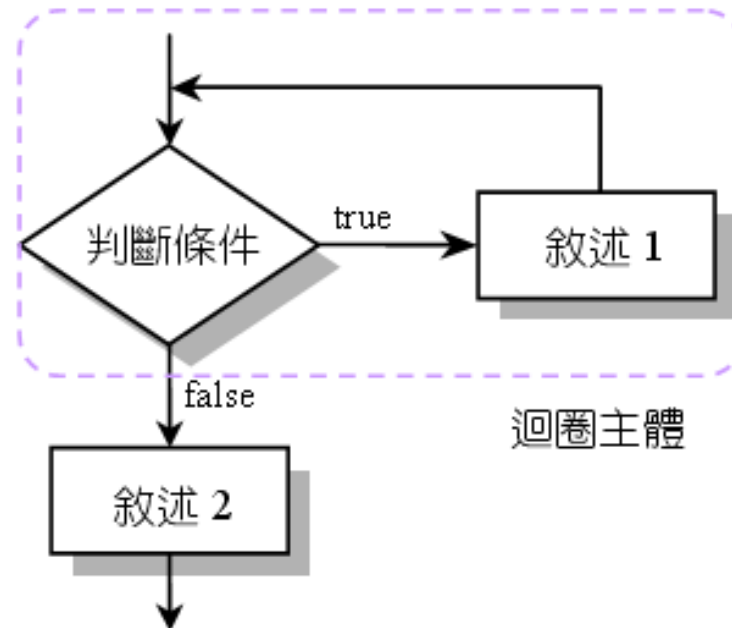
- According the condition(判斷條件) to decide which flow to execute.





Iteration Structure

- The statements in the loop body (迴圈主體) are executed repeatedly until the condition (判斷條件) is false.





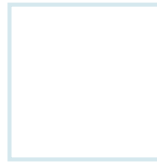
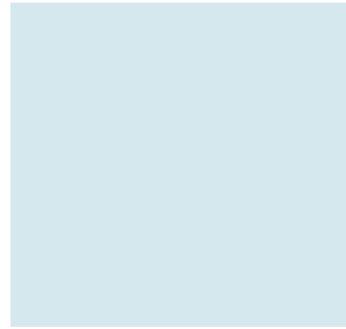
Repetition Essentials

- **Loop:** Execute a group of statements while the condition remains true.
 - **Counter-controlled repetition**
 - **Definite repetition (重覆次數已知):** know how many times loop will execute.
 - Control variable used to count repetitions.
 - **Sentinel-controlled repetition**
 - **Indefinite repetition (重覆次數未知):** the number of repetition is unknown
 - Sentinel value indicates the end of loop.

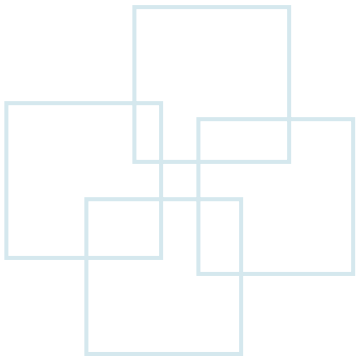


Repetition Essentials

- Counter-controlled repetition
 - Requirements
 - **Control variable** of the counter.
 - **Initial value** of the control variable.
 - **Updating** the control variable each time through the loop.
 - A **condition** that tests for the terminating condition.
 - E.g., **for**
- Sentinel-controlled repetition
 - Do not know the exact number of iterations
 - Usually need the user or input data to terminate the loop
 - **Example**: Let the user input multiple data iteratively, and input 'q' to terminate the loop.
 - E.g., **while, do... while**



for Statement





for Statement

• Format

- for (initialization; repetition condition; update) {
 statements;
}

• Example

- int cnt;
 for (cnt = 1; cnt <= 10; cnt++) {
 printf(“%d\n”, cnt);
 }

Execute the **for-loop**
statements, and then
increase cnt by 1

// print 1 ... 10

- for (int cnt = 1; cnt <= 10; cnt++) {
 printf(“%d\n”, cnt);
}

// print 1 ... 10

No semicolon is allowed



for Statement (Cont.)

- Can update the control variable in other ways.

- Example

- for (j = 0; j <= 100; j += 5) // 0, 5, 10, ..., 100

- for (j = 10; j > 0; j--) // 10, 9, 8, 7, ..., 1

- for (j = 1; j <= 100; j *= 2) // 1, 2, 4, 8, 16, ..., 64



for Statement (Cont.)

- **for** loops can usually be rewritten as **while** loops.
- Initialize and update more than one variable.
 - Can be a list separated by common (,)

– Example:

```
int i, j;
for (i = 0, j = 0; i + j <= 10; i++, j++) {
    printf("%d\n", i + j);
}
```

- Output:

- **Loop 1:** $i = 0, j = 0$, print 0
- **Loop 2:** $i = 1, j = 1$, print 2
- **Loop 3:** $i = 2, j = 2$, print 4
- **Loop 4:** $i = 3, j = 3$, print 6
- **Loop 5:** $i = 4, j = 4$, print 8
- **Loop 6:** $i = 5, j = 5$, print 10



for Statement (Cont.)

- `int i, j;`
`for (i = 0, j = 0; i + j <= 10; i++, j++) {`
 `printf(“%d\n”, i + j);`
`}`
- The above loop repeats 6 times.
- The above code will check the repetition condition 7 times (*6 true and 1 false*).
- After the above loop, the value of variable *i* is *6* and the value of variable *j* is *6*.



Infinite for Loop

• Example

– for (i = 1; i <= 100; i--)
 // i = 1, 0, -1, -2, -3, ...

– for (i = 1; i <= 100; i *= i)
 // i = 1, 1, 1, 1, ...



Sample for Problems

- Use for-loop to compute $1 + 2 + 3 + \dots + 100$
- Use for-loop to compute $1 + 3 + 5 + 7 + \dots + 99$
- Use for-loop to compute $1 + 2 + 4 + 8 + 16 + \dots + 1024$



