Divide and Conquer Algorithms: Mergesort

- Read 5.1
  - What is the main idea of the divide-and-conquer technique? How does it differ from the decrease-and-conquer approach?
  - What is the general form of the recurrence relation for divide-and-conquer algorithms? What do the parameters a, b and $f(n)$ represent?
  - What does the **master theorem** say about a function $T(n)$ that satisfies the general recurrence relation?
    - The three cases of the master theorem are written in terms of comparing a and b^d. Write them instead in terms of comparing $\log_b a$ and d.
  - If a recursive algorithm needs to solve 3 subproblems of half the size, and it takes a constant time ($\Theta(1)$) to put together the final answer, then what does the master theorem tell us about the runtime $T(n)$ of this algorithm?
  - Describe how the MergeSort algorithm sorts an array.
    - Is MergeSort stable? Is it in-place?
    - What does the Merge subprocess do? What do the indices i, j, k represent?
    - Do we need to use the index k for the Merge process, or can we compute the needed value from i and j?
    - Explain the meaning of the "copy" phase of the Merge algorithm.
    - Use the MergeSort algorithm to sort the characters in the word EXAMPLE.
  - Determine the runtime of MergeSort:
    - Start with a recurrence relation for the runtime $C(n)$ of MergeSort, in terms of the runtime $C_{\text{merge}}(n)$ of the Merge process.
    - Determine the runtime of the (non-recursive) Merge process.
    - Use the master theorem to determine the runtime $C(n)$ of MergeSort.