Heaps

- Read section 6.4 (pages 226-232)
- What are the three main operations of a **priority queue**?
- What are the two properties that a **heap** has? How do heaps differ from binary search trees?
  - What are the differences between "max-heaps" and "min-heaps"?
- We consider heaps as arrays with values starting at index 1. How do the indices in the array correspond to the tree structure of the heap?
  - How can we write the *shape property* in terms of the array entries?
- Study the **bottom-up heap construction** algorithm and figure 6.11.
  - Why is it important that the for loop move in the reverse direction?
  - Why does the for loop start from the middle of the array?
  - Each step of the for loop is often also called a "percolate down" operation. Explain why that is.
  - What is the role of the while loop? why is there not a single step for each i?
  - Roughly how many comparisons does the bottom-up construction require?
- How does the **top-down heap construction** algorithm work?
  - The key step in this algorithm is a "bubble up" operation. Explain why that is.
  - Make sure you understand how this algorithm differs from the bottom-up approach.
- How do we **delete** a key (in particular the maximum key) from the list?
- How does **heapsort** work? What is its time efficiency?
  - What type of transform-and-conquer does heapsort represent?
- Implement the heapsort algorithm to sort the letters of the word EXAMPLE.
- Practice problems: 6.4.1, 6.4.2, 6.4.3, 6.4.5, 6.4.6