<u>Q.1.3</u> - One possible algorithm for computing the convex hull of a set of convex polygons in time O(Nlog(k)) is the **divide and conquer algorithm**. This algorithm works by recursively dividing the set of polygons into smaller sets and computing the convex hull of each set, until there is only one polygon left in each set. Then, the convex hulls of the individual sets are combined to obtain the final result.

To achieve a time complexity of **O(Nlog(k))**, we can use a binary tree structure to divide the polygons into smaller sets at each step of the recursion. This allows us to split the polygons into two halves at each step, which results in a logarithmic number of recursive steps. Then, we can use a linear-time algorithm, such as the gift wrapping algorithm, to compute the convex hull of each set of polygons at each recursive step.

Here is a pseudocode outline of the algorithm:

```
function convex_hull(polygons):
  # base case: return the convex hull of a single polygon
  if polygons.length == 1:
    return compute_convex_hull(polygons[0])

  # divide the polygons into two sets using a binary tree structure
  left_polygons = polygons[0...polygons.length/2]
  right_polygons = polygons[polygons.length/2...polygons.length]

  # compute the convex hulls of the two sets of polygons recursively
  left_hull = convex_hull(left_polygons)
  right_hull = convex_hull(right_polygons)

  # combine the convex hulls of the two sets to obtain the final result
  return combine_convex_hulls(left_hull, right_hull)
```

This algorithm has a time complexity of **O(Nlog(k))**,where N is the total number of vertices in the input polygons and k is the number of polygons. This is because each recursive step divides the polygons into two smaller sets and then computes the convex hull of each set in linear time, resulting in a logarithmic number of recursive steps. Additionally, the combine_convex_hulls function can be implemented to run in linear time. To combine the convex hulls of two sets of polygons in linear time, we can use the **gift wrapping algorithm** (also known as Jarvis march) to compute the convex hull of the union of the two sets. This algorithm works by iteratively selecting points that are part of the convex hull, starting from the leftmost point. At each step, it selects the point that is the farthest to the right in the direction of the current hull edge. The algorithm continues until it reaches the starting point, at which point the convex hull is complete.