Vlad Bogdan-Tudor, 917
VladB

10)
```
function operation(n, a) is:
    while n>0 AND n mod 10 ≠ a execute
        n ← n-1
    end-while
    operation ← n
end-function
```

Best Case: $\Theta(1)$
Worst Case: $\Theta(n)$
Total Complexity: $O(n)$

```
subalgorithm main(n, x) is
    r ← 0
    i ← 1
    while i ≤ n execute
        j ← i
        if x[i] mod 10 < x[j] mod 10 then
            r ← r+1
        end-if                                   } Θ(1)
        while j ≤ n execute
            operation(n, x[j] mod 10)
            j ← j+1
        end-while
        i ← i+1
    end-while
end-subalgorithm
```

Vlad Bogdan-Tudor, 917
VlaB

The first while executes n times. $\Rightarrow \Theta(n)$

The second while executes $n, n-1, n-2, \ldots, 1$ times

$n + n-1 + n-2 + \ldots + 1 = \frac{n(n+1)}{2} \Rightarrow \Theta(n^2)$

$\Rightarrow$ The first + the second while's form a complexity of $\Theta(n^2)$

<operation> inside the 2nd while has Best/Worst cases

$\Rightarrow$
- Best Case: $\Theta(n^2)$
- Worst Case: $\Theta(n^3)$
- Total complexity: $O(n^3)$

$\leftarrow$ Complexity of the given piece of code