1 Packing of numbers

Estimate how many numbers there are in the interval from 1.0 and 2.0; and in between the interval of 255.0 to 256.0, for IEEE-754.

a) single precision

A single precision (binary 32) floating point number has 1 sign bit, 8 exponent bits and 24 bits of significand precision. Its range is from $\pm 1.18 \cdot 10^{-38}$ to $\pm 3.4 \cdot 10^{38}$

b) double precision

A double precision float can take on values from $\pm 2.23 \cdot 10^{-308}$ to $\pm 1.80 \cdot 10^{308}$.

2 Pitfalls of integer & floating point arithmetic

a) Consider the following *C/C++* code:

```
int i = 7;
float y = 2*(i/2);
float z = 2*(i/2.);
printf("%e %e \n", y,z)
```

The two variables y and z do not hold the same values, since $i/2$ is evaluated to 3, while for $i/2.$ it is 2.5. Thus, multiplying with 2 once yields the integer 4, and once the float 5.

b) Again, consider the following *C/C++* code:

```
double a = 1.0e17;
double b = -1.0e17;
double c = 1.0;
double x = (a + b) + c;
double y = a + (b + c);
```

3 Machine epsilon

For the datatypes float, double & long double, determine the smallest number ε_{min}, such that $1 + \varepsilon_{min}$ still returns something different than 1.

To find ε_{min} for floats, we can utilize the following *C* code snippet:

```c
#include <stdio.h>
#include <stdbool.h>


float find_epsilon () {

    float one = 1;
    float epsilon = 1;
    float new_epsilon;

    bool found_epsilon = false;
    while (!found_epsilon) {
        new_epsilon = epsilon / 2.;
        if (one + new_epsilon == one) {
            return epsilon;
        }
        epsilon = new_epsilon;
    }
}

int main(void) {
    printf(
        "%e \n", find_epsilon()
    );
}
```

All occurences of *float* can be switched out for *double* or *long double* to get the other values of ε_{min}.

This yields:

type	ε_{min}
long double	0
double	$\approx 10^{-16}$
float	$\approx 10^{-7}$

Evaluate and print out $1 + \varepsilon$. Do you see something strange?

4 Near-cancellation of numbers

Consider the following function:

$$f(x) = \frac{x + e^{-x} - 1}{x^2} \tag{1}$$

a) Determine $\lim_{x \to 0} f(x)$

b) Write a computer program that asks for a value of x from the user and then prints $f(x)$.

c) For small $x > 0$ this evaluation goes wrong. Determine experimentally at which values of x the formula goes wrong.

d) Explain why this happens.

e) Add an if-clause to the program such that for small values the function is evaluated in another way that does not break down, so that for all positive values of x the program produces a reasonable result.