

## Programming assignment 5.

**Due date:** Sunday, April 5 2020 at 11:59pm

---

### Part A

**NOTE:** You are **NOT** allowed to define a new array in any part of the code.

Create the following functions: **build\_MaxHeap**, **max\_heapify**, **heap\_sort**.

1. Request the user to enter a positive integer, and call it **n**.
2. Generate **n** random integers between **-100 to 100** and save them in array **a**.
3. Call **heap\_sort** function to sort the array. In order to sort the array using heapsort, you need to follow the below steps:
  - 3.1 Build a max-heap (call the function **build\_Maxheap**). In order to build the max-heap follow the below pseudocode:

*% new\_a is the output of the function, if you are using any other programming language, please write %  
return new\_a at the end of your code.*

**a = build\_MaxHeap(a)**

**n = length(a);**

**for i=n:-1:1**     *% i= n, n-1, n-2,..., 1, Can we start from n/2 instead of n? Why???*

**a(1:i) = max\_heapify(a,i);**     *% a(1:i) = contains a[1] a[2] ...a[i] (You could have a flag  
for the i too! (Assume that i is the index of the last element of the array! 😊)*

**end**

- 3.2 Keep removing the roots (first element in a) one by one until the tree/array becomes empty.

4. Determine the **average-running time** of **heap\_sort** function for **n=1000**, and **100** repetitions.
5. Compare your answer with the average-running time of selection sort (**you need implement it**).

### Part B

1. Generate and print a random array of size 10.
2. Call **heap\_sort** to sort the numbers.
3. Print the result.