

BOB35APO - Seminar 2: Examples

Bc. Štěpán Pressl

Násobení celých čísel (ručně)

Provedte v dvojkové soustavě

1. $25 \cdot 123$

2. $-100 \cdot 49$

IEEE 754

Znaménkový bit, mantissa, exponent. Normalizované číslo.

Denormalizované číslo. NaN. Nekonečno.

Example 1 (IEEE 754)

Dobrovolník nechť zodpoví na 2 otázky (uvažujme 32 bitový float):

1. O kolik je exponent posunutý a proč je posunut?
2. Proč se mantissa normalizuje? Kolik bitů obsahuje mantissa?
3. Jakého exponentu nabývají čísla $\pm\infty$ a NaN?

Example 2 (IEEE 754)

Dobrovolník nechť zodpoví na 2 otázky (uvažujme 32 bitový float):

1. Jaký je interval hodnot denormalizovaných čísel?
2. Jaké je maximální kladné číslo (kromě $+\infty$), které lze reprezentovat pomocí 32 bitového floatu?

Example 3 (Převod čísel IEEE754)

Převeďte následující čísla. Napište jejich 32 bitovou binární reprezentaci.

1. -0.75
2. 0.5
3. -0.4375

Example 4 (Převod čísel IEEE754)

Převeďte float z binární reprezentace 0xC0A00000 na reálné číslo v desítkové soustavě.

Example 5 (Aritmetika čísel v IEEE754)

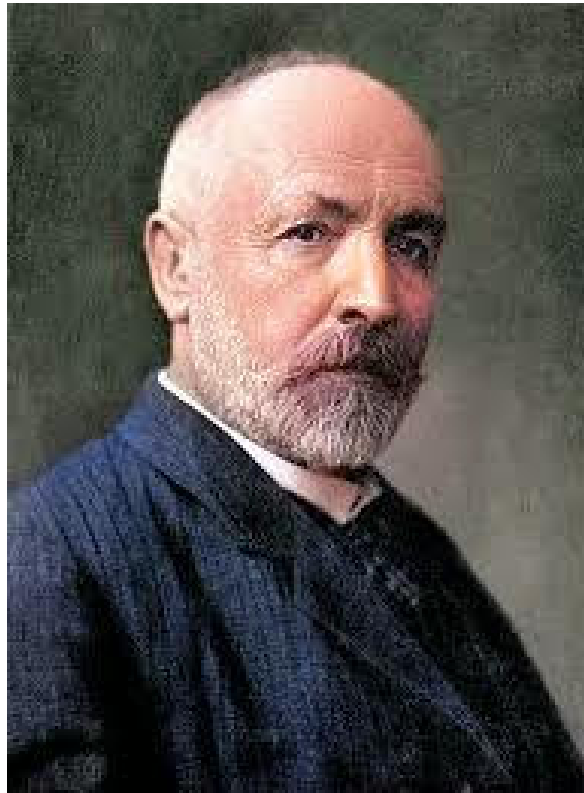
1. Demonstrujte výpočet (v desítkové soustavě)

$9.999 \cdot 10^1 + 1.1610 \cdot 10^{-1}$, předpokládejte, že je možné uložit pouze 4 cifry čísla a 2 cifry exponentu.

2. V binární reprezentaci sečtěte čísla 0.5 a -0.4375.

3. V binární reprezentaci vynásobte čísla 0.5 a -0.4375.

Kdo to je?



Odpověď je Georg Cantor. Studenti Ol zavzpomínají na to, co znamená spočítatelnost. Je množina reálných čísel spočítatelná?

Example 6 (počítačových reálných čísel není nespočítatelně mnoho!)

Uvažujme 32 bitový float. Kolik je denormalizovaných čísel? Kolik rozdílných hodnot lze 32 bitovým floatem reprezentovat?

Example 6 (řešení)

Denormalizovaných čísel je $2 \cdot 2^{23}$.

