1. We know that a certain algorithm takes 0.5 ms to run for input size \( n = 100 \). How long does it take for input size 500, when the running time of the algorithm is

(a) \( \Theta(n) \)
(b) \( \Theta(n \log n) \)
(c) \( \Theta(n^2) \)
(d) \( \Theta(n^3) \)
(e) \( \Theta(2^n) \)?

In each case assume that the lower order terms can be neglected. For example, in part (d) assume that the running time is \( cn^3 \) for some constant \( c \).

2. An algorithm takes 0.5 ms for input size 100, like in the previous problem, and we consider the same possible asymptotic running times (a)–(e). For each of these possibilities, what is the largest input size for which the algorithm runs in under one minute?

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 \( p(x) = a_0 + a_1 x + a_2 x^2 + \ldots + a_k x^k \), when the constant coefficients \( a_0, \ldots, a_k \) are stored in an array \( A[0 \ldots k] \):

\[
\text{EVALUATE-POLYNOMIAL}(A[0 \ldots k], x)
\]

1. \( s \leftarrow 0 \)
2. \( \text{for } i \leftarrow 0 \text{ to } k \)
   \( \quad \text{do} \)
3. \( z \leftarrow 1 \)
4. \( \text{for } j \leftarrow 1 \text{ to } i \)
   \( \quad \text{do} \)
5. \( z \leftarrow z \cdot x \)
6. \( s \leftarrow s + A[i] \cdot z \)
7. \( \text{return } s \)

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?

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

\[
\text{EVALUATE-POLYNOMIAL-2}(A[0 \ldots k], x)
\]

\[
\text{for } i \leftarrow 0 \text{ to } k \)
\( \quad \text{do} \)
\( \quad \triangleright z = x^i \text{ and } s = \sum_{j=0}^{i-1} A[j] x^j \)
\( \quad \text{return } s \)

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.