Example of for Statement

```

01
02  #include <stdio.h>
03  #include <stdlib.h>
04  int main(void)
05  {
06      int i,sum=0;
07      for(i=1;i<=10;i++)
08          sum+=i;
09      printf("1+2+3+...+10=%d\n",sum);
10
11      system("pause");
12      return 0;
13  }

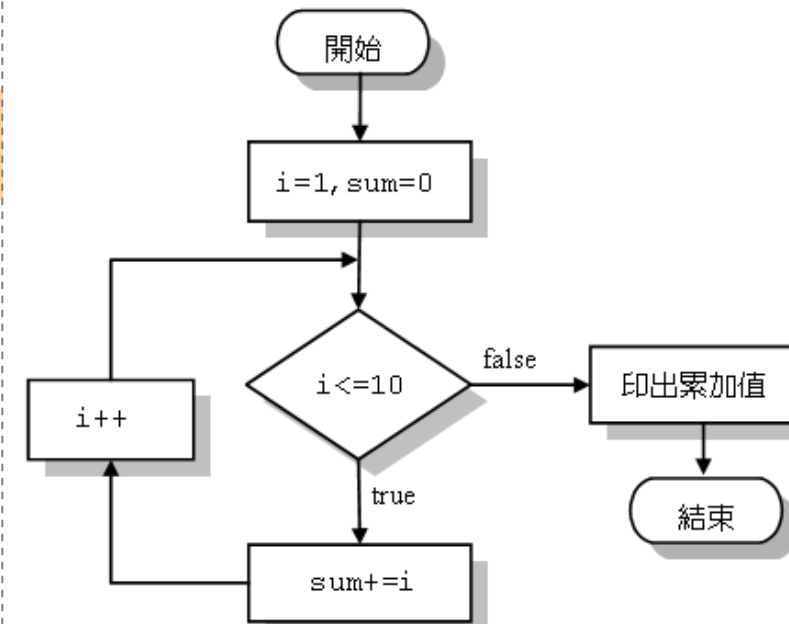
```

Sum 1 to 10

/* OUTPUT--

1+2+3+...+10=55

***/**





Example of for Statement (Cont.)

- The value of i and sum in each loop

```
06 int i, sum=0;
07 for(i=1; i<=10; i++)
08     sum+=i;
```

i 的值	sum 的值	計算 $sum+=i$ 之後，sum 的值
1	0	1
2	1	3
3	3	6
4	6	10
5	10	15
6	15	21
7	21	28
8	28	36
9	36	45
10	45	55

執行完 for 迴圈之後，
sum 的值



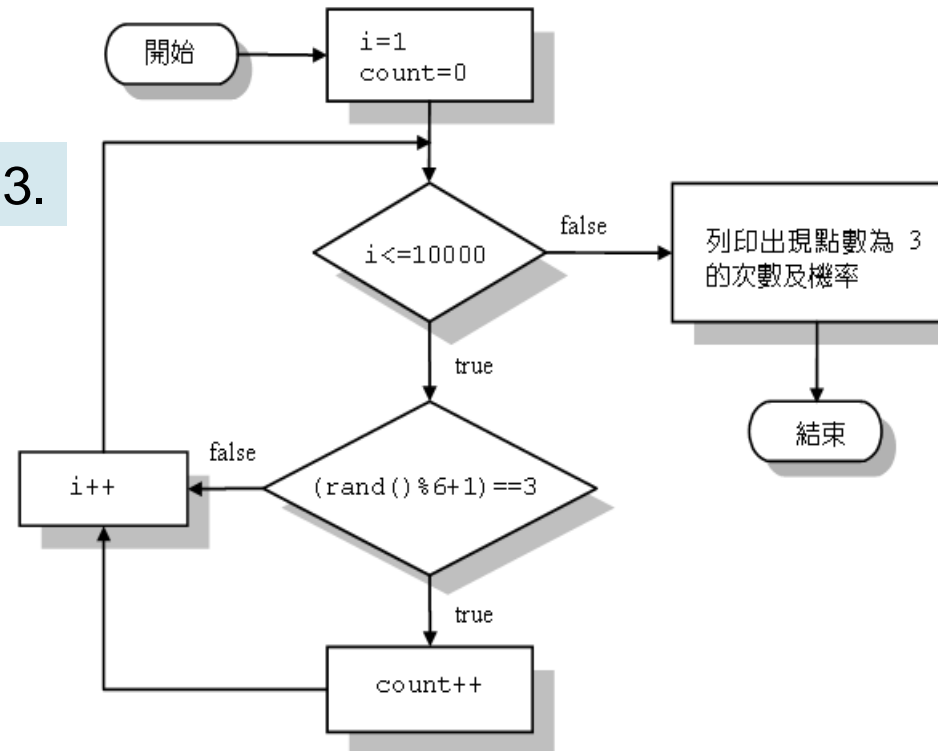
Another for Example

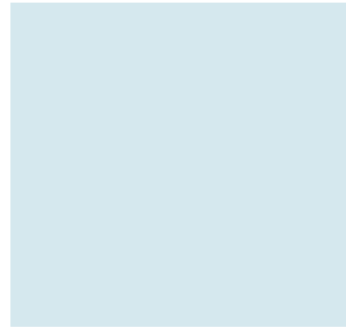
Throw a die and count the number of 3.

```

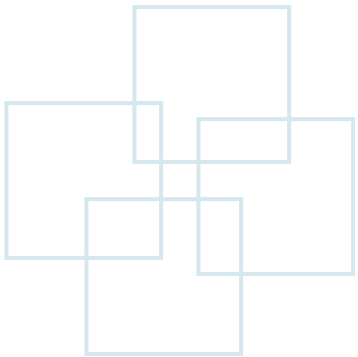
01
02 #include <stdio.h>
03 #include <stdlib.h>
04 int main(void)
05 {
06     int i, count=0;
07
08     for(i=1; i<=10000; i++)
09         if ((rand()%6+1)==3)
10             count++;
11     printf("擲 10000 次骰子時，出現 3 點的次數為%d 次\n", count);
12     printf("機率為%.3f\n", (float) count/10000);
13
14     system("pause");
15     return 0;
16 }