U normalizovaných čísel může exponent nabývat hodnot 1 až 254, tedy 254 kombinací. Mantissa může být jakákoli, tedy (včetně znaménka) $2 \cdot 254 \cdot 2^{23}$.

Speciální případy: ± 0 , $\pm \infty$, NaN.

Dá se to i využít!

```
float Q_rsqrt( float number )
{
    long i;
    float x2, y;
    const float threehalfs = 1.5F;

    x2 = number * 0.5F;
    y = number;
    i = * ( long * ) &y;
    i = 0x5f3759df - ( i >> 1 );
    y = * ( float * ) &i;
    y = y * ( threehalfs - ( x2 * y * y ) ); // 1st iteration
    // y = y * ( threehalfs - ( x2 * y * y ) ); // 2nd iteration, can be removed

    return y;
}
```

32-bit number
32-bit decimal number
1.5 (also 32-bit)

// evil floating point bit hack
// what the fuck?

// 1st iteration
// 2nd iteration, can be removed

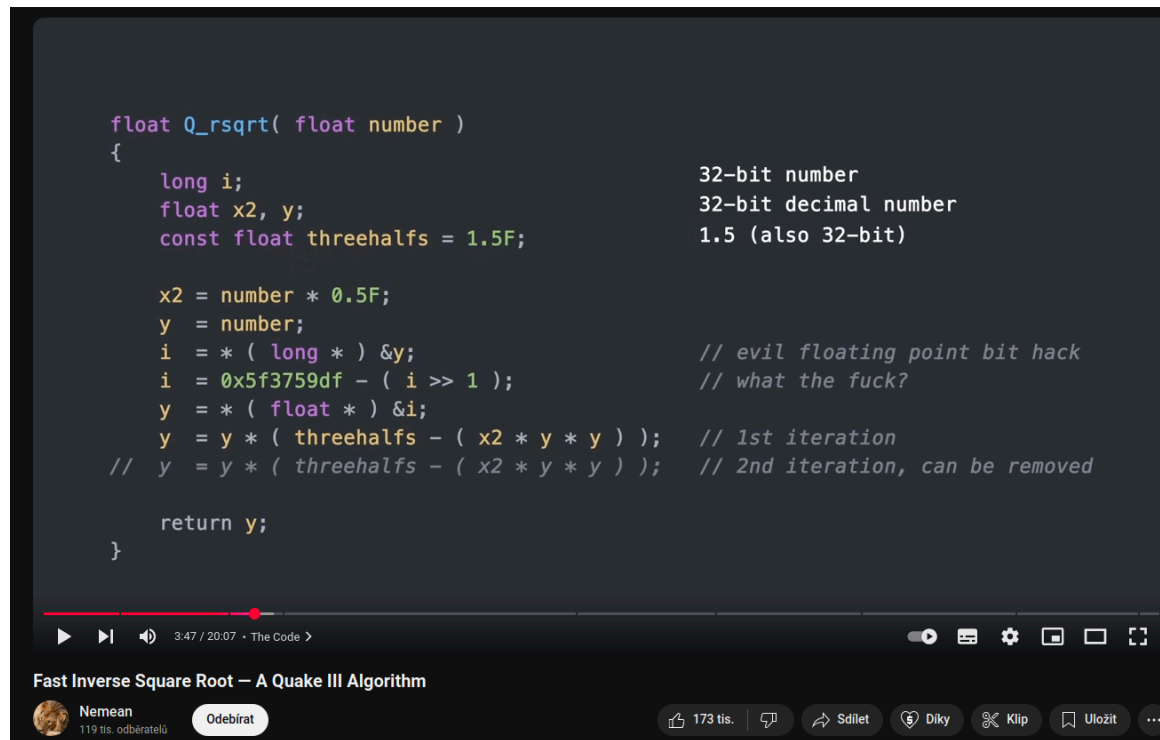


Figure 2: https://www.youtube.com/watch?v=p8u_k2LIZyo