1. Here is another variation on the code from your pre-tutorial exercise.

```java
int blah = 0;
for (int p = 1; p <= n; p++) {
    for (int q = 1; q <= n; q++) {
        for (int r = p; r <= q; r++) {
            blah = blah + r;
        }
    }
    blah = blah + p;
    System.out.println("Hello");
}
```

What is the big-oh time complexity of this code, in the worst case?

2. Can you write a loop that is both $O(n)$ and $O(n^2)$? If yes, write one; if no, explain why not.

3. Let $T(n) = 5n + (4 \times \log_3 n) - 7$. Which of the following are true? Circle all that apply.

   - $T(n)$ is $O(1)$
   - $T(n)$ is $O(\log n)$
   - $T(n)$ is $O(n)$
   - $T(n)$ is $O(n \log n)$
   - $T(n)$ is $O(n^2)$

4. Let $S(n) = (n + 1)(n + 2)$. Which of the following are true? Circle all that apply.

   - $S(n)$ is $O(1)$
   - $S(n)$ is $O(\log n)$
   - $S(n)$ is $O(n)$
   - $S(n)$ is $O(n \log n)$
   - $S(n)$ is $O(n^2)$
5. If we know that a sorting algorithm named sillySort is $O(n^2)$, then we know that ...
   (Circle exactly one answer)
   - sillySort is $O(n \log n)$.
   - sillySort is not $O(n \log n)$.
   - sillySort is $O(n^3)$.
   - sillySort is not $O(n^3)$.

6. What is the worst-case time complexity of each of the following, assuming that good algorithms are used?
   - Computing the sum of all the elements in an $n \times n$ two-dimensional array.
   - Printing all elements from the lower triangle of an $n \times n$ two-dimensional array.
   - Inserting an item into an unsorted linked list of $n$ elements.
   - Inserting an item into a sorted linked list of $n$ elements.
   - Pushing an item onto a stack of $n$ elements that is stored as an array.
   - Pushing an item onto a stack of $n$ elements that is stored as a linked list.

7. Write a Java method that has 2 loops and an if-statement and whose worst-case big-oh time complexity is $O(n + m)$
   ```java
   public static void whatever(int n, int m)
   ```