```





while Statement



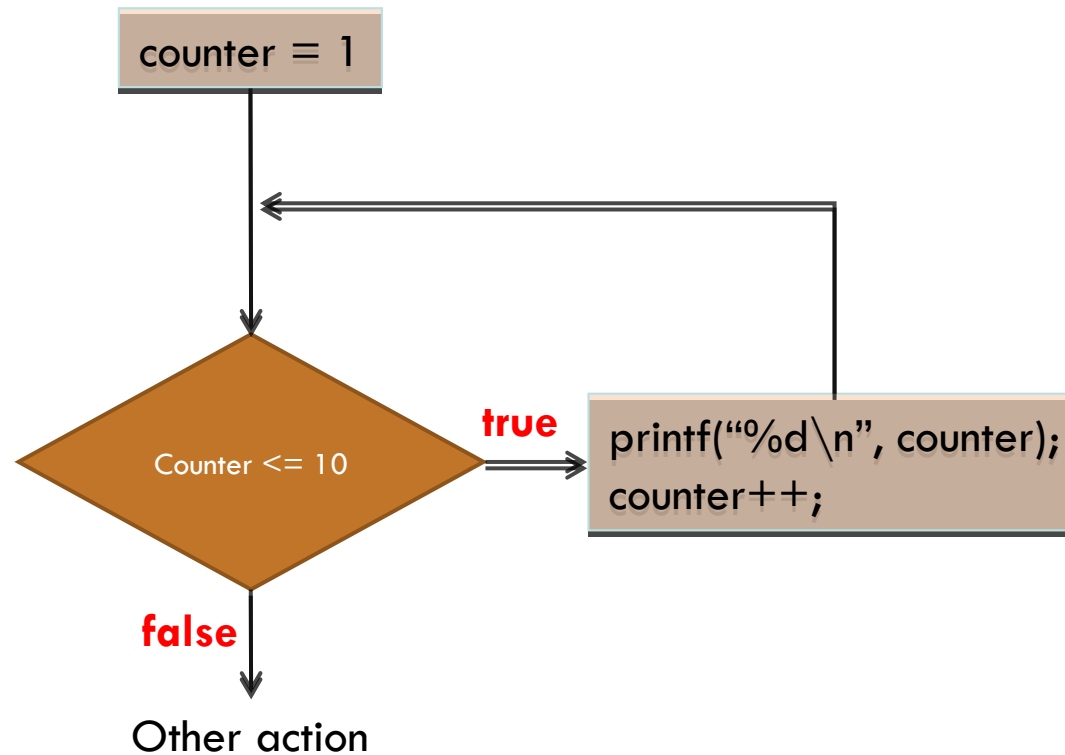


while Statement

- while (**repetition condition**) {
 statement 1;
 statement 2;
 ...
 statement n;
 update the control variable;
} **No semicolon;**
- Repeat the statements while the condition remains true.



while Statement (Cont.)





while Statement (Cont.)

- **Example 1: print 1 ... 10**

```
int counter;           // control variable
counter = 1;          // initialization
while (counter <= 10) { // repetition condition
    printf("counter is %d\n", counter);
    counter++;         // control variable updating
}
```

- **Example 2: print 1 ... 10**

```
int counter;           // control variable
counter = 0;          // initialization
while (counter < 10) { // repetition condition
    counter++;         // control variable updating
    printf("counter is %d\n", counter);
}
```



while Example

- **Example 3: print 2, 4, 6, 8, 10**

```
int counter; // control variable
counter = 2; // initialization
while (counter <= 10) { // repetition condition
    printf("counter is %d\n", counter);
    counter += 2; // control variable updating
}
```



Infinite Loop

- A sequence of instructions in a computer program which loops endlessly.
- **Example**

```
int cnt = 0;
while (cnt < 10) {
    printf("cnt = %d\n", cnt);
    cnt--;
```

Semantic error:
cnt never becomes a
value larger than 10.



Infinite Loop (Cont.)

- **Forget to update the control variable**

```
int cnt = 0;
while (cnt < 10) {
    printf("cnt = %d\n", cnt);
}
```

cnt is always equal to 0

- **Empty loop**

```
int cnt = 0;
while (cnt < 10) {
}
```



Skipping the Loop

- The initial expression is false such that the action block is never been executed.

Example

```
int cnt = 10;
while (cnt < 0) {
    printf("cnt = %d\n", cnt);
    cnt--;
}
```

Initial value of *cnt* is larger than 0, so the program never enters the iterative block.

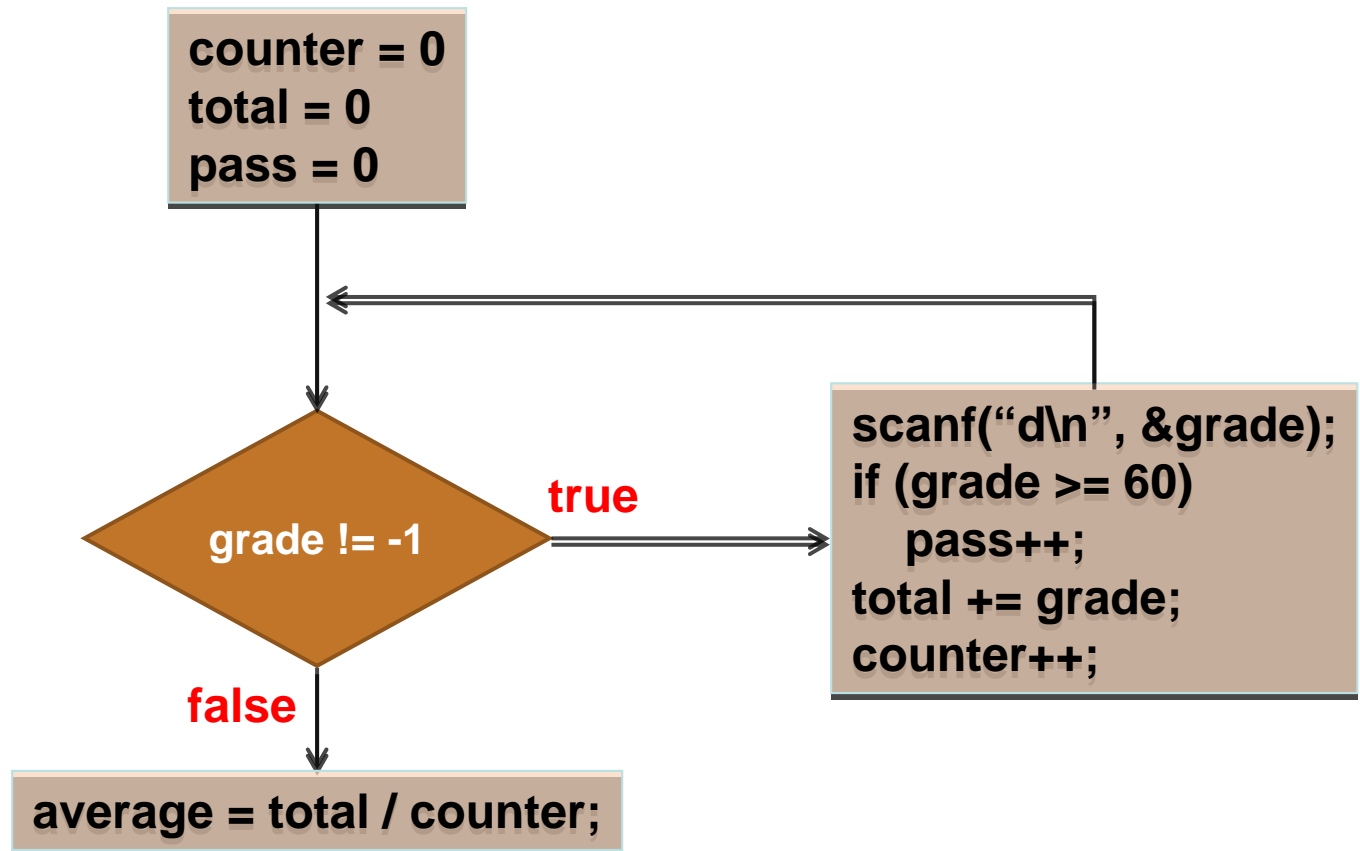


Sample while Problem

- Let the user enter the grades of students.
- Enter -1 to finish inputting.
- Compute the average of grades and count the number of students who pass the exam.



Sample while Problem (Cont.)





