CSci 2021 Fall 2018 Section 010 Written Exercise Set 1

Due on paper at the beginning of lecture (3:35pm) on Monday October 1, 2018. We strongly recommend that you type and print out your solutions. Please label your assignment with your name, UMN email address, and the time of your lab section (10:10, 11:15, 12:20, or 1:25).

Problem 1

In this problem, assume the variables a and b are signed 32-bit integers and that the machine uses two's complement representation, with the same overflow behavior we described in lecture. `TMAX` is the largest signed integer, and `TMIN` is the most negative signed integer. Match each numbered expression on the left with the corresponding expression on the right that is equivalent for all values of a and b. There is exactly one correct match for each expression on the left.

_____	1. `a ^ b`	a)	`a ^ (TMIN + TMAX)`
		b)	`~a | ~b`
_____	2. `~a`	c)	`a & TMIN`
		d)	`(a | b) & (~a | ~b)`
_____	3. `a / 8`	e)	`~TMAX`
		f)	`(1 << 31) >> 1`
_____	4. `a & 1`	g)	`a | TMAX`
		h)	`1 << (31 - 1)`
_____	5. `TMIN`	i)	`((a < 0) ? (a + 7) : a) >> 3`
		j)	`a >> 3`
_____	6. `(a < 0) ? 1 : -1`	k)	`~((a >> 31) << 1)`
		l)	`~((a << 31) >> 31)) + 1`

Problem 2

Assume you have a machine with a 16-bit word size and 65536 bytes of memory, so that memory addresses take up 16 bits (2 bytes). Likewise, `short`s are 2 bytes. Assume that the variables in this code are placed in memory contiguously in the order they are declared, starting at memory address `0x800`. Given the following code segment:

```
short x = 2;
short *px = &x;
short y = 7;
short *py = &y;
printf("starting x = %d\n", x);
printf("starting y = %d\n", y);
printf("a) %p\n", px);
printf("b) %p\n", py);
px = py;
printf("c) %d\n", *px);
printf("d) %p\n", &px);
*px = 25; y = -30;
printf("e) %d\n", *px);
printf("f) %d\n", *py);
printf("g) %d\n", x);
printf("h) %d\n", y);
```

Give the output of each `printf()` assuming the `%p` format specifier prints out a memory address in hexadecimal.

Problem 3

Assume 8-bit two's complement integer representation for this problem. Do all calculations by hand and show your work for full credit.

A. Give the bit sequence for the largest positive number and the most negative number. (I.e., TMax and TMin).

B. Give the base 10 integer equivalent value for TMax and Tmin.

For C-H, compute the value and answer these questions:

- What is the resulting bit sequence?
- What is the corresponding base 10 value?
- Did this computation result in a signed overflow? If yes, explain why.

C. 10010001 + 00110110

D. 00011001 + 01101011

E. 01100110 + 11111001

F. TMin + TMax

G. 0 - TMin

H. TMax * 2

Problem 4

In this problem, show your work for full credit. You can use a calculator for basic decimal arithmetic (addition, subtraction, multiplication, division), but show the operands and result of each such operation.

Convert the following hexadecimal numbers to octal (base 8).

A. $AA29_{16}$

B. $BF8A_{16}$

Convert the following numbers (bases show with subscripts) to decimal.

C. $25C0_{16}$

D. 88888_{9}

E. 1000100010001000_{2}

Problem 5

(Based on the textbook problem 2.87.) The 2008 version of the IEEE floating-point standard, named IEEE 754-2008, includes a 16-bit "half-precision" floating-point format. It was originally devised by computer graphics companies for storing data in which a higher dynamic range is required than can be achieved with 16-bit integers. This format has 1 sign bit, 5 exponent bits (k=5), and 10 fraction bits (n=10). The exponent bias is $2^{5-1} - 1 = 15$.

Fill in the table that follows for each of the numbers given, with the following instructions for each column:

- **Hex**: the four hexadecimal digits describing the encoded form.
- **M**: the value of the significand. This should be a number of the form x or x/y where x is an integer and y is an integral power of 2. Examples include 0, 67/64, and 1/256
- **E**: the integer value of the exponent.
- **V**: the numeric value represented. Use the notation x or $x \times 2^z$, where x and z are integers.

- **D**: the (possibly approximate) numeric value, as is printed using the `%f` formatting specification of `printf()`.

Example: to represent the number 7/8 we would have s=0, M=7/4, and E=-1. Our number would therefore have an exponent field of 01110_2 (decimal value of $15 - 1 = 14$) and a significand field of 1100000000_2, giving a hex representation `3B00`. The numerical value is 0.875. You need not fill in entries marked –.

Description	Hex	M	E	V	D
7/8	`3B00`	7/4	-1	7/8	0.875
-1				-1	-1.0
Smallest value > -2					
Smallest positive normalized value					
Number with hex `5BE2`	`5BE2`				
∞				–	–
`NaN`		–		–	–