3.3. Assignment Heap (Paper assignment)

# 1. What are the minimum and maximum numbers of elements in a heap of height h?

- Minimum elements: elements

- Maximum elements: elements

# 2. Where in a max-heap might the smallest element reside, assuming that all elements are distinct?

In a max-heap, the smallest element will reside in one of the leaf nodes. In a max-heap, the parent nodes always have larger values than their children, and since all elements are distinct, the smallest element will be in one of the bottom-level leaf nodes.

# 3. Is an array that is in sorted order a min-heap?

No, an array that is in sorted order is not necessarily a min-heap or a max-heap. For an array to be a heap (either min-heap or max-heap), it needs to satisfy the heap property, which states that the value of each node must be greater than or equal to (in the case of a max-heap) or less than or equal to (in the case of a min-heap) the values of its children. While a sorted array can be converted into a heap using heapify operations, the array itself, in its sorted order, does not automatically fulfill the heap property.

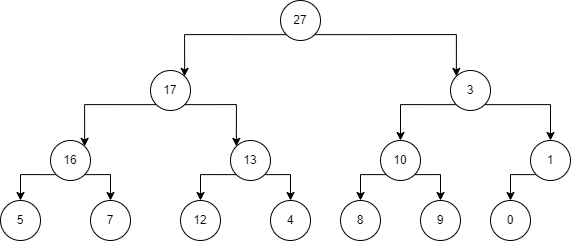
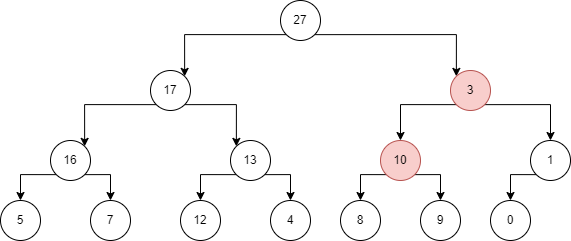
# 4. Is the array with values (23, 17, 14, 6, 13, 10, 1, 5, 7, 12) a max-heap?

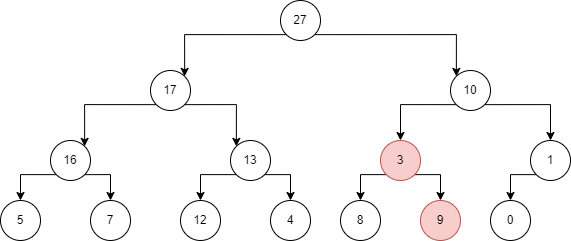
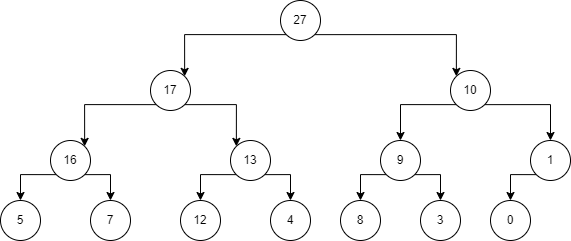
A diagram of a tree

Description automatically generatedThe above array can’t be a max-heap because there is a parent node that smaller than its child node if we construct a heap base on that array.

# 5. Draw a heap (27,17,3,16,13,10,1,5,7,12,4,8,9,0). Then, illustrate the operation of MAX-HEAPIFY (A,2) on that heap. (index = 2, 0-based)

The heap before the operation

After call MAX-HEAPIFY(A,2) then the heap will swap node 3 (index 2) with node 10 (index 5)

Then it continues to swap node 3 (index 5) with node 9 (index 12)

# What is the effect of calling MAX-HEAPIFY(A,i) when the element A[i] is larger than its children? Explain your answer.

There is no effect. All three if conditions fail, largest is set to i, and the process terminates wihtout having

changed anything in the heap.

# What is the effect of calling MAX-HEAPIFY(A,i) for i > A.heap-size/2? Explain your answer.

If then node has no children (and it is either at the second lowest or lowest level of the

binary tree). Moreover, Left(i) and Right(i) are larger than A.heap-size, meaning that going to its children will be error, because the array index will be out of range.

# Illustrate the operation of BUILD-MAX-HEAP on the array A = (5,3,17,10,84,19,6,22,9).

A diagram of a network

Description automatically generatedAfter creating the heap, then call MAX-HEAPIFY(A,4)

A diagram of a network

Description automatically generatedMAX-HEAPIFY(A,3)

A diagram of a network

Description automatically generatedMAX-HEAPIFY(A,2)

A diagram of a network

Description automatically generatedMAX-HEAPIFY(A,1)

A diagram of a network

Description automatically generatedFinal heap:

# Illustrate the operation of HEAPSORT on the array A = (5,13,2,25,7,17,20,8,4).

A diagram of a network

Description automatically generatedFirst we build a max-heap on this array

Then pop max node and place it into the last position of the array until the heap is empty

A diagram of a network

Description automatically generatedA diagram of a network

Description automatically generatedA diagram of a network

Description automatically generatedA diagram of a network

Description automatically generated

A group of circles with numbers

Description automatically generatedA group of circles with numbers

Description automatically generatedA diagram of numbers and circles

Description automatically generatedA diagram of numbers and circles

Description automatically generated

Then the resulting array is sorted:

# 10.What is the running time of HEAPSORT on an array A of length n that is already sorted in increasing order? What about decreasing order?

If A is sorted in increasing order, Build-Max-Heap will attain the maximum running time of , since it tries to order the array in a decreasing order. The n − 1 calls Max-Heapify(A, 1) will take at most time (there are no particular time saves from max-heapifying an ordered array), hence the running time of Heapsort will be .

If A is sorted in decreasing order, Max-Heapify(A, i) has running time for any i (since it never calls itself recursively, as largest = i for all i. However, this makes no difference in the running time of Build-Max-Heap, as we still get running time due to the calls to Max-Heapify. Here as well the n−1 calls Max-Heapify(A, 1) will take at most time (completely reversing the order of an array certainly does not save time). Hence the running time of Heapsort will be .

# 11.Implement heapSort(a,n).

**void** HEAPIFY(**int** a[], **int** N, **int** i)

{

**int** largest = i;

**int** l = 2 \* i + 1;

**int** r = 2 \* i + 2;

**if** (l < N && a[l] > a[largest])

        largest = l;

**if** (r < N && a[r] > a[largest])

        largest = r;

**if** (largest != i) {

        swap(a[i], a[largest]);

        HEAPIFY(a, N, largest);

    }

}

**void** heapSort(**int** a[], **int** N)

{

**for** (**int** i = N / 2 - 1; i >= 0; i--)

        HEAPIFY(a, N, i);

**for** (**int** i = N - 1; i > 0; i--) {

        swap(a[0], a[i]); // pop the current max node into the last position

        HEAPIFY(a, i, 0); // mofidy the heap

    }

}