

Rosen, Discrete Mathematics and Its Applications, 6th edition  
Extra Examples

Section 3.3—Complexity of Algorithms



— Page references correspond to locations of Extra Examples icons in the textbook.

---

**p.193, icon at Example 1**

**#1.** Determine the complexity function that measures the number of print statements in an algorithm that takes a positive integer  $n$  and prints one 1, two 2's, three 3's, ...,  $n$   $n$ 's.

**Solution:**

For input  $n$  the number of print statements is equal to  $f(n) = 1 + 2 + 3 + \cdots + n$ .

To find a big- $O$  estimate for  $f(n)$ , note that

$$f(n) = 1 + 2 + 3 + \cdots + n \leq n + n + n + \cdots + n = n \cdot n = n^2,$$

which is  $O(n^2)$ .

This is the best possible big- $O$  function for  $f(n)$ . To see this, we will give a formula for the sum  $1 + 2 + 3 + \cdots + n$ :  $f(n) = 1 + 2 + 3 + \cdots + n = n(n+1)/2$ . To develop this formula, begin by writing the sum for  $f(n)$  both forward and backward:

$$f(n) = 1 + 2 + 3 + \cdots + (n-1) + n$$

$$f(n) = n + (n-1) + \cdots + 3 + 2 + 1.$$

Add the first term in each sum, the second terms in each sum, etc., to obtain

$$2f(n) = (n+1) + (n+1) + \cdots + (n+1)$$

where there are  $n$  terms in this sum. Therefore  $2f(n) = n(n+1)$  and hence  $f(n) = n(n+1)/2$ .

(The technique we just used to obtain this sum has been attributed to the mathematician Gauss when he was very young.) In Section 4.1 we will discuss another way to prove that this formula is correct. Therefore we have  $f(n) = n(n+1)/2$ , which is  $O(n^2)$ , but clearly not  $O(n)$ .

---

**p.193, icon at Example 1**

**#2.** Suppose an algorithm takes a sequence of  $n$  ( $\geq 2$ ) integers and determines if it contains an integer that is a repeat of the first integer in the list. Find the complexity function for the:

- (a) best case analysis.
- (b) worst case analysis.

**Solution:**

(a) The complexity function for the best case is  $f(n) = 1$ . Making the second integer equal to the first will force the algorithm to terminate after only one comparison.

(b) The complexity function for the worst case is  $f(n) = n$ . Having no repeat of the first integer will force the algorithm to terminate after making all  $n - 1$  comparisons.

---