1. Give the asymptotic time complexity, in terms of $n$, of each of these code fragments. (10 points each)
   
   (a) 
   ```java
   for(int i = 1; i < n; i=2*i)
   for(int j = 1; j < i; j++)
   ```
   
   (b) 
   ```java
   for(int i = 1; i < n; i=2*i)
   for(int j = i; j < n; j++)
   ```
   
   (c) 
   ```java
   for(int i = 1; i < n; i++)
   for(int j = i; j > 0; j = j/2)
   ```
   
   (d) 
   ```java
   for(int i = 1; i < n; i++)
   for(int j = n; j > i; j = j/2)
   ```
   
   (e) 
   ```java
   for(int i = 1; i*i < n; i++)
   ```
   
   (f) 
   ```java
   for(int i = 1; i*n < n; i++)
   ```
   
   (g) 
   ```java
   for(int i = 1; i < n; i++)
   for(int j = 0; j < n; j = j+i)
   ```
   
   (h) This problem requires two answers. Its time complexity is not $\Theta$ of any of the usual functions we deal with. Instead, it’s $\Omega$ of some function of $n$ and $O$ of some other function of $n$. Give both.
   ```java
   for(int i = 2; i < n; i=i*i)
   for(int j = 1; j < i; j++)
   ```

2. Give asymptotic solutions to the following recurrences.
   
   (a) $F(n) = F(n/2) + F(n/3) + n$;
   
   (b) $G(n) = G(n/4) + 2G(n/16) + \sqrt{n}$;
   
   (c) $H(n) = H(n - \log n) + \log n$

3. [10 points] Draw an acyclic directed graph of 6 vertices and 15 arcs.

4. [10 points] Draw a directed graph with exactly two strong components, each of which has 4 vertices. The graph must have a “source” vertex $s$ from which every vertex is reachable.

5. [10 points] Draw a planar graph with 5 vertices and 10 edges.

Write pseudocode for the Bellman-Ford algorithm. Be sure to incorporate the shortcut.

7. [20 points] If you need to solve the all-pairs problem for a weighted graph with \( n \) nodes and \( m \) edges, which algorithm would you use?
8. [20 points] Write the Polish and reverse Polish expressions equivalent to \( a \ast (-b - c) \ast d \).

9. [20 points] Prove that there is no comparison-based algorithm for sorting six items that never uses more than nine comparisons.

10. [20 points]
    I made a mistake writing this code in Part I of the practice final. Here is the correct version.

    ```c
    int product(int a, int b)
    {
        assert(b >= 0);
        int c = a;
        int d = b;
        int total = 0;
        while(d > 0)
        {
            if(d%2) total = total + c;
            c = 2*c;
            d = d/2;
        }
        return total;
    }
    ```

    (a) What does this function do?

    (b) What is the loop invariant of the while loop?