1. (10) Prove that $\sum_{k=1}^{n} \frac{1}{k} \leq \log n + 1$.

2. (10) Let $f(n) = \log(n!)$ and $g(n) = (\log(n))!$.
   (a) Is $f(n) \in O(g(n))$?
   (b) Is $g(n) \in O(f(n))$?

3. (10) Let $c$ be a positive number. Show that $f(n) = 1 + c + c^2 + \cdots + c^n$ is
   (a) $\Theta(1)$ if $c < 1$,
   (b) $\Theta(n)$ if $c = 1$,
   (c) $\Theta(c^n)$ if $c > 1$.

4. (10) Show that $\log(n!) = \Theta(n \log n)$.
   (Hint: To show an upper bound, compare $n!$ with $n^n$. To show a lower bound, compare it with $(n/2)^{n/2}$.)

5. (15) Let $a$ and $b$ be two $n$-bit integers, and $n$ is very large.
   (a) Is it possible to design an algorithm for computing $a^2$ which is asymptotically faster than computing $a \times b$?
   (b) Is it possible to design an algorithm for computing $a \times b$ which is asymptotically faster than computing $a^2$?
   (c) Supposed that someone has shown that a program for computing $a^2$ takes less time than the best program for computing $a \times b$. How do you explain this?

6. (20) Give asymptotically tight upper bounds $T(n)$ for each of the following recurrences. Justify your answers.
   (a) $T(n) = 2T(n/2) + n$
   (b) $T(n) = 9T(n/4) + n^2$
   (c) $T(n) = T(\sqrt{n}) + 1$
   (d) $T(n) = T(n/3) + T(n/5) + n^3$

7. (20) Consider the following program for computing the greatest common divisor of two positive integers $a$ and $b$.

   **while** $(b > 0)$ $\{r = a\%b;\ a = b;\ b = r;\}$ print$(a)$;

   (a) Show that the program will eventually stop for any input $a, b > 0$.
   (b) Show that the number of iterations for the **while** loop is bounded by $O(\log_2 n)$.

8. (20) Efficient algorithm for the multiplication of very large integers can be designed using divide and conquer approach. Let $x = x_l 2^{n/2} + x_r$, and $y = y_l 2^{n/2} + y_r$. Then $xy = (x_l 2^{n/2} + x_r)(y_l 2^{n/2} + y_r) = x_l y_l 2^n + (x_l y_r + x_r y_l) 2^{n/2} + x_r y_r$. Note that $x_l y_r + x_r y_l = (x_l + x_r)(y_l + y_r) - x_l y_l - x_r y_r$. We can do it with only 3 multiplications of length $n/2$. Recursively apply this strategy, we have: $T(n) = 3T(n/2) + O(n)$. Thus, $T(n)$ is $O(n^{\log_2 3})$, and $log_2 3 \approx 1.5849625 < 2$. Let $U(n)$ be the time required to do the *additions*, not multiplications. Give a tight bound for $U(n)$ for the divide and conquer approach described above.