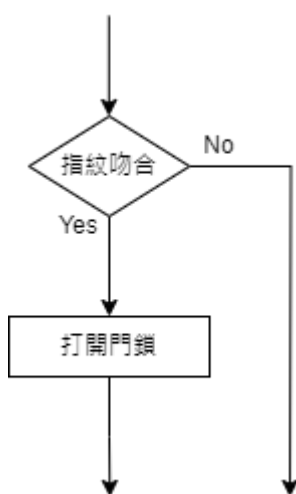


3.1 if ... else ...

每一行程式碼都會執行到？

我們寫的每一行程式碼都有用嗎？當然有用。那每一行都會被執行嗎？這要看情況。

之前我們寫的程式，會從 `main()` 函數的第一行開始一行一行依序執行下去，直到程式結束。但在真實世界運作的程式是要有彈性的，例如：指紋鎖必須要在使用者指紋與內部設定吻合時，才會開鎖，否則什麼都不做。也就是這樣一個結構：



接下來我們我們以「計算絕對值」為例，來看看這種結構。

if 敘述

練習：絕對值

讀入使用者輸入的整數 `a`，計算並輸出其絕對值 `|a|`。

絕對值表示數線上原點到該數值的距離，所以若 `a ≥ 0` 則 `a` 的值就是其絕對值，否則將 `a` 的值乘上 `-1` 才是其絕對值。

```
#include <iostream>

using namespace std;

int main()
{
    int a;

    cin >> a;
```

```
    if(a<0)
    {
        a = -1*a;
    }

    cout << "|a|=" << a << endl;

    return 0;
}
```

在這裡我們使用到 if 敘述，它的語法如下：

```
if( 條件判斷式 )
{
    條件成立時要做的事
}
```

if 後面的小括號裡是個 **條件判斷式**，它的運算結果必需是 **布林值(boolean)**。

如果條件判斷式的運算結果為 **true** 則執行接下來那組大括號內的程式碼，否則就略過那整個大括號的內容。

關鍵往往在於你是否能找到一個合宜的條件判斷式，來抓到你要的狀態。

練習：判斷奇數(odd number)

讀入使用者輸入的整數 \$a\$，若其為奇數，輸出 "奇數" 否則什麼都不做。

我們要怎麼判定 \$a\$ 是奇數呢？只要把它除以 2，看餘數是不是 1 就知道了。

```
int a;
cin >> a;

if(a%2==1)
{
    cout << "奇數" << endl;
}
```

if ... else ...

如果我們在條件判斷式成立時要做「工作A」，不成立時要做「工作B」，該怎麼辦呢？

這種 2 條分支的狀況，可以使用 **if ... else ...** 敘述，語法如下。

```
if( 條件判斷式 )
{
    條件成立時要做的事
}
else{
    條件不成立時要做的事
}
```

可以把它想成是「**如果 (...) 就做 {...} 否則做 {...}**」

練習：判斷奇、偶數

讀入使用者輸入的整數 `$a$`，若其為奇數，輸出 "奇數" 否則輸出 "偶數"。

```
int a;
cin >> a;

if(a%2==1)
{
    cout << "奇數" << endl;
}
else
{
    cout << "偶數" << endl;
}
```

if ... else if ... else

狀況再複雜一點，如果分支多於 2 條呢？例如：判斷 `$a$` 是「正數」、「負數」還是「0」

這時我們需要加入 **else if**，來做到「**如果 (...) 就做 {...} 否則如果 (...) 就做 {...} 否則做 ...**」

```

if( 條件判斷式1 )
{
    條件1成立時要做的事
}
else if( 條件判斷式2 )
{
    條件2成立時要做的事
}
else
{
    前面條件都不成立時要做的事
}

```

練習：正、負數與零的判斷

讀入使用者輸入的整數 `$a$`，輸出其為 "正數"、"負數" 或 "零"。

```

int a;

cin >> a;

if(a>0)
{
    cout << "正數" << endl;
}
else if(a<0)
{
    cout << "負數" << endl;
}
else
{
    cout << "零" << endl;
}

