1. On each row of the table below there is some function \( f(n) \). Assume that \( f(n) \) is the time in milliseconds taken by some algorithm for input size \( n \). For each of the time limits given in the columns of the table, calculate the largest input that can be processed by the algorithm within this time.

<table>
<thead>
<tr>
<th>time</th>
<th>1 second</th>
<th>1 minute</th>
<th>1 hour</th>
<th>24 hours</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \log_2 n )</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( n )</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( n \log_2 n )</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( n^2 )</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( 2^n )</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

2. Which of the following are true, which false? Justify your answers.
   (a) \( \log_2(n^2) = O(\log n) \)
   (b) \( (\log_2 n)^2 = O(\log n) \)
   (c) \( 5n^3 + 2n^2 + 1000 = O(n^3) \)
   (d) \( 5n^3 + 2n^2 + 1000 = \Theta(n^3) \)
   (e) \( 5n^3 + 2n^2 + 1000 = O(n^4) \)
   (f) \( 5n^3 + 2n^2 + 1000 = \Omega(n^4) \)
   (g) \( 10^{18} = O(1) \)

3. Suppose we have a party with lots of people. We shall say that a person at the party is a celebrity if everyone knows him, but he does not know anyone else. Present a method for finding a celebrity at a party, or deciding that no celebrities are present, by making questions of the form “Does person \( x \) know person \( y \)?” Your method should need only \( O(n) \) questions, where \( n \) is the number of people in the party. Explain why your method works.

4. The following algorithm evaluates the polynomial function \( p(x) = a_0 + a_1x + a_2x^2 + \ldots + a_kx^k \), when the constant coefficients \( a_0, \ldots, a_k \) are stored in an array \( A[0..k] \):

   ```
   EVALUATE-POLYNOMIAL(A[0..k], x)
   1   s ← 0
   2   for i ← 0 to k
       do
   3       z ← 1
   4       for j ← 1 to i
       do
   5               z ← z \cdot x
   6       s ← s + A[i] \cdot z
   7   return s
   ```

   What is the running time of the algorithm? (Assume arithmetic operations take a constant time.) What is the loop invariant for the outer for loop?

   **Continued on next page!**
5. The algorithm given in the previous problem for evaluating a polynomial function could be made more efficient. The outline for a more efficient algorithm would be

\[
\text{Evaluate-Polynomial-2}(A[0..k], x)
\]

\[
\begin{align*}
\ldots \\
\text{for } i &← 0 \text{ to } k \\
\text{do} \\
\quad \triangleright z = x^i \text{ and } s = \sum_{j=0}^{i-1} A[j] x^j \\
\ldots \\
\text{return } s
\end{align*}
\]

where a new loop invariant has been written in as a comment. Fill in the missing parts of the algorithm so that the loop will indeed have the given invariant.