Another while Example

- Indefinite number of loops

```

01
02  #include <stdio.h>
03  #include <stdlib.h>
04  int main(void)
05  {
06      int i=1, sum=0;      /* 設定迴圈初值 */
07      while (sum<=100)    /* while 迴圈，當 sum 小於 100 則繼續累加 */
08      {
09          sum+=i;
10          printf("從 1 累加到%2d=%2d\n", i, sum);
11          i++;
12      }
13      printf("必須累加到%d\n", i-1);
14      system("pause");
15      return 0;
16  }

```

/* OUTPUT---

從 1 累加到 1= 1

從 1 累加到 2= 3

...

從 1 累加到 14=105

必須累加到 14

-----*/



Another while Example (Cont.)

- Infinite Loop (無窮迴圈)

```
01
02  #include <stdio.h>
03  #include <stdlib.h>
04  int main(void)
05  {
06      int i=1;
07
08      while (i > 0)      /* 當 i>0 時執行 while 迴圈的主體 */
09          printf("i=%d\n",i++);
10
11      system("pause");
12      return 0;
13  }
```

/* OUTPUT---

i=1

i=2

i=3

... (無窮迴圈的輸出)

***/**



Another while Example (Cont.)

- Utilize infinite loop

/* OUTPUT---

```
ASCII of ch=117
ASCII of ch=104
ASCII of ch=13
ASCII of ch=17
您已按了 Ctrl+q...
```

-----*/

```
01
02 #include <stdio.h>
03 #include <stdlib.h>
04 int main(void)
05 {
06     char ch;
07     while(ch!=17)          /* 當按下的鍵不是 Ctrl+q 時 */
08     {
09         ch=getch();        /* 從鍵盤取得字元 */
10         printf("ASCII of ch=%d\n",ch); /* 印出取得字元的 ASCII 碼 */
11     }
12     printf("您已按了 Ctrl+q...\n");
13
14     system("pause");
15     return 0;
16 }
```



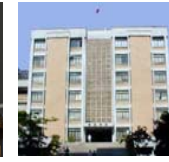
Comparison Between `for` and `while`

`for`

```
for (initialization; repetition condition; update) {  
    statements;  
}
```

`while`

```
initialization  
while (repetition condition) {  
    statements;  
    update;  
}
```

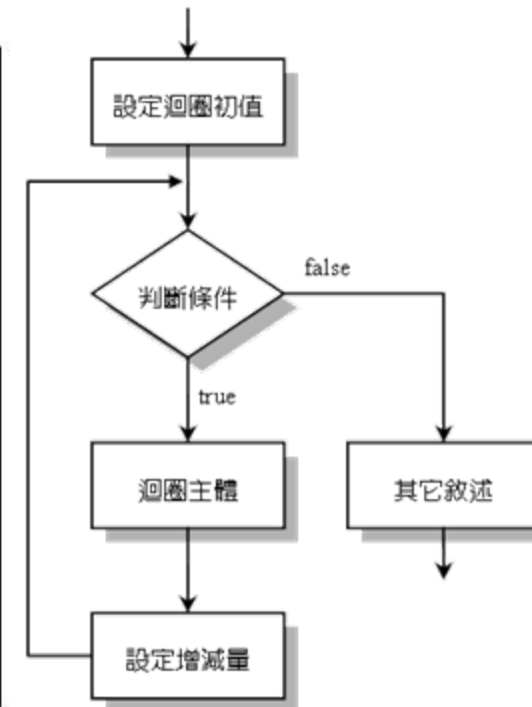
Comparison Between for and while (Cont.)

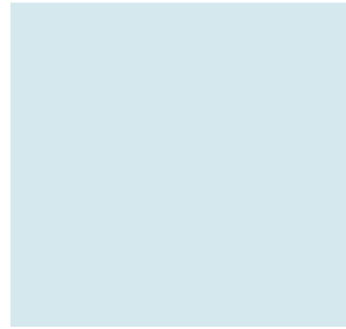
for 迴圈

```
for (設定初值; 判斷條件; 設定增減量)
{
    敘述 1;
    敘述 2;
    ⋮
    敘述 n;
}
```

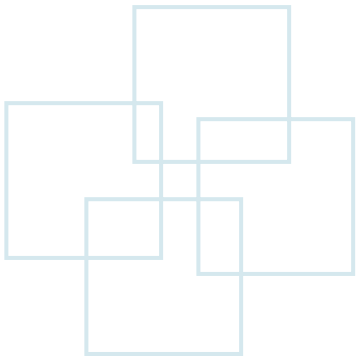
while 迴圈

```
設定初值;
while (判斷條件)
{
    敘述 1;
    敘述 2;
    ⋮
    敘述 n;
    設定增減量
}
```





do...while Statement





do-while Statement

- **Format**

initialization

```
do {  
    statements  
    update  
} while (repetition condition);
```

Put semicolon(;) here

- **Example**

```
int cnt = 1;  
do {  
    printf(“%d\n”, cnt);  
    cnt++;  
} while (cnt <= 10);
```



Comparison Between **while** and **do-while**

- **while**
 - Check the repetition before executing loop-statements.
- **do-while**
 - Enter loop-statements at least once.

Need to initialize “grade”
to a value $\neq -1$

```
grade = 0;  
while (grade != -1) {  
    scanf(“%d”, &grade);  
}
```

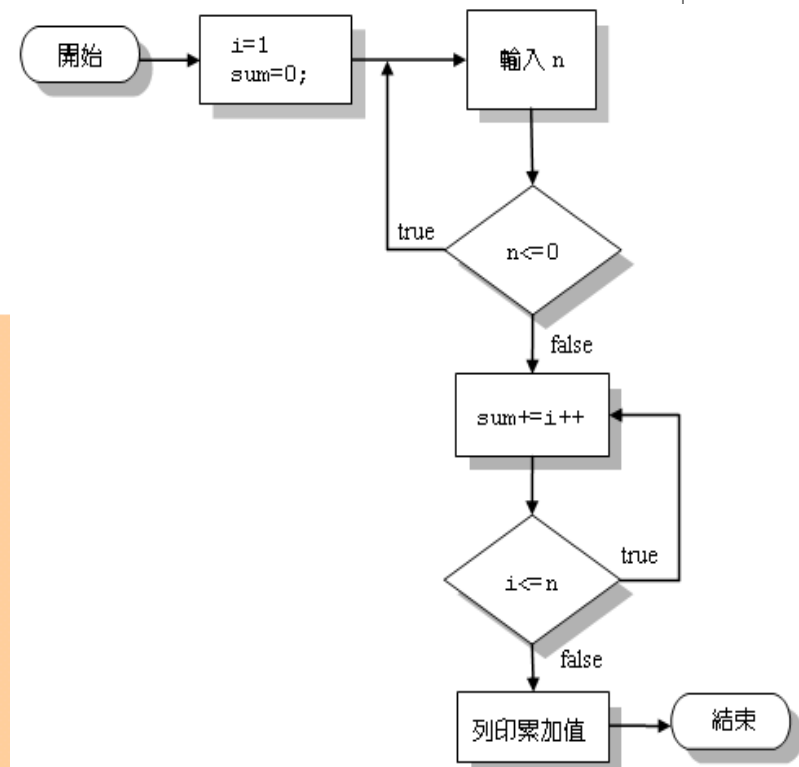
```
do {  
    scanf(“%d”, &grade);  
} while (grade != -1)
```



