C++基礎語法 Unit-5

- 迴圈 (while)
- 進階迴圈控制 (continue/break)

while 迴圈

常用於不確定圈數的重複結構

【範例】while 迴圈

【Input】正整數 N
【Output】5 的倍數中，
不小於 N 的最小值

$\geq N$

0, 5, 10, 15, 20, 25, 30, 35, 40, 45, 50, … …

N = 37

```cpp
#include <iostream>
using namespace std;

int main() {
    int N;
    cin >> N;
    int ans = 0;
    while (ans < N) {
        ans += 5;
    }
    cout << ans << "\n";
    return 0;
}
```

【試試看】改用 for 迴圈

Example 5-7B

```cpp
#include <iostream>
using namespace std;

int main() {
    int N;
    cin >> N;

    int ans = 0;

    for (; ans < N; ans += 5) {
    }

    //for (; ans < N; ans += 5);

    cout << ans << "\n";
    return 0;
}
```

【範例】輸出 N^2 (N 的平方)

【Input】多筆測資。

每筆輸入一個正整數 N。

（當 N 為 0 時，結束程式。）

【Output】輸出 N^2 (N 的平方)。

```cpp
#include <iostream>
using namespace std;

int main() {
    int N;

    while (cin >> N) {
        if (N == 0) break;
        cout << N * N << "\n";
    }

    return 0;
}
```

【試試看】改用 for 迴圈

Example 5-1B

```cpp
#include <iostream>
using namespace std;

int main() {
    int N;

    for (cin >> N; N != 0; cin >> N) {
        cout << N * N << "\n";
    }

    return 0;
}
```

【練習】計算 1 加到 100 的和 (使用 while)

Example 5-8

```cpp
#include <iostream>
using namespace std;

int main() {
    int i = 1, ans = 0;
    while (i <= 100) {
        ans += i;
        i++;
    }
    cout << ans << "\n";
    return 0;
}
```

https://pastebin.ubuntu.com/p/PQCC24nfsQ/

【練習】取出整數的每一位數

【輸入】一個正整數 N
計算 S = (N 的每一「位數」的「數字」總和)

【輸出】
如果 N 可以被 S 整除，輸出 Yes，否則輸出 No

【例1】
N = 12，
S = 1 + 2 = 3，
12 / 3 = 4 … 0 → 答案為 Yes

【例2】
N = 1234，
S = 1 + 2 + 3 + 4 = 10，
1234 / 10 = 123 … 4 → 答案為 No

N = 1234

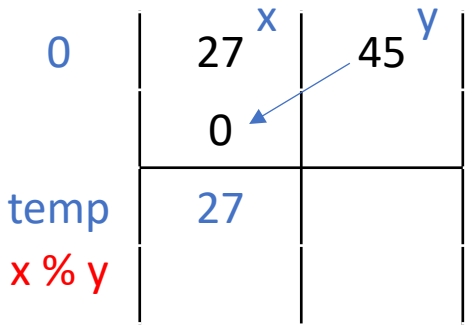
temp	temp / 10	temp % 10	
1234	123	4	個位數
123	12	3	十位數
12	1	2	百位數
1	0	1	千位數
0			

https://pastebin.ubuntu.com/p/ph4GNKc3Qv/

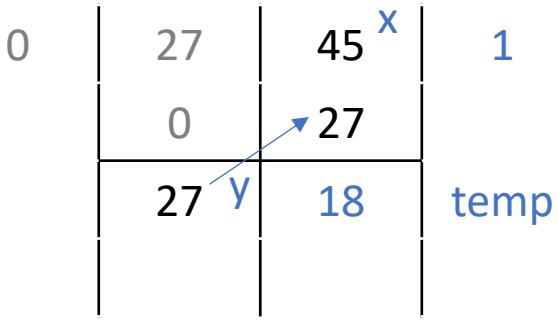
【練習】計算 x 和 y 的最大公因數 (GCD)

