1. The Big-Oh notation is a convenient device for simplifying upper bounds. A useful device for dealing with lower bounds is the Big-Omega notation. Consider nonnegative functions \( f(N) \) and \( g(N) \) of a natural number \( N \). We say \( f(N) \) is \( \Omega(g(N)) \) and write \( f(N) = \Omega(g(N)) \) if there exist positive constants \( K \) and \( N_0 \) such that
\[
f(N) \geq Kg(N) \quad \text{for all} \quad N \geq N_0.
\]
(a) Verify that \( f(N) = \Omega(g(N)) \) if and only if \( g(N) = O(f(N)) \).
(b) Show that \( N^3 - 10N^2 + 10,000 = \Omega(N^3) \).
(c) Consider the following conditions, where \( K \) is a positive constant:
\[
(i) \lim_{N \to \infty} \frac{f(N)}{g(N)} = \infty \quad (ii) \lim_{N \to \infty} \frac{f(N)}{g(N)} = 0 \quad (iii) \lim_{N \to \infty} \frac{f(N)}{g(N)} = K
\]
Which of
\[
(iv) f(N) = \Omega(g(N)) \quad (v) g(N) = \Omega(f(N))
\]
\[
(vi) f(N) \neq \Omega(g(N)) \quad (vii) g(N) \neq \Omega(f(N))
\]
are implied by each of (i), (ii), (iii)? (There is no need to supply proofs.)
(d) Assuming \( \lim_{N \to \infty} \frac{g(N) + f(N)}{N \log N} \) exists, use limits to verify that if \( f(N) = O(N) \) and \( g(N) + f(N) = \Omega(N \log N) \) then \( g(N) = \Omega(N \log N) \).

2. Let \( x, y, z \) be real numbers and put
\[
A = \begin{bmatrix} 1 & 1 & 1 \\ x & y & z \\ x^2 & y^2 & z^2 \end{bmatrix}.
\]
(a) Verify that \( \det A = (y-x)(z-x)(z-y) \). Deduce that \( \det A > 0 \) if and only if either \( x < y < z \) or \( y < z < x \) or \( z < x < y \).
(b) Now deduce that if \( n \geq 3 \) and \( x_1, \ldots, x_n \) are positive real numbers then the points
\[
(x_1, x_1^2), (x_2, x_2^2), \ldots, (x_n, x_n^2)
\]
are vertices of a convex polygon in standard form if and only if
\[
x_1 < x_2 < \ldots < x_n.
\]
3. It is an important theorem in algorithm analysis that all sorting algorithms have running time

\[ T(N) = \Omega(N \log N) . \]

(The proof uses decision trees, and will not be given in this course.) Hence it is impossible to improve on \( O(N \log N) \) as the running time of sorting algorithms. Thus, for example, MERGESORT, whose running time is \( O(N \log N) \), is an optimal sorting algorithm. The purpose of this exercise is to use the above theorem to prove an analogous result for convex hull algorithms.

Suppose \( C \) is a convex hull algorithm which takes as input a finite set \( X \) of points in the plane and outputs the list of vertices of the boundary of \( \text{HULL}(X) \) as a convex polygon in standard form. Let \( S \) be the following algorithm which uses \( C \) to sort a finite set of \( N \) distinct positive reals. To avoid trivial cases suppose that \( N \geq 3 \).

**Algorithm \( S \):**

**Step (1)** Input \( Y = \{x_1, x_2, \ldots, x_N\} \) where \( x_1, x_2, \ldots, x_N \) are distinct positive reals.

**Step (2)** Form \( X = \{(x_1, x_1^2), (x_2, x_2^2), \ldots, (x_N, x_N^2)\} \).

**Step (3)** Apply \( C \) to \( X \) to output the vertices

\[ (y_1, y_1^2), (y_2, y_2^2), \ldots, (y_N, y_N^2) \]

of the boundary of \( \text{HULL}(X) \) as a polygon in standard form.

**Step (4)** Halt and output the sequence \( y_1, y_2, \ldots, y_N \).

(a) Use 2(b) to explain why in **Step (3)**, \( C \) is guaranteed to output \( N \) points (not fewer).

(b) Let the running time of \( S \) be \( T_S(N) \) and of \( C \) be \( T_C(N) \). Explain why

\[ T_S(N) = T_C(N) + O(N) . \]

(c) Use the theorem quoted in the preamble to this question and 1(d) to deduce that

\[ T_C(N) = \Omega(N \log N) . \]

(Assume that the relevant limit exists when you apply 1(d).)

This establishes that the convex hull algorithm in lectures, which has \( O(N \log N) \) running time, is optimal.