Example of do..while Statement

```

01
02 #include <stdio.h>
03 #include <stdlib.h>
04 int main(void)
05 {
06     int n,i=1,sum=0;
07     do
08     {
09         printf("請輸入 n 值 (n>0): ");
10         scanf("%d",&n);
11     }
12     while (n<=0);
13     do
14         sum+=i++;
15     while (i <= n);
16     printf("1+2+...+%d=%d\n",n,sum);
17     system("pause");
18     return 0;
19 }
    
```



/* OUTPUT ---

請輸入 n 值 (n>0): **-6**

請輸入 n 值 (n>0): **10**

1+2+...+10=55

***/**

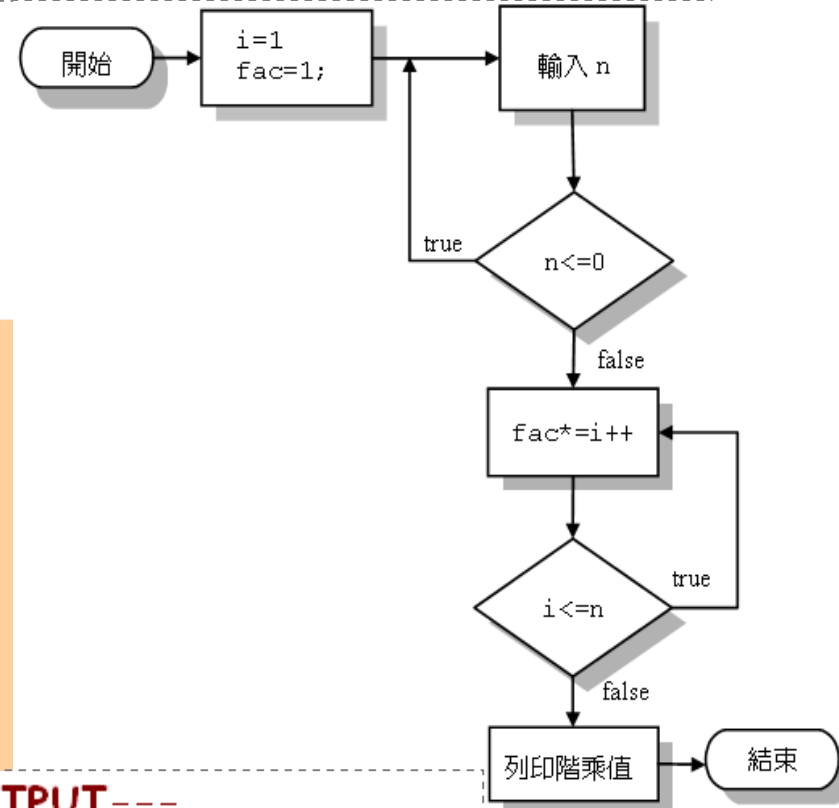


Example of do..while Statement (Cont.)

```

01
02 #include <stdio.h>
03 #include <stdlib.h>
04 int main(void)
05 {
06     int n,i=1, fact=1;
07     do
08     {
09         printf("請輸入 n 值 (n>0): ");
10         scanf("%d",&n);
11     }
12     while (n<=0);
13     do
14         fact*=i++;
15     while (i <= n);
16     printf("%d!=%d\n",n, fact);
17     system("pause");
18     return 0;
19 }

```

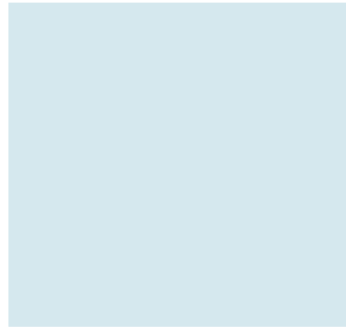


/* OUTPUT ---

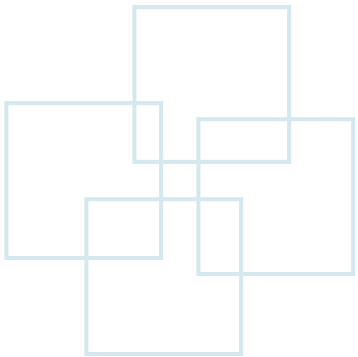
請輸入 n 值 (n>0): -3

請輸入 n 值 (n>0): 6

6!=720



Notes





Empty for Loop

- Empty is a for loop without any statement.

```
for (initialization; repetition condition; update) {  
}
```

- or

```
for (initialization; repetition condition; update);
```

Need to add
semicolon



Example of Empty for Loop

```
01
02  #include <stdio.h>
03  #include <stdlib.h>
04  int main(void)
05  {
06      int i;
07      for(i=1;i<=10000;i++);      /* 空迴圈 */
08          printf("i=%d\n",i);
09
10      system("pause");
11      return 0;
12  }
```

```
/* OUTPUT--
```

```
i=10001
```

```
-----*/
```



Nested Loops

- A loop within another loop.
- Each iteration of outer loop triggers the inner loop.

- Example

```
– for (int i = 1; i <= 10; i++) {  
    for (int j = 1; j <= 10; j++) {  
        printf("(%2d,%2d)\n", i, j);  
    }  
}
```

Use braces to include a loop

Use indentation to indicate
inner and outer loops

```
( 1, 1)  
( 1, 2)  
( 1, 3)  
( 1, 4)  
...  
( 1,10)  
( 2, 1)  
( 2, 2)  
...  
(10, 9)  
(10,10)
```

- The above code will repeat 100 times (print 100 lines).



Nested Loops (Cont.)