GCD = Greatest Common Divisor

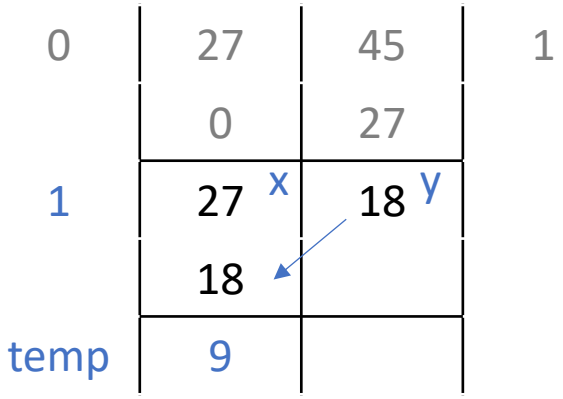
1



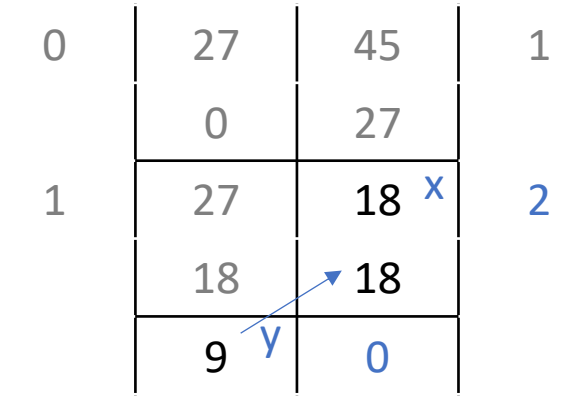
2



3



4



【練習】計算 x 和 y 的最大公因數 (GCD)

Example 5-9

```cpp
#include <iostream>
using namespace std;

int main() {
    int x, y;
    cin >> x >> y;

    while (x % y != 0) {
        int temp = x % y;
        x = y;
        y = temp;
    }
    cout << y << "\n";
    return 0;
}
```

https://pastebin.ubuntu.com/p/MrNngCMFGR/

當讀入測資為 0 時，結束程式

Example 5-1

【作業】 d070: 格瑞哥里的煩惱 (0 尾版)

```
7       while (cin >> N) {
8           if (N == 0) break;
9           cout << N * N << "\n";
10      }
```

```
7       while (cin >> N && N != 0) {
8           //if (N == 0) break;
9           cout << N * N << "\n";
10      }
```

```
7       while (cin >> N && N) {
8           //if (N == 0) break;
9           cout << N * N << "\n";
10      }
```

當遇到EOF (End of File)時，結束程式

Input from a terminal never really "ends" (unless the device is disconnected), but it is useful to enter more than one "file" into a terminal, so a key sequence is reserved to indicate end of input. In UNIX the translation of the keystroke to EOF is performed by the terminal driver, so a program does not need to distinguish terminals from other input files. By default, the driver converts a Control–D character at the start of a line into an end–of–file indicator. To insert an actual Control–D (ASCII 04) character into the input stream, the user precedes it with a "quote" command character (usually Control–V). AmigaDOS is similar but uses Control–\ instead of Control–D.

In DOS and Windows (and in CP/M and many DEC operating systems such as RT–11 or VMS), reading from the terminal will never produce an EOF. Instead, programs recognize that the source is a terminal (or other "character device") and interpret a given reserved character or sequence as an end–of–file indicator; most commonly this is an ASCII Control–Z, code 26. Some MS–DOS programs, including parts of the Microsoft MS–DOS shell (COMMAND.COM) and operating–system utility programs (such as EDLIN), treat a Control–Z in a text file as marking the end of meaningful data, and/or append a Control–Z to the end when writing a text file.

參考資料：https://en.wikipedia.org/wiki/End-of-file

當遇到EOF (End of File)時，結束程式

Example 5-2

【作業】 d071: 格瑞哥里的煩惱 (EOF 版)

標準輸入輸出，模擬EOF：

- Linux / MacOS: Ctrl-D
- Windows: Ctrl-Z

或者，Ctrl-C 通常會終止接受來自控制台輸入的程序

do while 迴圈

do-while 迴圈至少會執行一次

【範例】計算 1 加到 100 的和

Example 5-10

```cpp
#include <iostream>
using namespace std;

int main() {
    int i = 1, sum = 0;
    do {
        sum += i;
        i++;
    } while (i <= 100);
    cout << sum << "\n";
    return 0;
}
```

