Name: \_\_\_Joshua Schmidt\_\_\_\_\_\_ Date: \_\_\_\_\_9/18/2019\_\_\_\_\_

Pledge: *\_\_\_\_\_I pledge my honor that I have abided by the Stevens Honor System. - Joshua Schmidt\_\_\_\_*

For each function below, trace through it with reasonably small integer values. What does each function do?

**HINT:** You should assume integers are 8 bits for the purpose of this exercise.

**int** **mystery1**(**int** a, **int** b) {

**int** c = a - b,

d = (c >> 7) & 1,

mystery = a - c \* d;

**return** mystery;

}

3-7 = -4

c = 11111100

shift right 7:

d = 11111111 &1 => 00000001

mystery = 3 - (-4) \* 1

Trace: mystery1(3, 7) returns \_\_7\_\_

c = 8 – 7 = 1 = 00000001

d = 11111110 &1 => 0

mystery = 8 – (8-7) \* 0 = 8 – 0 = 8

Trace: mystery1(8, 7) returns \_\_8\_\_

Summary: \_This function returns b if a < b, else it returns a.\_

**void** **mystery2**(**int** values[], **int** i, **int** j) {

values[i] = values[i] ^ values[j];

values[j] = values[i] ^ values[j];

values[i] = values[i] ^ values[j];

}

Note: Improper C++ syntax found below.

[1, 2, 3, 4]

first index = 00000001 ^ 00000100 = 00000101

last index = 00000101 ^ 00000100 = 00000001 = 1

first index = 00000101 ^ 00000001 = 00000100 = 4

Trace: mystery2([1, 2, 3, 4], 0, 3) values = [ \_4\_, \_2\_, \_3\_, \_1\_ ]

[1, 2, 3, 4]

second index = 00000010 ^ 00000011 = 00000001

third index = 00000001 ^ 00000011 = 00000010 = 2

second index = 00000001 ^ 00000010 = 00000011 = 3

Trace: mystery2([1, 2, 3, 4], 1, 2) values = [ \_1\_, \_3\_, \_2\_, \_4\_ ]

Summary: \_This function takes an array as an input and swaps the values of the two indexes given.

**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;

}

s= 5 ^ 7 = 00000101 ^ 00000111 = 00000010 = 2

c = 5 & 7 = 00000101 & 00000111 = 00000101 = 5

first loop:

c = 00001010 = 10

x = s = 2

y = c = 10

s = x ^ y = 00000010 ^ 00001010 = 00001000 = 8

c = x & y = 00000010 & 00001010 = 00000010 = 2

second loop:

c = 00000100 = 4

x = s = 00001000 = 8

y = c = 00000100 = 4

s = x ^ y = 00001000 ^ 00000100 = 00001100 = 12

c = x & y = 00001000 & 00000100 = 0

return s = 12

Trace: mystery3(5, 7) returns \_\_12\_\_

s = 2 ^ 8 = 00000010 ^ 00001000 = 00001010

c = 2 & 8 = 00000010 & 00001000 = 00000000

doesn’t enter loop

return s = 10

Trace: mystery3(2, 8) returns \_\_10\_\_

Summary: \_\_\_This is an adder. It just adds the two values given.\_\_\_\_