Name: Natalie Zoladkiewicz Date: 10/1/2023

Pledge: I pledge my honor that I have abided by the Stevens Honor System

For each function below, trace through it with reasonably small integer values. What does each function do?
Requirement: You should assume integers are only **8 bits** for the purpose of this exercise. The sign bit is the leftmost of the 8 bits.

```
int mystery1(int a, int b) {
    int c = a - b,
        d = (c >> 7) & 1,
        mystery = a - c * d;
    return mystery;
}
```

Trace: mystery1(3, 7) returns 7
Trace: mystery1(8, 7) returns 8

Summary: In the first step, the function takes in two integers and subtracts them and stores the result as c. Then d is defined as c shifted by 7 bits, which evaluates to -1. Next, we perform a bitwise AND operation and since it is true, this statement evaluates to 1. Then we calculate the value of mystery as 3 – (-4) * 1 and get 7. In the same way, the second trace starts by subtracting 8-7 to get 1. Then we perform the shifting, but this evaluates to 0. So the bitwise operation evaluates to 0 because 0 & 1 = 0. The last step is 8 – 1 * 0 which equals 8 by the order of operations.

```
int mystery2(int x) {
    return (x && !(x & (x - 1)));
}
```

Trace: mystery2(1) returns 1
Trace: mystery2(2) returns 1
Trace: mystery2(3) returns 0
Trace: mystery2(4) returns 1
Trace: mystery2(5) returns 0
Trace: mystery2(6) returns 0
Trace: mystery2(7) returns 0
Trace: mystery2(8) returns 1

Summary: In this program, the nested operation x – 1 happens first. Then the bitwise operation x AND x – 1 returns 1 if true and 0 if false. The ! operator then negates the result of the bitwise operation we just performed. Lastly, we perform AND (not bitwise) operation between x and the result of the negated bitwise operation.

```
int mystery3(int x, int y) {
    int s, c;
    s = x ^ y;
    c = x & y;
    while (c != 0) {
        c = c << 1;
        x = s;
        y = c;
```

```
        s = x ^ y;
        c = x & y;
    }
    return s;
}
```

Trace: mystery3(5, 7) returns 12
Trace: mystery3(2, 8) returns 10

Summary: This function takes integers x and y as input and first assigns the XOR operation of x and y to the variable s. Then it assigns the variable c to the bitwise operation of AND between x and y. Then there is a while loop that terminates as soon as c is equal to 0. Inside the loop, c is left-shifted by 1, the current value of s is assigned to x, the current value of c is assigned to y, there is another bitwise XOR operation of x and y, and there is another AND computation of x and y. The loop ends when c becomes 0 and the function returns the value of s.