Fall '17 CIS 314 Assignment 2 – 100/100 points – Due Monday, 10/9, 11:59 PM

Please submit individual source files for coding exercises (see naming conventions below) and a single solution document for non-coding exercises (.txt or .pdf only). Your code and answers need to be documented to the point that the graders can understand your thought process. Full credit will not be awarded if sufficient work is not shown.

1. [20] B&O'H 2.71.

You just started working for a company that is implementing a set of procedures to operate on a data structure where 4 signed bytes are packed into a 32-bit unsigned. Bytes within the word are numbered from 0 (least significant) to 3 (most significant). You have been assigned the task of implementing a function for a machine using two's-complement arithmetic and arithmetic right shifts with the following prototype:

```
/* Declaration of data type where 4 bytes are packed
   into an unsigned */
typedef unsigned packed_t;
/* Extract byte from word.  Return as signed integer */
int xbyte(packed_t word, int bytenum);
```

That is, the function will extract the designated byte and sign extend it to be a 32-bit int.

Your predecessor (who was fired for incompetence) wrote the following code:

```
/* Failed attempt at xbyte */
int xbyte(packed_t word, int bytenum)
{
   return (word >> (bytenum << 3)) & 0xFF;
}
```

A. What is wrong with this code?

B. Give a correct implementation of the function that uses only left and right shifts, along with one subtraction.

Example calls with a correct implementation:

xbyte(0x00007700, 1): 0x77
xbyte(0x00EE0000, 2): 0xFFFFFFEE

Also write a main() function to test your function. Name your source file 2-71.c and write your answer for part A in your comments.

2. [20] B&O'H 2.72.

You are given the task of writing a function that will copy an integer val into a buffer buf, but it should do so only if enough space is available in the buffer.

Here is the code you write:

```
/* Copy integer into buffer if space is available */
/* WARNING: The following code is buggy */
void copy_int(int val, void *buf, int maxbytes) {
   if (maxbytes-sizeof(val) >= 0)
         memcpy(buf, (void *) &val, sizeof(val));
}
```

This code makes use of the library function memcpy. Although its use is a bit artificial here, where we simply want to copy an int, it illustrates an approach commonly used to copy larger data structures.

You carefully test the code and discover that it *always* copies the value to the buffer, even when maxbytes is too small.

A. Explain why the conditional test in the code always succeeds. **Hint:** The sizeof operator returns a value of type size_t.

B. Show how you can rewrite the conditional test to make it work properly.

The above code doesn't no print anything; you should verify that memcpy is only being called when appropriate. For example:

copy_int(42, buffer, 4): memcpy should be called
copy_int(42, buffer, 2): memcpy should NOT be called

Also write a main() function to test your function. Name your source file 2-72.c and write your answer for part A in your comments.

3. [15] B&O'H 2.83.

Fill in the return value for the following procedure, which tests whether its first argument is less than or equal to its second. Assume the function f2u returns an unsigned 32-bit number having the same bit representation as its floating-point argument. You can assume that neither argument is *NaN*. The two flavors of zero, +0 and −0, are considered equal.

```
int float_le(float x, float y) {
   unsigned ux = f2u(x);
   unsigned uy = f2u(y);
```

```
    /* Get the sign bits */
    unsigned sx = ux >> 31;
    unsigned sy = uy >> 31;
    /* Give an expression using only ux, uy, sx, and sy */
    return _____; }
```

Use the following code for f2u:

```
unsigned f2u(float f) {
    return *((unsigned*)&f);
}
```

Example calls:

float_le(0.0f, 0.0f): 1
float_le(-0.0f, 0.0f): 1
float_le(-1.0f, 0.0f): 1
float_le(0.0f, 1.0f): 1
float_le(1.0f, 0.0f): 0
float_le(0.0f, -1.0f): 0

Also write a main() function to test your function. Name your source file 2-83.c.

4. [15] Convert the following hex values to decimal assuming that they are stored as 2s complement integers.

a. (5) 0x0000000F

b. (5) 0xFFFFFCE6

c. (5) 0xFFFFFFFF

Write your answers in your solutions document.

5. [15] Convert the following hex values to decimal assuming that they are encoded as IEEE 754 single-precision floating-point numbers:

a. (5) 0x00000000

b. (5) 0x41120000

c. (5) 0xC39D0000

Write your answers in your solutions document.

6. [15] Convert the following decimal numbers to hex encoded as IEEE 754 single-precision floating-point numbers. Write your answers in your solutions document.

a. (5) -1.0

b. (5) 10.5

c. (5) -85.125

Write your answers in your solutions document.

Zip the source files and solution document (if applicable), name the .zip file <Your Full Name>Assignment2.zip (e.g., EricWillsAssignment2.zip), and upload the .zip file to Canvas (see Assignments section for submission link).