Binary Heaps: Homework 2

- *By modifying the code written during the last lessons, provide an array-based implementation of binary heaps which avoids to swap the elements in the array* `A`.
  (***Hint**: use two arrays,* `key_pos` *and* `rev_pos`*, of natural numbers reporting the position of the key of a node and the node corresponding to a given position, respectively)*

  The solution can be found in the folder 04_Binary_heaps_noswap.

- *Consider the next algorithm:*

  ```
  def Ex2(A)
      D ← build(A)

      while ¬ is_empty(D)
          extract_min(D)
      end while
  end def
  ```

  where `A` *is an array. Compute the time-complexity of the algorithm when:*

  - `build`, `is_empty` $\in \Theta(1)$, `extract_min` $\in \Theta(|D|)$;
  - `build` $\in \Theta(|A|)$, `is_empty` $\in \Theta(1)$, `extract_min` $\in O(\log n)$;

  In the first case, the time complexity is $\Theta(1) + |D| \cdot \Theta(|D|) = \Theta(|D|^2)$, since `build` costs $\Theta(1)$ and the while is repeated until D is empty, so $|D|$ times, with inside `extract_min` that costs $\Theta(|D|)$.

  In the second case, $\Theta(|A|) + |D| \cdot O(\log n) = O(|A| + |D| \log n)$, since `build` costs $\Theta(|A|)$ and the while is repeated until D is empty, so $|D|$ times, with inside `extract_min` that costs $O(\log n)$.