Instance Simplification Algorithms

- Read section 6.1 (pages 201-205)
- The general theme of chapter 6 is **transform-and-conquer** techniques. These algorithms are divided into three types. What are those three types? What kind of transformation do we perform in each type?
- What is **presorting**? How is it different from sorting?
- The brute force solution to looking for the maximum element in an array has running time $O(n)$. How would a presort-based algorithm work? Would this problem benefit from a presorting approach?
- Describe the presort-based algorithm for *element uniqueness*. What is the running time of this algorithm?
  - What is the brute-force approach to this algorithm? How do the running times of the two approaches compare?
- Describe the presort-based algorithm for finding the element in a list with the highest multiplicity (mode).
  - What do the indices `i`, `modefrequency`, `runlength` and `runvalue` represent?
  - What is the role of the `if runlength > modefrequency` check?
  - What does the assignment `i <– i + runlength` do? Shouldn't it be `i <– i + 1`?
  - What is the running time of this algorithm?
- Practice problems: 6.1.1, 6.1.5
- Challenge: 6.1.8, 6.1.9