Worth: 2% **Due:** By 9:59pm on **Wednesday 29 October**

Remember to write your *full name* and *student number* prominently on your submission.

---

*Please read and understand the policy on Collaboration given on the Course Information Sheet. Then, to protect yourself, list on the front of your submission **every** source of information you used to complete this homework (other than your own lecture and tutorial notes, and materials available directly on the course webpage). For example, indicate clearly the **name** of every student with whom you had discussions, the **title** of every additional textbook you consulted, the **source** of every additional web document you used, etc.*

*For each question, please write up detailed answers carefully. Make sure that you use notation and terminology correctly, and that you explain and justify what you are doing. Marks **will** be deducted for incorrect or ambiguous use of notation and terminology, and for making incorrect, unjustified, ambiguous, or vague claims in your solutions.*

You have a network of wireless sensors that you would like to make more reliable, by selecting some number of "backup" sensors for each sensor in the network. Formally, consider the following problem.

Input: Sensors $s_1, s_2, \ldots, s_n$ (each sensor s_i has real-number coordinates (x_i, y_i)), distance parameter $d \in \mathbb{R}^+$, redundancy parameter $r \in \mathbb{Z}^+$, and backup parameter $b \in \mathbb{Z}^+$ with $b \geqslant r$.

Output: Backup sets $B_1, \ldots, B_n$ where $B_j \subseteq \{s_1, \ldots, s_n\} - \{s_j\}$ for each j, every sensor in B_j is within distance d of s_j, every B_j contains at least r elements, and every sensor belongs to at most b backup sets. (If this is not possible, output the special value NIL.)

Give an efficient algorithm to solve this problem, based on network flow techniques. Write a detailed justification that your algorithm is correct (in particular, explain how backup sets and flows correspond to each other) and analyze the worst-case running time of your algorithm. (HINT: Consider using *two* nodes for each sensor.)