- Print out the 9x9 timetable

```

01
02 #include <stdio.h>
03 #include <stdlib.h>
04 int main(void)
05 {
06     int i,j;
07     for (i=1;i<=9;i++)      /* 外層迴圈 */
08     {
09         for (j=1;j<=9;j++)  /* 內層迴圈 */
10             printf("%d*%d=%2d  ",i,j,i*j);
11         printf("\n");
12     }
13     system("pause");
14     return 0;
15 }

```

/* OUTPUT-----

```

1*1= 1  1*2= 2  ...  1*9= 9
2*1= 2  2*2= 4  ...  2*9=18
...
9*1= 9  9*2=18  ...  9*9=81

```

-----*/

} 內層迴圈 } 外層迴圈

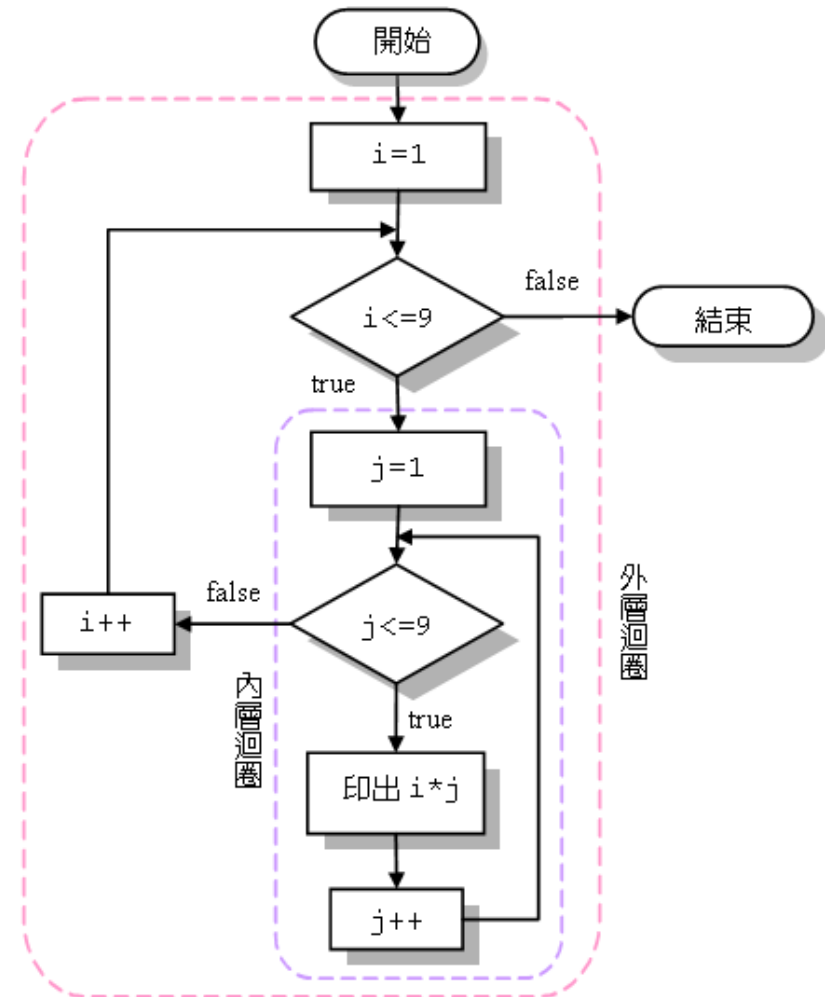


Nested Loops (Cont.)

- Flowchart of printing out the timetable

```

for (i=1;i<=9;i++)      /* 外層迴圈 */
{
    for (j=1;j<=9;j++)  /* 內層迴圈 */
        printf("%d*%d=%2d ",i,j,i*j);
    printf("\n");
}
  
```





Nested Loops with while Statement

```
01
02  #include <stdio.h>
03  #include <stdlib.h>
04  int main(void)
05  {
06      int i=1, j=1;      /* 設定迴圈控制變數的初值 */
07      while (i<=9)      /* 外層迴圈 */
08      {
09          while (j<=9)  /* 內層迴圈 */
10          {
11              printf("%d*%d=%2d  ",i,j,i*j);
12              j++;
13          }
14          printf("\n");
15          i++;
16          j=1;
17      }
18      system("pause");
19      return 0;
20  }
```

內層迴圈

外層迴圈



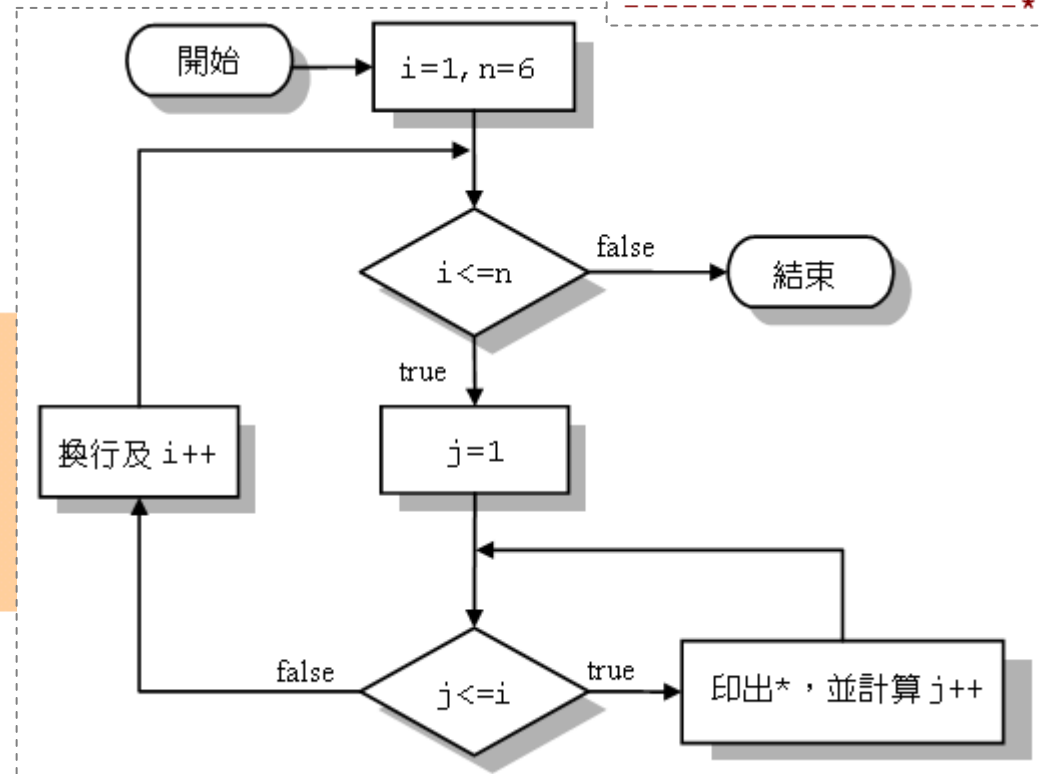
Another Example of Nested Loop

- Use nested for loops to print a triangle

/* OUTPUT--

```
*
**
***
****
*****
*****
*****
*/
```

```
01
02 #include <stdio.h>
03 #include <stdlib.h>
04 int main(void)
05 {
06     int i, j, n=6;
07
08     for (i=1; i<=n; i++)
09     {
10         for (j=1; j<=i; j++)
11             printf("*");
12         printf("\n");
13     }
14
15     system("pause");
16     return 0;
17 }
```





Another Example of Nested Loop (Cont.)

