Due date: January 25 at 7:30 AM. You will need to submit this via GradeScope.

1. An undirected graph is called d-regular if it is a simple graph and, for every vertex v, $\delta(v) = d$. For example, in a 3-regular graph, every vertex has three neighbors.

   (a) Either draw a 3-regular graph that has exactly nine (9) vertices, or explain why it cannot be done.

   (b) Either draw a 3-regular graph that has exactly twelve (12) vertices, or explain why it cannot be done.

   (c) Suppose I have a d-regular graph and I want to find a path that contains $d+1$ (or more) vertices. It turns out this is easy to do: pick an arbitrary start vertex. Until I have a path with $d+1$ vertices, pick an adjacent vertex to my current one that I have not yet visited. Add that edge to my growing path and set my current vertex to that one.

   Explain in 1-2 sentences why this will always produce a path with at least $d+1$ vertices. How do I know I won't "get stuck" at a vertex until I have visited $d+1$ vertices?

2. Give the asymptotic complexity for the worst case run-time of the following function `foo()` with respect to its input N. Function `bar()` runs in constant time with respect to N. For full credit you should show how you arrived at your answer. An answer that solely provides a value in Big O notation without reason will receive zero points. Answers that include variables other than N will receive fewer points.

```

void foo(unsigned int N) {
    unsigned int S = 0;

    for (unsigned int i = 1; i <= N; i++) {
        S += i;
    }

    for (unsigned int j = 1; j <= S; j++) {
        bar();
    }
}
```

3. *You will probably want to wait until after lecture on January 23 before attempting this problem.* Suppose we have a counter that stores an arbitrary number of bits and counts in binary. It always begin at 0. The only mutator operation it can perform is to increment, adding one to the current count. This changes one or more bits. **Show** that if we start at 0 and perform k increment operations, a total of $\mathcal{O}(k)$ bits will change. Correct answers will reason about the **asymptotic** behavior of the total number of bit changes.

   Hint: There are multiple ways to approach this problem. One approach could use a credit argument, and you should refer to the array expansion of array-based Stacks and Queues for inspiration here. Another approach is numerical analysis. Try to find patterns in the total number of bit changes as k increases and inductively reason about the overall asymptotic behavior.