```

分支大於 3 條時，只要重覆多個 **else if** 即可。

練習：分數轉換為等第

讀取使用者輸入的成績，輸出其相應的等第，轉換參考表如下。

分數	等第
90 ~	A

分數	等第
80 ~ 89	B
70 ~ 79	C
60 ~ 69	D
~ 59	E

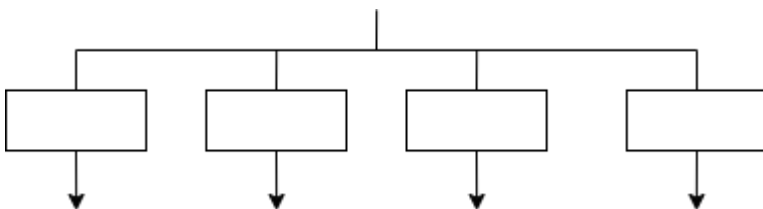
```
int score;

cin >> score;

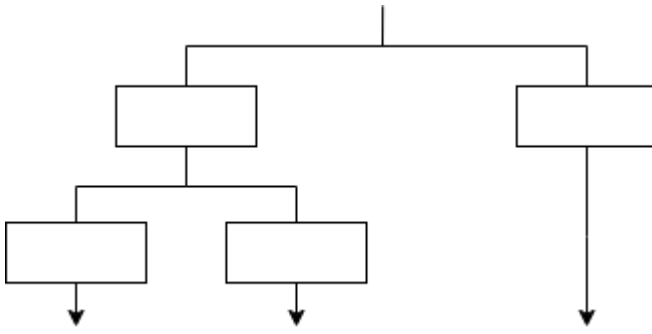
if(score>=90)
{
    cout << "A" << endl;
}
else if(score>=80)
{
    cout << "B" << endl;
}
else if(score>=70)
{
    cout << "C" << endl;
}
else if(score>=60)
{
    cout << "D" << endl;
}
else
{
    cout << "E" << endl;
}
```

巢狀(nested)/多層 結構

目前我們遇到的選擇結構是像這樣 **單層多分支**。



但是也有像這樣 **多層多分支** 的選擇結構。



練習：是否需服兵役

由使用者輸入性別、年齡，只有男生且年齡大於等於 20 歲才需要服兵役。

只有 2 個檢查條件都成立，才會被判定需當兵。

```
string gender;
cout << "性別(男, 女):";
cin >> gender;

int age;
cout << "年齡:";
cin >> age;

if(gender=="男")
{
    cout << "你是男生, ";
    if(age>=20)
    {
        cout << "需要當兵" << endl;
    }
    else
    {
        cout << "但是年紀太小, 還不用當兵" << endl;
    }
}
else
{
    cout << "你是女生, 不用當兵" << endl;
}
```

在這裡我們用到了如下的巢狀結構。

```
if( 條件判斷式1 )
{
    條件 1 成立時要做的事
    if( 條件判斷式2 )
    {
        條件 1, 2 都成立時要做的事
    }
    else
    {
        條件 1 成立、2 不成立時要做的事
    }
}
else
{
    條件 1 不成立時要做的事
}
```

一組大括號包起來的部分，我們稱為一個程式區塊(block)。每個區塊內都可以再塞進其他的區塊。

縮排(indent)

如前所述我們可以有各式各樣很多層次的程式碼。當層次一多起來，程式碼就會開始亂，連自己的東西都不容易看懂。

所以在撰寫 C++ 程式時，我們都會遵守一個規範，在出現大括號時，裡面的程式碼就會縮一層(4個空白字元或是一個 [tab])，這樣可以讓程式碼的層次一目了然。

雖然不縮排，程式也能執行，但是你的伙伴會看不懂你在寫什麼，也不會想看你的程式。

```
string genger;  
cout << "性別(男, 女):";  
cin >> gender;  
  
int age;  
cout << "年齡:";  
cin >> age;  
  
if(gender=="男")  
{  
    cout << "你是男生,";  
    if(age>=20)  
    {  
        cout << "需要當兵" << endl;  
    }  
    else  
    {  
        cout << "但是年紀太小, 還不用當兵" << endl;  
    }  
}  
else  
{  
    cout << "你是女生, 不用當兵" << endl;  
}
```

🕒Revision #62

★Created 2023-05-03 00:00:47 UTC by huihui

✎Updated 2024-03-06 16:14:56 UTC by huihui