# CS 354 Spring 2021
# Bitwise operations practice problems

1. Assume a system where an integer takes up one byte of memory. Suppose that x and y have byte values `0x83` and `0x4D`, respectively. Fill in the following table indicating the byte values of the different C expressions. All values must be written in hexadecimal notation. The first one is already answered for you! :)

| Expression | Value |
|---|---|
| `x & y` | `0x01` |
| `~x \| ~y` | |
| `x & !y` | |
| `!x \|\| !y` | |
| `x && ~y` | |
| `x << 3` | |
| `x >> 2 (arithmetic)` | |

2. The following function has a bug and doesn't work as expected. What is the **issue** with this function and how will you **fix** it?

```
// If x is greater than y, this function should return 1.
// Else, this function returns 0.
int is_greater(unsigned int x, unsigned int y)
{
    if (x - y > 0)
        return 1;
    else
        return
0;

}
```

**Issue:**

**Fix:**

3. What is the output of the code below on a 32-bit **little endian** and **big endian** machines?

```
#include <stdio.h>
int main()
{
   unsigned int i = 1;
   char *c = (char*)&i;
   printf("0x%x", *c);
   return 0;
}
```

**Little Endian: _____**

**Big Endian: _____**

4. Suppose that integers are **signed** and they take up only 1 byte. What is the **binary** representation of the following integers? You may assume that the machine uses two's complement representation to represent negative numbers. **[5 points]**

| Integer | Binary |
|---|---|
| -1 | |
| 4 | |
| 27 | |
| -14 | |
| 0 | |

5. What is the output of the following print statement? **[5 points]**

```
printf("%d, %d, %d, %d\n",
       mystery(1), mystery(-1), mystery(0), mystery(INT_MIN));

int mystery(int x) {
   return (x >> (8 * sizeof(int)-1)) & 0x1;
}
```

**OUTPUT:**

What is a suitable name for this function based on what it is doing?

**Suitable name:**