換成 for 或 while 試試看！

continue / break

continue：忽略後續的執行敘述，繼續下一圈

break：提早結束迴圈

continue / break

- continue：忽略後續的執行敘述，繼續下一圈
- break：提早結束迴圈

- 可用來終止無窮迴圈
  - `while (1) { … … }`
  - `while (true) { … … }`

- 也可搭配 `for` 或 `do-while` 使用
  - `for ( ; ; ) { … … }`
  - `do { … … } while ();`

【範例】break

Example 5-11

【Input】一個正整數 N

【Output】判斷 N 是質數或合數

```cpp
#include <iostream>
using namespace std;

int main() {
    int N;
    while (cin >> N) {
        bool prime = true;
        for (int i = 2; i * i <= N; i++) {
            if (N % i == 0) {
                prime = false;
                break;
            }
        }
        if (prime) {
            cout << "質數\n";
        } else {
            cout << "合數\n";
        }
    }
    return 0;
}
```

【範例】continue

Example 5-12

【Input】連續輸入 N 個整數

【Output】輸入的數字中，
所有正整數加總的和

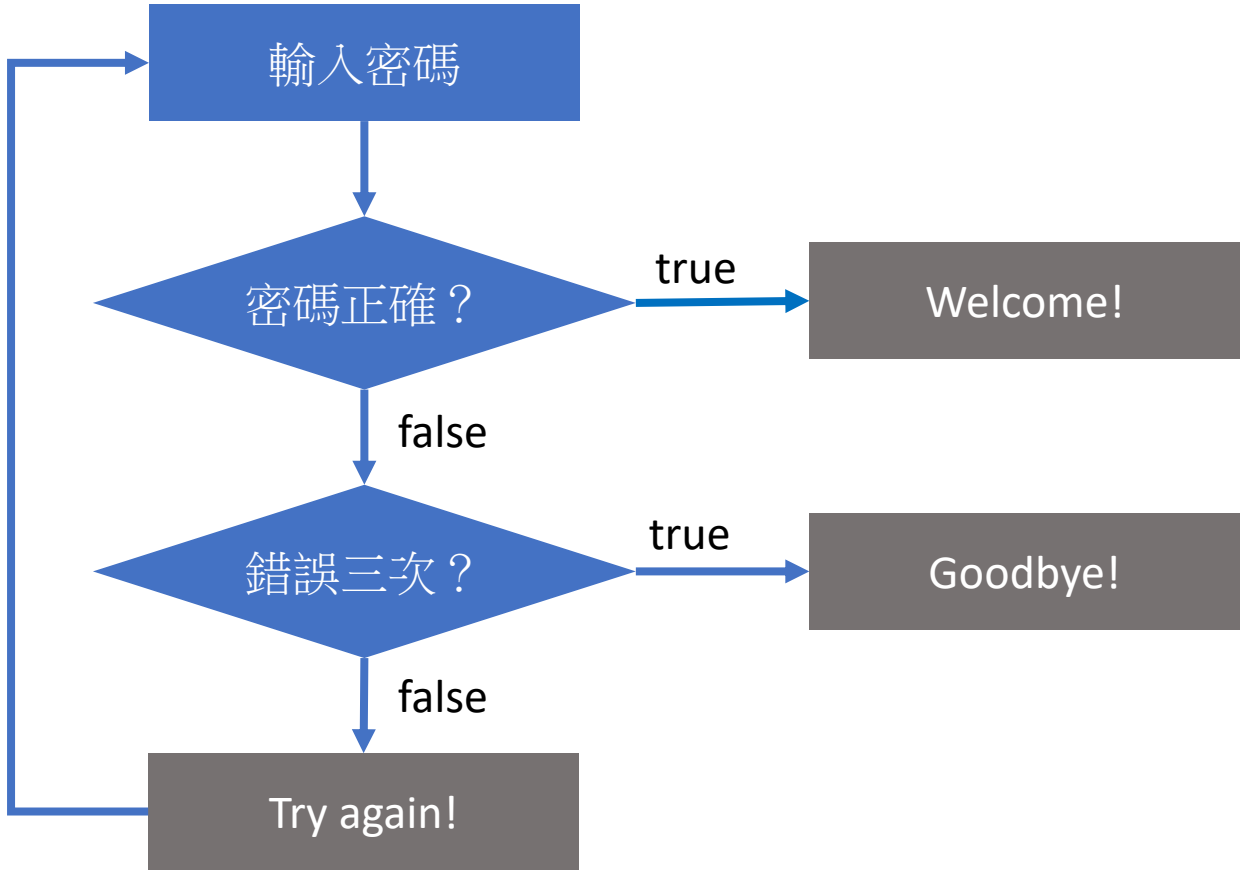
移除continue的話，
該怎麼改寫？

```cpp
#include <iostream>
using namespace std;

int main() {
    int N, x, sum = 0;
    cin >> N;
    for (int i = 0; i < N; i++) {
        cin >> x;
        if (x < 0) {
            continue;
        }
        sum += x;
    }
    cout << sum << "\n";
    return 0;
}
```

