CS Basics - Exercises
Encoding Numbers

E. Benoist & C. Fuhrer

Fall Term 2022-23

1 Encoding

1.1 Little Endian

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

- 0x00 0x00 0x01 0x00
- 0xAF 0x01 0x00 0x00
- 0x00 0x01 0x00 0x00
- 0x10 0x10 0x00 0x00
- 00 FF 01 00

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. Think about how the result would look like if two bytes where used for storing.

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

Exercise 4. Perform the following additions in hexadecimal (using two's complement notation):

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

1.3 Unsigned Integers with Bias

Exercise 5. Compute the value of the following unsigned integers using 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 Fixed Point Numbers: Conversion from Decimal to Binary

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 = 0b0.1**

- 0.125
- 0.4
- 0.89
- 25.1
- 9.90

1.5 Floating Point Numbers

Exercise 7. Compute the representation of the following 32 bit numbers (write it in binary first, then in hexadecimal notation). You may check your answer using `https://www.h-schmidt.net/FloatConverter/IEEE754.html`.

- *Example:* 0.5 Sign bit is = `0`;

  $0.5 = (\texttt{1.0} * 2^{-1})_2$

  Mantissa is `1.000 0000 0000 0000 0000 0000`, but without the *hidden bit* it becomes `000 0000 0000 0000 0000 0000`.

  Exponent is -1, and is encoded -1+127 = 126 = 0b0111 1110

  Encoding of 0.5 in float is

  $$0\ 0111\ 1110\ 000\ 0000\ 0000\ 0000\ 0000\ 0000$$

  Binary representation: **0b0011 1111 0000 0000 0000 0000 0000 0000**

  Hexadecimal representation: **0x3F 00 00 00**

- 12
- 34.25
- 0.1
- -10.98

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