**Space complexity**

**Document Id 10**

This is essentially the number of memory cells which an algorithm needs

=> means how much memory

**int r = x + y + z;** => O(1)

**for (int i = 0; i < n; ++i) {  
 r += a[i];  
 }**

=> O(n)

If an algorithm calls itself recursively N times then its space can't be reused because every

call is still in progress, so it needs O(N2) units

Big Theta (Θ)

Big-O is an upper bound.

Big-Theta is a tight bound, i.e. upper *and* lower bound.

Big-O(some expression) => time complexity is at most (worst case)

Big-O(some expression) => **less than or equal to** some expression

Big-Theta(some expression) => time complexity is proportional too some expression

Big-Theta(some expression) => time complexity is **equal to**  some expression