3) (10 pts) DSN (Bitwise operators)

Two useful utility functions when dealing with integers in their binary representation are

(a) `int lowestOneBit(int n)` - returns the value of the lowest bit set to 1 in the binary representation of n. (eg. `lowestOneBit(12)` returns 4, `lowestOneBit(80)` returns 16.)

(b) `int highestOneBit(int n)` - returns the value of the highest bit set to 1 in the binary representation of n. (eg. `highestOneBit(12)` returns 8, `highestOneBit(80)` returns 64.) **Note: You may assume that the input is less than 10^9. The largest positive bit value in an integer is equal to $2^{30} > 10^9$.**

The pre-condition for the first function is that n must be a positive integer. The pre-condition for the second function is that n must be a positive integer less than 10^9. Write both of these functions in the space below. To earn full credit, you ***must*** use bitwise operators when appropriate. (Namely, there are ways to solve this question without using bitwise operators, but these solutions will NOT receive full credit.)

```
int lowestOneBit(int n) {

    int res = 1;
    while ((res & n) == 0)
        res = res << 1;
    return res;

}

int highestOneBit(int n) {

    int res = 1;
    while (((res<<1) <= n) && (res<<1) > 0)
        res = res << 1;
    return res;

}
```

Grading:

lowestOneBit - 2 pts for selecting a single bit at a time somehow, 2 pts for going in order from lowest to highest, 1 pt for returning the correct answer.

highestOneBit - 2 pts for selecting a single bit, 2 pts for a valid method to find the most significant one, 1 pt for returning the correct answer. Note - there is no need to check if (res<<1) > 0, this is unnecessary if $n < 10^9$.

Note: There are other ways of writing these functions. Please carefully trace through all student solutions and map points from the criteria above as best as possible.