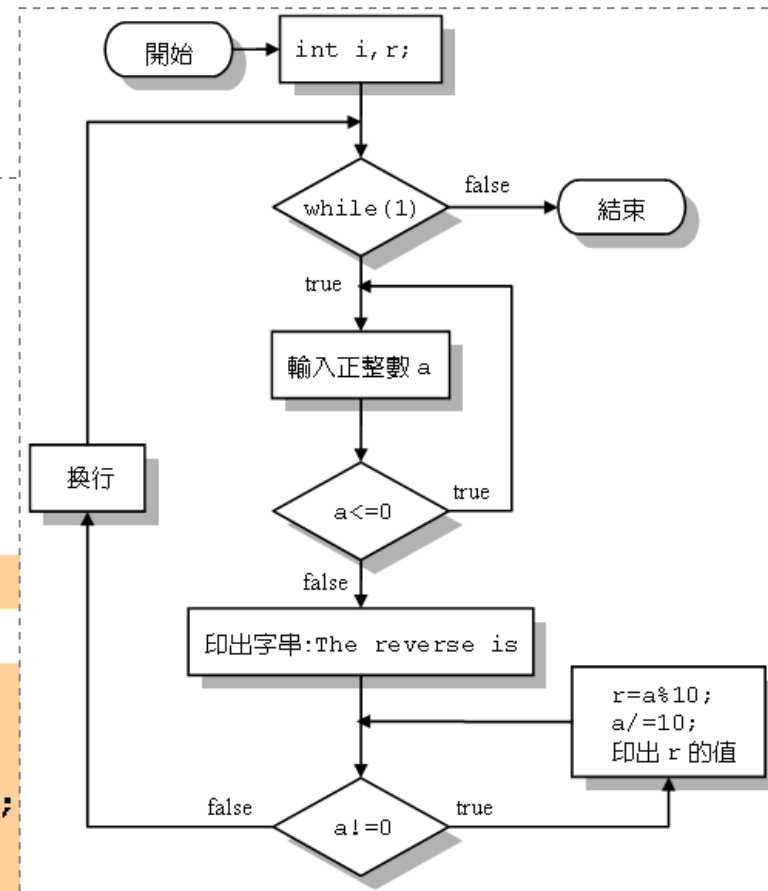
- Inverse the digits of an integer.

E.g., **5123** → **3215**

```

01
02 #include <stdio.h>
03 #include <stdlib.h>
04 int main(void)
05 {
06     int a,r;
07
08     while(1)
09     {
10         do
11         {
12             printf("Input an integer:");
13             scanf("%d",&a);
14         }
15         while (a<=0);          /* 必須輸入大於 0 的正整數 */
16

```





Another Example of Nested Loop (Cont.)

- Inverse the digits of an integer (Cont.)
E.g., **5123** → **3215**

```

17     printf("The reverse is ");
18     while (a!=0)
19     {
20         r=a%10; /* 計算 a/10 的餘數 */
21         a/=10; /* 計算 a/10，再設回給 a */
22         printf("%d", r);
23     }

```

```

24     printf("\n\n");
25 }
26 system("pause");
27 return 0;
28 }

```

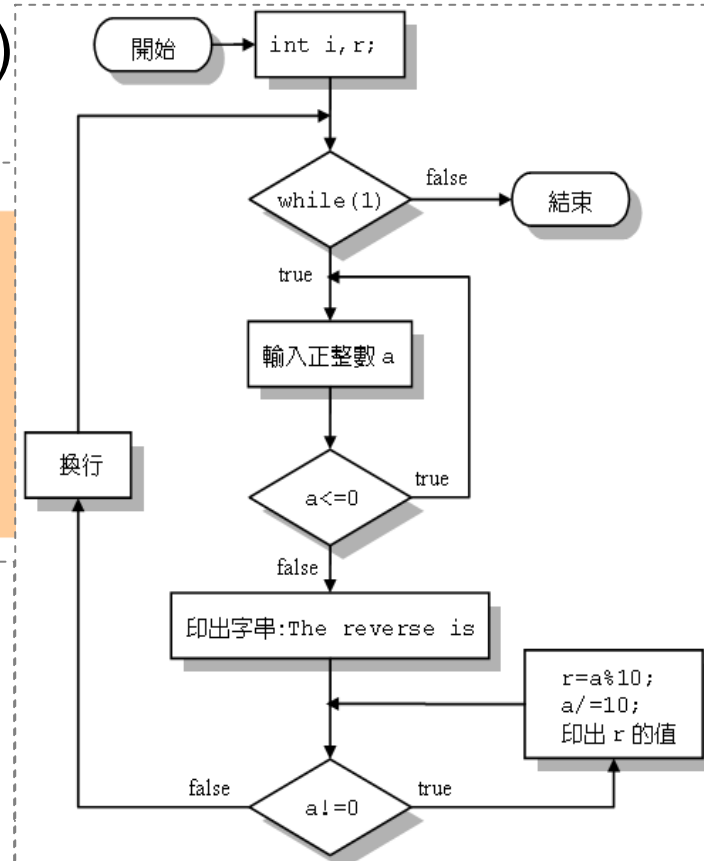
/* OUTPUT----

Input an integer: **-58**
 Input an integer: **13579**
 The reverse is 97531

Input an integer: **2468**
 The reverse is 8642

Input an integer:

-----*/





break Statement

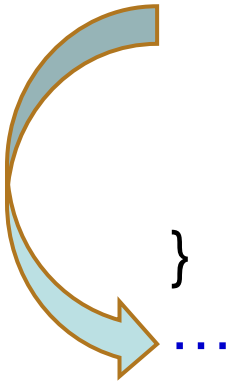
- **break** statement causes immediate exit from a **while**, **for**, **do...while** or **switch** statement.
- Program execution continues with the first statement after the structure.
- Common uses of the **break** statement
 - Escape early from a loop.
 - Skip the remainder of a switch statement.



break Statement (Cont.)

- **Example**

```
for (initialization; repetition condition; update) {  
    statement 1;  
    ...  
    break;  
    statement n;  
    ...  
}
```





break Statement (Cont.)

