**Exercise 1**

In each of the following situations indicate whether f=O(g), or f=Ω(g), or both (in which case f=Θ(g)).

|  |  |  |
| --- | --- | --- |
|  | f(n) | g(n) |
| 1 | n-100 | n-200 |
| Both are O(n), so f = Θ(g). | | |
| 2 | n1/2 | n2/3 |
| For powers of n, just compare the powers. We have that 1/2 < 2/3, so f = O(g). | | |
| 3 | 100n+logn | n+(logn)2 |
| Both are O(n), so f = Θ(g). | | |
| 4 | nlogn | 10nlog10n |
| Both are O(n log n), so f = Θ(g). | | |
| 5 | log2n | log3n |
| Both are O(log n), so f = Θ(g). | | |
| 6 | 10logn | log(n2) |
| Both are O(log n), so f = Θ(g). | | |
| 7 | n1.01 | nlog2n |
| If we divide both sides by n, we need to compare n 0.01 and log2 n. It takes a really long time, but ultimately the power function wins out, so f = Ω(g). You can play with www.wolframalpha.com to see this. | | |
| 8 | n2/logn | n(logn)2 |
| If we divide both sides by n/ log n, we need to compare n and (log n) 3 . The result is f = Ω(g). | | |
| 9 | n0.1 | (logn)10 |
| Once again, f = Ω(g). | | |
| 10 | (logn)logn | n/logn |
| The function f(n) = n log log n , so f = Ω(g). | | |
| 11 | n2n | 3n |
| Here, f = O(g). | | |
| 12 | 2n | 2n+1 |
| Here, f = Θ(g). | | |
| 13 | n! | 2n |
| Lookup “factorial” in www.wikipedia.org, and you will find that n! > √ 2πn( n e ) n . Hence f = Ω(g). | | |

**Exercise 2**

Show that, if ***c*** is a positive real number, then g(n)=1+c+c2+…+cn is:

a) Θ(1) if c < 1.

b) Θ(n) if c = 1.

c) Θ(cn) if c > 1.

The moral: in big-Θ terms, the sum of a geometric series is simply the first term if the series is strictly decreasing, the last term if the series is strictly increasing, or the number of terms if the series is unchanging.



**Exercise 3**

Implement procedures of Linear-Search, Better-Linear-Search, Sentinel-Linear-Search and Recursive-Linear-Search using any programming language and find how long each procedure is performed.