https://pastebin.ubuntu.com/p/wP6vZBdVxv/

【範例】檢查密碼

- 密碼是任意長度的字串，也可能包含空白字元
  - 無法用 cin 讀取
  - 改用 getline(cin, s)
- 檢查輸入的密碼是否正確
  - 最多只能錯三次，否則輸出Goodbye!
  - 密碼正確時，輸出Welcome!
  - 密碼錯誤時，輸出Try again!



https://pastebin.ubuntu.com/p/FkzMWF5q5v/

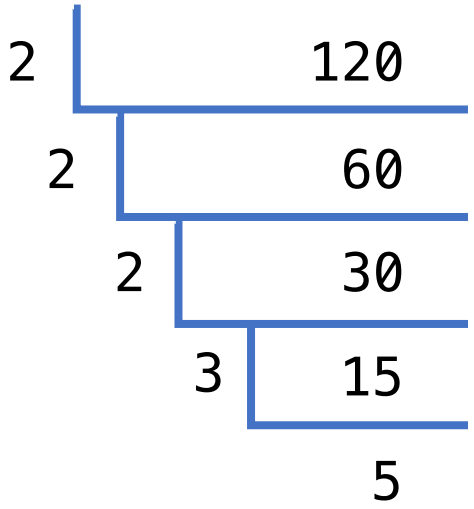
```cpp
#include <iostream>
using namespace std;

int main() {
    string password = "I love C++.";
    string s;
    int cnt = 0;
    while (++cnt <= 3 && getline(cin, s)) {
        if (s == password) {
            cout << "Welcome!\n";
            break;
        } else {
            if (cnt < 3) {
                cout << "Try again!\n";
            } else {
                cout << "Goodbye!\n";
            }
        }
    }
    return 0;
}
```

密碼可能含空白字元，須改用 getline( ) 整行讀取


```
//如果程式中混用cin 和getline，
//cin 之後，需加一行dummy的getline，把緩衝區中殘留的"\n"清空。

int N;
string s;

cin >> N;
cout << "\n";

getline(cin, s); // 此處的 s，只讀到一個換行符號
cout << "s : " << s << "\n";

getline(cin, s);
cout << "s : " << s << "\n";
```

https://pastebin.ubuntu.com/p/RH4W8ZTYbs/

【練習】作業 a010: 因數分解



列印格式：`20 = 2^3 * 3 * 5`

https://pastebin.ubuntu.com/p/H7hK8b4xgt/

注意遞增/遞減運算字 (++ 或 --) 的位置

Example 5-5

先判斷真假，再遞減

```cpp
#include <iostream>
using namespace std;

int main() {
    int T; // 共有 T 組測試資料
    cin >> T;
    cout << "==================\n";

    while (T--) {
        cout << T << "\n";

        /*
         處理每一組測試資料的程式碼
         */
    }
    return 0;
}
```

先遞減，再判斷真假

```cpp
#include <iostream>
using namespace std;

int main() {
    int T; // 共有 T 組測試資料
    cin >> T;
    cout << "==================\n";

    while (--T) {
        cout << T << "\n";

        /*
         處理每一組測試資料的程式碼
         */
    }
    return 0;
}
```