Introduction to Computer Science
Jacobs University Bremen
Dr. Jürgen Schönwälder

Course: CH-232-A
Date: 2020-10-09
Due: 2020-10-16

ICS 2020 Problem Sheet #5

Problem 5.1: *b-complement* (1+1+1 = 3 points)

We plan to use a fixed size b-complement number system with the base $b = 5$ and $n = 4$ digits.

a) What are the smallest and the largest number that can be represented and why?

b) What is the representation of -1 and -8 in b-complement notation?

c) Add the numbers -1 and -8 in b-complement notation. What is the result in b-complement representation? What is the result converted back into the decimal number system?

Problem 5.2: *IEEE 754 floating point numbers* (4+1 = 5 points)

IEEE 754 floating point numbers (single precision) use the following format (the numbers on the top of the box indicate bit positions, the fields in the box indicate what the various bits mean).

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|S|    exponent   |         mantissa (23 bits)                    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The encoding starts with a sign bit, followed by the biased exponent (8 bits), followed by the mantissa (23 bits). For single-precision floating-point numbers, the exponents in the range of -126 to +127 are biased by adding 127 to get a value in the range 1 to 254 (0 and 255 have special meanings).

a) The absolute zero, 0 Kelvin, is at -273.15 degree Celsius. Explain step by step (and in your own words) how the decimal fraction -273.15_{10} is converted into a single precision floating point number.

b) What is the decimal fraction that is actually stored in the single precision floating point number?

Problem 5.3: *unicode and utf-8 encoding* (1 point)

You are given the following UTF-8 sequence in hexadecimal notation (4 bytes):

```
f0 9f 90 84
```

Write each byte in binary notation and identify the bits representing the unicode code point. What is the Unicode code point (in hexadecimal) and which character does it represent?

Problem 5.4: *evil and odious numbers (haskell)* (1 point)

An evil number is a non-negative integer that has an even number of 1s when represented in binary. An odious number has an odd number of 1s in its binary representation.

a) Implement a function `isEvil :: Int -> Bool` that returns `True` if the argument has an even number of 1s in its binary representation and `False` otherwise. Use this function to implement the function `evils :: [Int]` that returns the list of non-negative evil numbers.

b) Using function composition, implement a function `isOdious :: Int -> Bool` that returns `True` if the argument has an odd number of 1s in its binary representation and `False` otherwise. Use this function to implement the function `odious :: [Int]` that returns the list of non-negative odious numbers.

```
Prelude> take 10 evils
[0,3,5,6,9,10,12,15,17,18]
Prelude> take 10 odious
[1,2,4,7,8,11,13,14,16,19]
```

Submit your Haskell code as a plain text file.