1. Homework Guidelines

(1) Each question (part of a question) should be written on a separate sheet so that the TAs can split the assignment and grade each problem separately.
(2) Group work is encouraged; however, your answers should be written up separately. Whenever you discuss a problem with someone else or other source that source must be noted as a collaborator for the problem. This includes professors, TAs, websites. The book does or class notes do not need to be referenced.
(3) There are typically two types of problems: proof problems and algorithm problems.
   (a) Proof Problems: For these questions you should often use the definitions or equivalent formulations for the definition - algorithms will often not be helpful for these problems.
   (b) Algorithm Problems: More questions will be of the form of algorithm problems. A complete solution will discuss the following parts of the problem:
      - The input and output of the algorithm: the type of data, points, lines, etc., if the data is sorted, the data structure holding the data (only if you need to be able to use specific operations), and anything else someone reading the solution should know.
      - The algorithm should not be described in pure pseudocode - but should be described in a series of clear steps. For instance, part of a solution could say: sort the data points by x-coordinate, then find the point with the smallest x-coordinate. Any algorithms discussed in Algorithms (comp160) or in class does not need to be described in the homework, for instance a solution could say: find the convex hull of the data set (you don't even need to be specific about which algorithm you are using unless it is necessary). However, if you only use part of an algorithm or a data structure from an algorithm, you should discuss what parts you use.
      - The algorithm should be shown to be correct. Only the most naïve algorithms do not need a correctness discussion. Typically, the more problem specific information that you use, the more you will need to prove - any additional information that you use in designing the algorithm should have some sort of justification.
      - For whichever algorithm you present, an analysis of the time complexity must be provided. The analysis of the algorithm should be as tight as possible.
      - If you create any new data structures they should be described - what operations can be performed and their time complexities.
      - For algorithm questions, a working algorithm is worth credit even if it is not optimal (or with the same asymtotics as the question). However, both the algorithm and the correctness are important and a complete solution needs both parts.
(4) These descriptions are just guidelines and for each problem more or less information may be necessary.