Heaps: homework 2

Ex. 1

An array-based binary heap does not necessarily require swaps in the array `A` it is built on. In fact, it is convenient to fix the aforementioned array and perform swaps using auxiliary arrays of integers.

In this exercise, I used two arrays of natural numbers,`key_pos` and `rev_pos`:

- `key_pos[i]` stores the position of the `i`-th key in the array `A`
- `rev_pos[i]` stores the position of the `i`-th element of the array `A` in the structure of the heap

Doing so, costly swaps in memory can be avoided and replaced by quicker swaps of integer numbers.

Ex. 2

Considering the algorithm:

```
def Ex2(A)
    D ← build (A)
    while ¬ is_empty(D)
        extract_min(D)
    endwhile
enddef
```

If `build`, `is_empty` $\in \Theta(1)$ and `extract_min` $\in \Theta(|D|)$ and supposing that `extract_min` extracts the minimum from the heap `D`, the execution time of `Ex2` is:

$$T = \Theta(1) + \sum_{i=0}^{|D|}(\Theta(1) + \Theta(i)) \in \Theta(|D|^2)$$

If `build` $\in \Theta(|A|)$, `is_empty` $\in \Theta(1)$ and `extract_min` $\in O(log(|D|))$:

$$T = \Theta(|A|) + \sum_{i=0}^{|D|}(\Theta(1) + O(log(i))) = \Theta(|A|) + \Theta(|D|) + O(|D|log(|D|)) = \Theta(|A|) + O(|D|log(|D|))$$

It seems reasonable to assume that $|A| = |D|$. In this case, $T \in O(|D|log(|D|))$.