Homework 1

By Buchnev Arseniy

1. **Check if the following equality holds using MATLAB:**

   `0.1 + 0.2 == 0.3`

   Try to represent distinct numbers using `fprintf('%.20e', x)` and explain the result.

   Solution:

```
>> 0.1 + 0.2 == 0.3

ans =

  logical

   0
```

   This happens due to floating point numbers having finite accuracy in n-bit representation. The number represented in computer is the closest floating-point number that can be written in binary:

```
>> fprintf('%.20e', 0.1)
1.00000000000000005551e-01>>
>> fprintf('%.20e', 0.2)
2.00000000000000011102e-01>>
>> fprintf('%.20e', 0.3)
2.99999999999999988898e-01>>
```

   Obviously, this combination will not hold the truth check.

2. **Check the associativity of summation:**

   `(0.1 + 0.2) + 0.3 == 0.1 + (0.2 + 0.3)`

   Find the exponents of these numbers in decimal format for 32-bit float according to IEEE Standard with the help of online converter. Explain the loss of accuracy during floating point addition on binary level. Is the loss of accuracy always guaranteed?

   Solution:
   As it may seem obvious,

```
>> (0.1 + 0.2) + 0.3 == 0.1 + (0.2 + 0.3)

ans =

  logical

   0
```

Let us check the binary repr. online:

0.1

Decimal	0.1

32 bit – float

Decimal (exact)	0.100000001490116119384765625
Binary	0 01111011 10011001100110011001101

0.2

Decimal	0.2

32 bit – float

Decimal (exact)	0.20000000298023223876953125
Binary	0 01111100 10011001100110011001101

0.3

Decimal	0.3

32 bit – float

Decimal (exact)	0.300000011920928955078125
Binary	0 01111101 00110011001100110011010

0.5 (= 0.2 + 0.3)

Decimal	0.5

32 bit – float

Decimal (exact)	0.5
Binary	0 01111110 00000000000000000000000

```
>> (0.2 + 0.3) == 0.5

ans =

  logical

   1
```

As we can see, the loss of accuracy won't occur, if the real number can be represented exactly as a combination of powers of 2, which 0.5 obviously can (2^{-1}). In the example above we have (0.2 + 0.3) which wraps into 0.5, which, in turn, is represented exactly. Not so for (0.1 + 0.2) = 0.3.

3. **Check if the following equalities hold using MATLAB:**

   $(2^{53} + 1) - 2^{53} == 1$

   $(2^{53} + 2) - 2^{53} == 2$

   Solution:

   We can find the answer in the IEEE 754 standard before running the code:

   Precision limitations on integer values

   - Integers from -2^{53} to 2^{53} (−9,007,199,254,740,992 to 9,007,199,254,740,992) can be exactly represented
   - Integers between 2^{53} and 2^{54} = 18,014,398,509,481,984 round to a multiple of 2 (even number)
   - Integers between 2^{54} and 2^{55} = 36,028,797,018,963,968 round to a multiple of 4

   Thus, our 2^{53} + 1 will be rounded to a multiple of 2, causing the equality to not hold:

```
>> (2^53 + 1) - 2^53 == 1

ans =

  logical

   0

>> (2^53 + 2) - 2^53 == 2

ans =

  logical

   1
```