Data structures

1. Arrange the following values in a heap 15, 14, 3, 7, 12, 6, 17, 2 by always inserting the next number in the root. Is the array with values \(\langle 23, 17, 14, 13, 10, 1, 5, 7, 12 \rangle\) a max-heap? Is an array that is sorted in order a max-heap? What is the minimum and maximum number of elements in a heap of height \(h\)?

2. In the previous problem, you had to build a heap by always inserting the next number in the root. Now assume that an unordered array of numbers is given. How would you go about building a heap from this array?

3. How would you use a heap to sort a list of numbers?

4. What is the running time of HeapSort on an array that is sorted in decreasing order? What about an array with numbers in increasing order?

5. If you add or delete an item from a heap, what do you need to do to maintain the heap-property?

6. (CLRS 9.1-1) Show that the second smallest of \(n\) elements can be found with \(n + \lceil \lg n \rceil - 2\) comparisons in the worst case.

7. (CLRS 18.2-1) Show the results of inserting the keys


   in order into an empty B-tree with minimum degree 2.

8. (CLRS 18.2-3) Explain how to find the minimum key stored in a B-tree and how to find the predecessor of a given tree stored in a B-tree.

9. Let \(B\) be a B-tree where every node contains at least \(t\) and at most \(2t\) keys. \(t\) is called the degree of \(B\). How many keys can this tree have on level \(h\)? How many children are there on level \(h\)? (CLRS 18.1-4) As a function of \(t\), what is the maximum number of keys that can be stored in a B-tree of height \(h\)? (CLRS 18.1-3) Show all legal B-trees of degree 2 that represent 1, 2, 3, 4, 5.