Programming Assignment #1:

Tiny FP (8-bit floating point) representation

Due: September 27th (Wednesday), 11:59PM

1. Introduction

The purpose of this assignment is to get familiar with the floating-point representation by implementing a simplified 8-bit floating-point representation.

2. Problem specification

2.1. Overview

tinyfp is a simplified 8-bit floating point representation which follows the IEEE 754 standard for floating-point arithmetic. The overall structure of the tinyfp representation is shown below. The MSB (Most Significant Bit) is used as a sign bit (s). The next four bits are used for exponents (exp) with a bias value of 7. The last three bits are used for the fractional part (frac).

\[
\begin{array}{cccc}
7 & 6 & 3 & 2 & 0 \\
| & s | \text{exp} | \text{frac} |
\end{array}
\]

In C, the new type tinyfp can be defined as follows.

```c
typedef unsigned char tinyfp;
```

Your task is to implement the following four C functions that convert int or float type values to the tinyfp format and vice versa.

```c
tinyfp int2tinyfp(int x);
int tinyfp2int(tinyfp x);
tinyfp float2tinyfp(float x);
float tinyfp2float(tinyfp x);
```
2.2. Implementation details

2.2.1. int2tinyfp()

- Integer zero (0) should be converted to plus zero (+0.0) in tinyfp.

- An integer value that exceeds the range of the tinyfp representation should be converted to the infinity in tinyfp (+∞ or -∞ depending on the sign).

- If necessary, use the round-toward-zero mode.

2.2.2. tinyfp2int()

- Drop the fractional part when you convert the value in the tinyfp format to integer. (e.g. the value 1.5 in tinyfp is converted to 1)

- Convert +∞ and -∞ in tinyfp to TMax and Tmin in integer, respectively.

- +NaN and -NaN in tinyfp is converted to Tmin in integer.

2.2.3. float2tinyfp()

- A floating-point value that exceeds the range of the tinyfp representation should be converted to the infinity in tinyfp (+∞ or -∞ depending on the sign).

- If necessary, use the round-toward-zero mode.

2.2.4. tinyfp2float()

- The tinyfp type is a subset of the float type. Hence, all the values in tinyfp should be represented in the float format without any error.
3. Example

The skeleton code is available in the course homepage (http://csl.skku.edu/SSE2030F17/Assignment)

The results of some sample runs are as follows.

```
kisik@kisik-desktop:~:/sse2030/pa1$ ./pa1-test
Test 1: casting from int to tinyfp
int(11111111 11111111 11111111 11011110) => tinyfp(11100000), CORRECT
int(00000000 00000000 00000000 01000011) => tinyfp(01101000), CORRECT
Test 2: casting from tinyfp to int
tinyfp(11101010) => int(11111111 11111111 11111111 10110000), CORRECT
tinyfp(01010101) => int(00000000 00000000 00000000 00001101), CORRECT
Test 3: casting from float to tinyfp
float(11000001 01000101 10000101 00111111) => tinyfp(11010100), CORRECT
float(00111111 11001100 11001100 11001101) => tinyfp(00111100), CORRECT
Test 4: casting from tinyfp to float
tinyfp(11101010) => float(11000001 01000101 10000101 00111111), CORRECT
tinyfp(01010101) => float(01000001 01010000 00000000 00000000), CORRECT
kisik@kisik-desktop:~:/sse2030/pa1$
```

4. Hand in instructions

- Register an account to the submission site (http://sys.skku.edu).
  - You must type your real name & student ID
  - Wait for an enrollment.
- Submit only the pa1.c file to the submission site.

5. Logistics

- You will work on this assignment alone.
- Only the assignments submitted before the deadline will receive the full credit. 25% of the credit will be deducted for every single day delay.
  - You can use up to 5 slip days
- Any attempt to copy others’ work will result in heavy penalty (for both the copier and the originator). Don’t take a risk.
Good luck!

TA: Kisik Jeong (kisik.jeong@cs1.skku.edu)

Computer Systems Laboratory

Sungkyunkwan University