- *Escape from nested loops*

```
for(initialization; repetition condition; update) {  
    ...  
    for(initialization; repetition condition; update) {  
        statement 1;  
        ...  
        break;  
        statement n;  
        ...  
    }  
    ...  
}
```

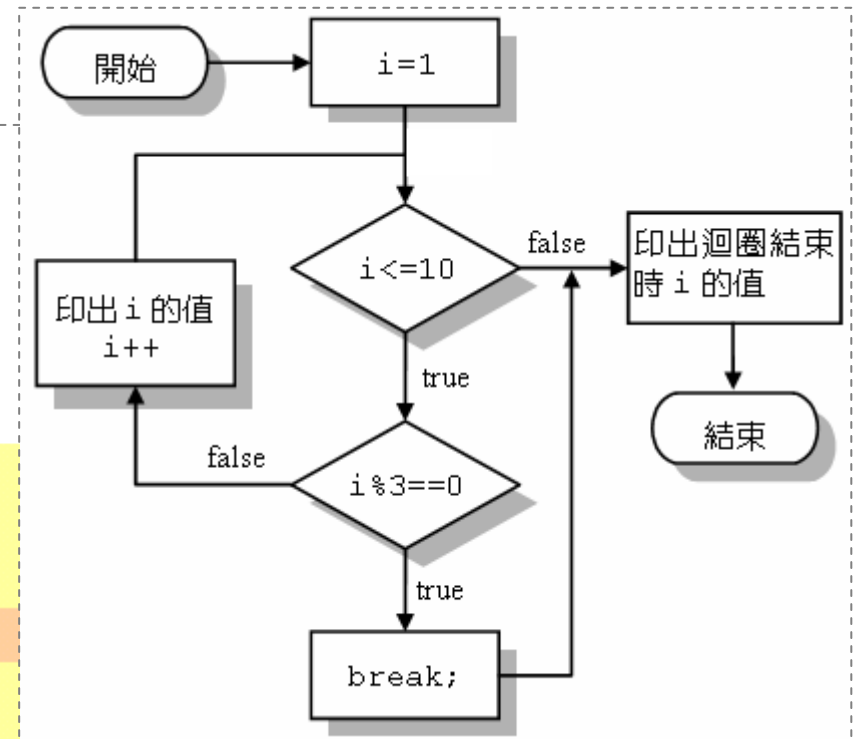
Skip from the nearest loop



Example of break Statement

```

01
02 #include <stdio.h>
03 #include <stdlib.h>
04 int main(void)
05 {
06     int i;
07     for(i=1;i<=10;i++)
08     {
09         if(i%3==0)
10             break; /* 跳離迴圈 */
11         printf("i=%d\n",i);
12     }
13     printf("跳離迴圈時, i=%d\n",i);
14
15     system("pause");
16     return 0;
17 }
    
```



/* OUTPUT--

i=1

i=2

跳離迴圈時, i=3

***/**



continue Statement

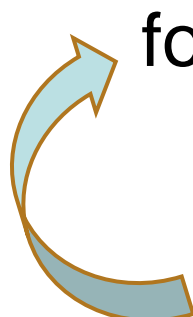
- Skip the remaining statements in the body of a **while**, **for** or **do-while** statement.
 - Proceed to the next iteration of the loop.
- **while** and **do-while**
 - Loop-continuation test is evaluated immediately after the continue statement is executed.
- **for**
 - Increment expression is executed, then the loop-continuation test is evaluated.



continue Statement (Cont.)

- *Example*

```
for (initialization; repetition condition; update) {  
    statement 1;  
    ...  
    continue;  
    statement n;  
    ...  
}  
...
```





continue Statement (Cont.)

- **Example of nested loops**

```
for(initialization; repetition condition; update) {
```

```
...
```

```
for(initialization; repetition condition; update) {
```

```
statement 1;
```

```
...
```

```
continue;
```

```
statement n;
```

```
...
```

```
}
```

```
...
```

```
}
```

Go back to the beginning
of the nearest loop

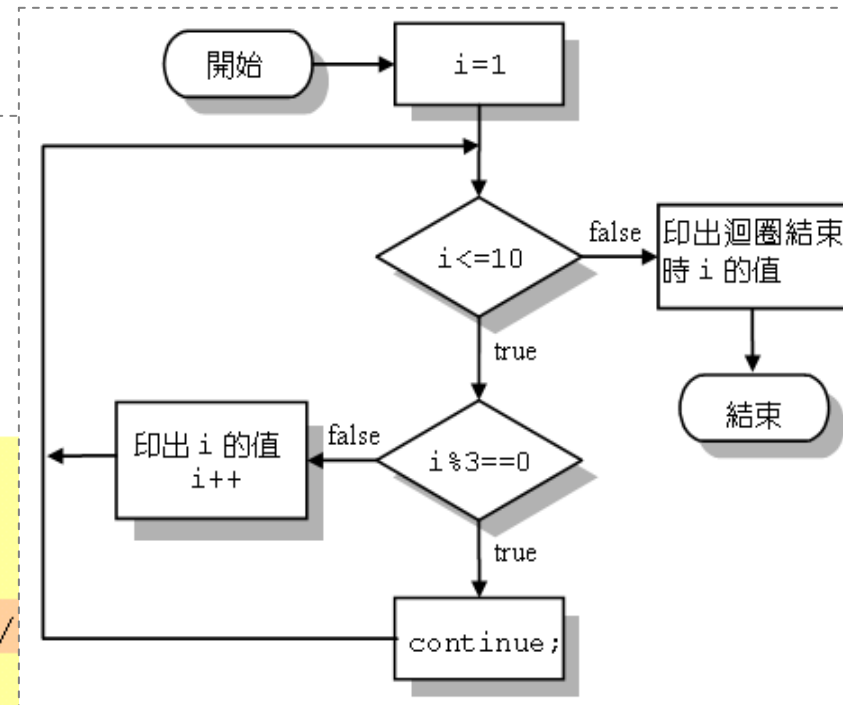


Example of continue Statement

```

01
02 #include <stdio.h>
03 #include <stdlib.h>
04 int main(void)
05 {
06     int i;
07     for(i=1;i<=10;i++)
08     {
09         if(i%3==0)
10             continue; /* 回到起始處執行 */
11         printf("i=%d\n",i);
12     }
13     printf("跳離迴圈時, i=%d\n",i);
14
15     system("pause");
16     return 0;
17 }

```



/* OUTPUT--

```

i=1
i=2
i=4
i=5
i=7
i=8
i=10
跳離迴圈時, i=11

```

-----*/



Lab 07-1

- Use `do...while` to print out the 9x9 timetable.
- Use `for` statement to calculate the sum of $2 + 4 + \dots + n$, where n is an even number.
Note: n is an integer derived through `scanf()`.
- Print out the values (between 1 and 100) that can be divided by 7 and 3 at the same time.
- Suppose there is a rope of length 1000 inches. You cut out half of the rope per day. Print out how many days later the length of the rope would be shorter than 5 inches.
- Let users input an integer from the keyboard and print out whether the integer is a prime number or not.



Lab 07-2

- 試撰寫一無窮迴圈，利用 `break` 敘述來撰寫4個數字之密碼輸入的過程。使用者有三次輸入機會，並須滿足下列的條件(預設密碼為 0000)：
 - 如果密碼輸入不對，則會再次出現 "請輸入密碼:" 字串。
 - 如果三次的輸入都不對，則程式會印出 "密碼輸入超過三次!!" 字串，然後結束程式的執行。
 - 如果輸入正確，則印出 "密碼輸入正確，歡迎使用本系統!!" 字串。
- 試利用 `continue` 敘述，找出小於100的整數裡，所有可以被2與3整除，但不能被12整除的整數。