**★☆☆☆☆**

**題組：Problem Set Archive** **with Online Judge**

**題號：10323: Factorial! You Must be Kidding!!!**

**解題者：孫世諭**

**解題日期：2024年2月29日**

**題意：**

輸入一個整數n，計算其階乘n!。如果計算結果滿足10000 < n! <6227020800，則輸出n!的值；如果超出上限6227020800，則輸出「Overflow!」；如果低於下限 10000，則輸出「Underflow!」。階乘公式如下：

*F(0) = 1*, F(n) = n\*F(n-1)

**題意範例：**

2 : Underflow!

10 : 3628800  
100 : Overflow!

**解法：**

首先，要先確定 n 為負數的情況：  
 F(n) = n \* F(n-1) → F(n-1) = F(n) / n  
 F(0) = 1  
 F(-1) = 1 / F(0) = 正無限大  
 F(-2) = -1 / F(-1) = 負無限大  
 F(-3) = -2 / F(-2) = 正無限大 ……

故可得知：

(1) n < 0, n mod 2 = 0: Underflow!

(2) n < 0, n mod 2 = 1: Overflow!  
接下來是正數的方法：  
(方法1) 用暴力法直接計算即可：  
 (3) n>0, n! > 6227020800: Overflow!  
 (4) n>0, n! < 10000: Underflow!

(方法2) 直接找出 upperbound 與 lowerbound：  
 (3) ⸪13!=6227020800, ⸫n>13: Overflow!  
 (4) ⸪7!=5040, 8!=40320, ⸫n<8: Underflow!  
**解法範例：**

(1) 10 → n! = 3628800 → 3628800  
(2) -55 → n < 0, n mod 2 = 1 → Overflow!  
(3) -14 → n < 0, n mod 2 = 0 → Underflow!  
(4) 7 → (方法1) 7!= 5040 < 10000 → Underflow!  
 → (方法2) 7<8 → Underflow!  
(5) 25 → (方法1) 25!=1.551121e+25 → Overflow!  
 → (方法2) 25 > 10 → Overflow!

**討論：**

1. 如果是使用方法一的話，會有 C 語言大數 overflow 的問題，可以在每次迴圈執行時檢查值是否大於 6227020800。
2. 因為答案會重複使用，可以建立一個陣列儲存已經計算過的階乘值，加快程式執行速度（本題非必要）。
3. 本題時間複雜度為 O(n)，但因為 n 的合法範圍為 8! ~ 13!，少到可以直接不用程式計算，達到 O(1) （本題非必要）。
4. 我認為這題沒有標明n的範圍，很容易誤導解題者，以致認為n沒有負值。

**程式：**

#include <iostream>

using namespace std;

int main()

{

typedef long long int ull;

ull s=1;

ull input;

while(cin >> input)

{

s=1;

if(input < 0)

{

if(input%2 == 0) cout << "Underflow!" << '\n';

else cout << "Overflow!" << '\n';

continue;

}

for(ull i=1; i<=input; ++i)

{

s\*=i;

if(s>6227020800)

{

cout << "Overflow!" << '\n';

break;

}

}

if(s<10000) cout << "Underflow!" << '\n';

else if(s>6227020800) {}

else cout << s << '\n';

}

return 0;

}