1. Write a function `innerprod` that takes two vectors \( v \) and \( w \) represented by lists and returns the inner product. The inner product of two vectors is defined as \( \vec{v} \cdot \vec{w} = v_1 \cdot w_1 + v_2 \cdot w_2 + \ldots v_n \cdot w_n \). For example, if \( v = [3, 5, 0, 2] \) and \( w = [2, 3, 1, 4] \), then the product of \( v \) and \( w \) is equal to \( 3 \cdot 2 + 5 \cdot 3 + 0 \cdot 1 + 2 \cdot 4 = 6 + 15 + 0 + 8 = 29 \). The input vectors must have equal length. Otherwise the return value must be -1.

2. Implement the sieve of Eratosthenes in Haskell. This algorithm determines all the prime numbers in a range of numbers by removing all the multiples of 2, 3, 5, 7, ... from the range. What is left in the range are only prime numbers. For example, for the range \([2..20]\) (1 is not a prime number), we would first remove all multiples of 2 and are left with \([2, 3, 5, 7, 9, 11, 13, 15, 17, 19]\). In the next step we remove multiples of 3 and are left with \([2, 3, 5, 7, 11, 13, 17, 19]\). Once we reach a number whose first multiple is larger than 20, we stop.

3. (a) Write a function `qs_lol` which employs the quicksort algorithm (see exercise sheet 5), gets a list of lists as an input and returns a list of lists as an output, such that each outputted list is sorted using the values of the last list as sort keys. For example, the input

\[
[[0,1,2],[23,26,30],[3400,1700,5000]]
\]

should result in the output:

\[
[[1,0,2],[26,23,30],[1700,3400,5000]]
\]

(b) Write two functions `qs_lc_tuple_f` and `qs_lc_tuple_l` which employ the quicksort algorithm, get a list of tuples of the form \((\text{Int}, \text{Int})\) and return the list sorted in ascending order of the first element or the last element respectively.
For example,

\[
\begin{align*}
\text{qs_lc_tuple_f} & \quad [(5,1),(6,4),(2,8),(4,2)] \\
& \quad [(2,8),(4,2),(5,1),(6,4)]
\end{align*}
\]

and

\[
\begin{align*}
\text{qs_lc_tuple_l} & \quad [(5,1),(6,4),(2,8),(4,2)] \\
& \quad [(5,1),(4,2),(6,4),(2,8)]
\end{align*}
\]

You can use the build-in functions: \texttt{minimum}, which returns the minimum element of a list, and \texttt{delete}, which deletes the first occurrence of an element from a list. In order to use \texttt{delete} you have to import the module \texttt{Data.List}.

4. A point \((x, y)\) dominates another point \((x', y')\) if we both have \(x < x'\) and \(y < y'\). Write a function \texttt{nonDom} that computes for a given a set of points (with positive integer coordinates), which points are not dominated by any other point in the given set. For example, \texttt{nonDom} \([(2,3),(1,5),(3,4),(1,4)]\) returns the set \([(1,4),(2,3)]\).