# CS Basics - Exercises
# Codification of numbers

E. Benoist

Fall Term 2018-19

## 1 Encoding

### 1.1 Little endian

**Exercise 1** *Compute in decimal numbers the following 32-bit integers in little endian convention.*

- *00H 00H 01H 00H*
- *AFH 01H 00H 00H*
- *00H 01H 00H 00H*
- *10H 10H 00H 00H*
- *00H FFH 01H 00H*

**Exercise 2** *Write the value in memory of the following numbers in little endian on a 32-bit computer.*

- *35*
- *390*
- *450*
- *1003*

## 1.2 Signed integers

**Exercise 3** *Using the two's complement notation write the representation of the following numbers in one byte. Write the result in hexadecimal.*

- *100*
- *67*
- *-10*
- *-5*
- *-67*
- *-130*
- *-89*
- *-255*

**Exercise 4** *Execute the following additions in hexadecimal (using two's compelement notation):*

- *100 + (-67)*
- *67 + (-5)*
- *(-67) + (-5)*

## 1.3 Unsigned integers with bias

**Exercise 5** *Compute the value of the following unsigned integers with the bias 127 (e.g -10 is represented by the number 117 and 20 is represented by the number 147). Write the number in hexadecimal form.*

- *0*
- *10*
- *120*
- *-20*
- *-15*

## 1.4 Conversion of decimal to binary of floating point numbers

**Exercise 6** *Convert the following decimal numbers into binary.*

- Example: *0.5*

  *For a number smaller than 1, each time, we multiply the number by 2. If the result is larger than 1, we note 1 (for the result) and subtract 1. If the result is smaller than 1, we go on with the same number. At each step we multiply the resulting number by 2.*

  ```
  0.5          | 0.
  1.0 (=0.5*2) | 1 -> 0.0
  0 -> End
  ```

  *Result:* ***0.5 = 0.1B***

- *0.125*
- *0.4*
- *0.89*
- *25.1*
- *9.90*

## 1.5 Floating point numbers

**Exercise 7** *Compute the representation of the following numbers on 32 bits (write it in bits, then in hexadecimal notation):*

- Example:*0.5*

  *Sign bit is = 0;*

  *0.5 = 1.00000000 * $2^{-1}B$*

  *Mantissa is 1.00000000000000000000000, but without its first one (hidden bit) it becomes 00000000000000000000000*

  *Exponent is -1, and is encoded -1+127 = 126 = 0111 1110B*

  *Encoding of 0.5 in float is*

  ```
  0 0111 1110 00000000000000000000000
  ```

  *This gives the binary representation:* ***0011'1111'0000'0000'0000'0000'0000'0000B***
  *This gives the Hexadecimal representation:* ***3F 00 00 00 H***

- *12*
- *34.25*
- *0.1*
- *-10.98*

**Exercise 8** *Compute the biggest number that can be represented by a 32 bit float.*
*Hint :* $(1/2) + (1/2^{2)} + (1/2^3) + (1/2^4) + ... + (1/2^n) = (2^n - 1)/(2^n) = 1 - (1/2^n)$