**Algorithms and Data Structures**
**Midterm Example**

**Start** - 1 point

**E1** (3 points) Let's consider the following algorithm:

(a) (2 points) Propose an invariant for the repetitive processing in the algorithm `ALG` described above, and use this invariant to demonstrate that the algorithm returns $\sum_{i=0}^{n} a[i] * x^i$.

```
1: function ALG(integer a[1..n], x)
2:     integer s, i
3:     s ← a[n]; i ← n
4:     while i > 0 do
5:         i ← i − 1; s ← s ∗ x + a[i]
6:     end while
7:     return s
8: end function
```

(b) (1 point) Analyze the efficiency of the algorithm `ALG` by following these steps: determine the problem size, choose the dominant operation, estimate the execution time, and establish the order of complexity.

**E2** (3 points)

(a) (2 points) Consider an array $x[0..n]$ with n integer elements. Describe in pseudocode or in Python an algorithm that returns the index of the first strictly positive value encountered while traversing the array from left to right. If such a value does not exist, return $-1$. For example, for $x = [-1, 0, 3, -1, 0, 1]$, the algorithm should return 2.

(b) (1 point) Analyze the efficiency of the algorithm proposed in part (a) by following these steps: determine the problem size, choose the dominant operation, estimate the execution time, and establish the order of complexity.

**E3** (3 points)

(a) (2 points) Describe (in pseudocode or in Python) an algorithm that, for a matrix $A[1..n, 1..n]$, returns $True$ if the matrix is lower triangular (all elements above the main diagonal are zero) and $False$ otherwise.

(b) (1 point) Analyze the efficiency of the algorithm proposed in part (a) by following these steps: determine the problem size, choose the dominant operation, estimate the execution time, and establish the order of complexity.