**Functions that often appear in algorithm analysis:**

---

Constant ≈ 1
Logarithmic ≈ log n
Linear ≈ n
N-Log-N ≈ n log n
Quadratic ≈ n^2
Cubic ≈ n^3
Exponential ≈ 2^n

**Algorithm Pseudo-code Example - Finding max element from an array**

---

**arrayMax(A,n):**

```
    Input: An array A storing n>=1 integers
    Output: Maximum element in A
    current_max ← A[0]
    for i ← to n-1 do
        if current_max < A[i] then
            current_max ← A[i]
    return current_max
```

**The Stack Abstract Data Type**

---

Stacks are the simplest of all data structures, yet they are also among the most important. They are used in a host of different applications, and as a tool for many more sophisticated data structures and algorithms.
Formally, a stack is an abstract data type (ADT) such that an instance S supports the following two methods:
**S.push(e):** Add element e to the top of stack S.
**S.pop( ):** Remove and return the top element from the stack S;
an error occurs if the stack is empty.
Additionally, let us define the following accessor methods for convenience:
**S.top( ):** Return a reference to the top element of stack S, without
removing it; an error occurs if the stack is empty.
**S.is empty( ):** Return True if stack S does not contain any elements.
**len(S):** Return the number of elements in stack S; in Python, we
implement this with the special method len .