Precisión Limitada en el Punto Flotante

El punto flotante tiene distintos formatos, por ejemplo “simple precisión”. ¿A qué se refiere con precisión?

- El formato tiene una precisión para representar números muy grandes, o muy chicos, y esto depende del tamaño de la mantisa.
- Esto quiere decir que cuanto más grande o más chico sea el número, el formato en algunos casos no será capaz de representar el número completamente.

El formato de simple precisión tiene 23 bits en la mantisa. El número más grande que se puede representar en la mantisa es el siguiente (veintitrés 1):

$$11111111111111111111111_2$$

Hay que recordar aquí que a la izquierda de la mantisa siempre hay un 1 que se omite cuando se traduce un número a punto flotante, por lo tanto, a los 23 bits de la mantisa le agregamos 1 bit más para saber el número que estábamos representando:

$$11111111111111111111111_2 = 16777215_{10}$$

Este número, representa el número más grande que se puede representar sin perder precisión. ¿Por qué?

Supongamos el siguiente número en punto flotante:

0 10010111 00000000000000000000000 ***(a)***

Si traducimos el exponente a decimal, el resultado es 151. Le restamos el desplazamiento y queda: 151 – 127 = 24

Esto quiere decir que hay que “mover la coma” 24 veces hacia la derecha, pero… ¿Cuántos bits tiene la mantisa? ¡23!, esto quiere decir que hay que mover la coma más allá de la mantisa.

Por lo tanto, queda:

$$1{,}00000000000000000000000 \times 2^{24} = 100000000000000000000000\mathbf{0}{,}00$$

¿Qué número decimal representa ese binario?

$$100000000000000000000000_2 = 16777216_{10}$$

Si quisiéramos representar el número 16777217_{10} en punto flotante, haríamos

$16777217_{10} = 100000000000000000000001_2$

Lo pasamos a notación científica binaria como:

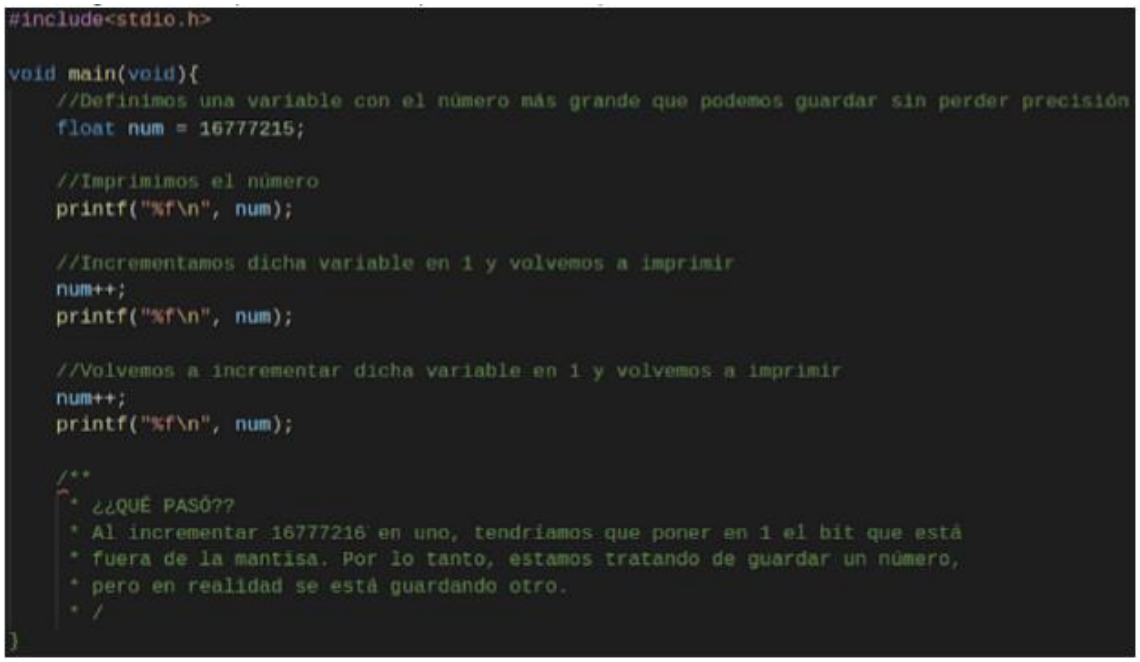
$$100000000000000000000001_2 = 1{,}00000000000000000000001 \times 2^{24}$$

Y representado en punto flotante quedaría:

0 10010111 00000000000000000000001 ***(b)***

Porque el 1 final en la posición 24 no entra en los 23 bits de mantisa. ¿Qué pasa cuando comparamos **(a)** con **(b)**?

El siguiente código en C ilustra el problema de la precisión en el punto flotante:



```c
#include<stdio.h>

void main(void){
    //Definimos una variable con el número más grande que podemos guardar sin perder precisión
    float num = 16777215;

    //Imprimimos el número
    printf("%f\n", num);

    //Incrementamos dicha variable en 1 y volvemos a imprimir
    num++;
    printf("%f\n", num);

    //Volvemos a incrementar dicha variable en 1 y volvemos a imprimir
    num++;
    printf("%f\n", num);

    /**
     * ¿¿QUÉ PASÓ??
     * Al incrementar 16777216 en uno, tendríamos que poner en 1 el bit que está
     * fuera de la mantisa. Por lo tanto, estamos tratando de guardar un número,
     * pero en realidad se está guardando otro.
     */
}
```

Creación del apunte: Martín Isusi Seff

Correcciones: Nicolás Díaz Repice