

在電腦世界 所有得數字都表示成 2 進位

以 c 語言的 double 來說 它採用 IEEE754 的雙精度標準

0.1 二進位表示成:0.00011 (0011) 循環 以 IEEE754 標準來說即為 $2^{-4} \times 1.10011(0011 \times 12 \text{ 次循環})001$

0.2 二進位表示成:0.0011 (0011) 循環 以 IEEE754 標準來說即為 $2^{-3} \times 1.10011(0011 \times 12 \text{ 次循環})010$

0.3 二進位表示成:0.01001(1001)循環 以 IEEE754 標準來說即為 $2^{-2} \times 1.0011(0011 \times 13 \text{ 次循環})$

$0.1+0.2=2^{-2} \times 1.0011(0011 \times 11 \text{ 次循環}) 0100$ 約為 0.300000000000000004

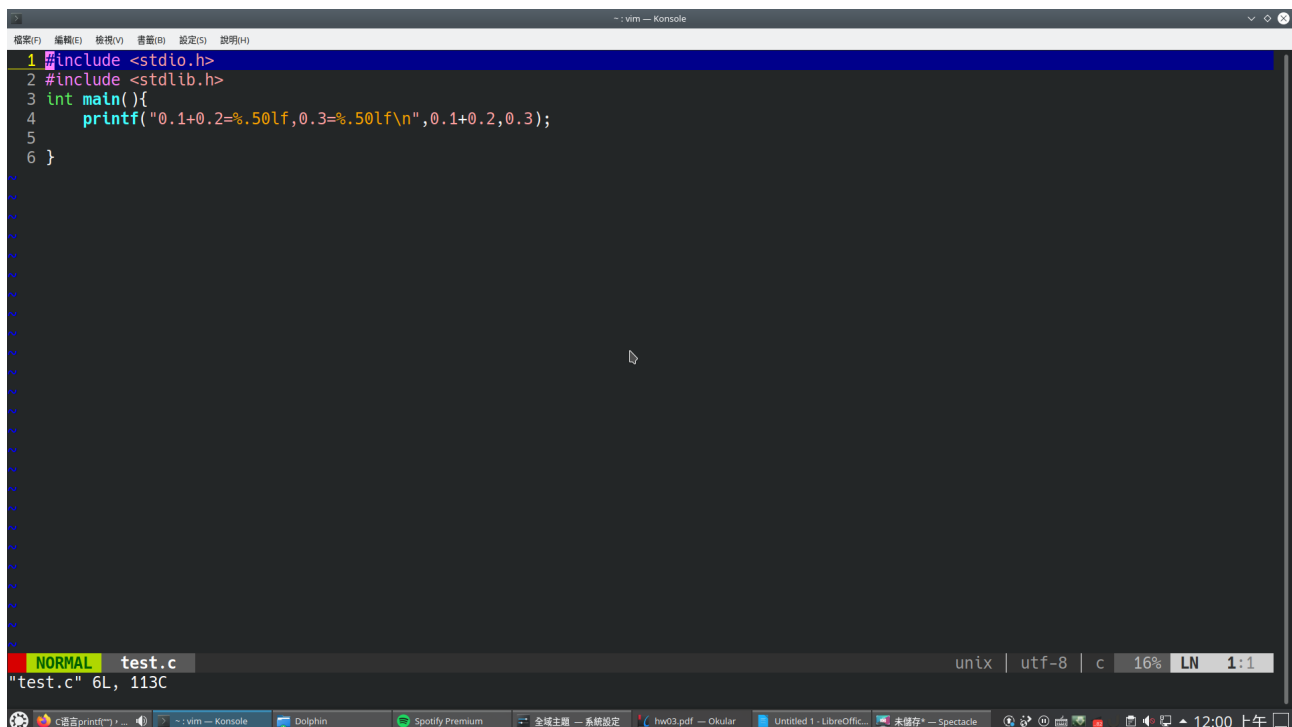
與 0.3 的 $2^{-2} \times 1.0011(0011 \times 13 \text{ 次循環})$ 不相同 且超過

所以題目的判斷只做了 3 次 printf 就跳出了

驗證：

為了驗證上述觀點的正確 我們使用 printf 將 0.1+0.2 得結果和 0.3 的結果打印出來 並保留小數後 50 位

程式碼如下：



```
1 #include <stdio.h>
2 #include <stdlib.h>
3 int main(){
4     printf("0.1+0.2=%50lf,0.3=%50lf\n",0.1+0.2,0.3);
5 }
6 }
```

The screenshot shows a terminal window with a dark background. The code is written in a light blue font. The terminal output is not visible, only the code is shown. The terminal window has a title bar that says "vim - Konsole". The bottom of the window shows a status bar with "NORMAL", "test.c", "unix", "utf-8", "c", "16%", "LN", "1:1". The system tray at the bottom shows various icons including a clock showing 12:00 上午.

結果為：

```
blast@blast-IdeaPad-L3-15IML05:~$ vim test.c
blast@blast-IdeaPad-L3-15IML05:~$ gcc test.c
blast@blast-IdeaPad-L3-15IML05:~$ ./a.out
0.1+0.2=0.30000000000000004440892098500626161694526672363281,0.3=0.29999999999999998889776975374843459576368331909180
blast@blast-IdeaPad-L3-15IML05:~$
```

如我們之前所說的 $0.1+0.2$ 約等於 $0.3\dots04 > 0.3$
故得證