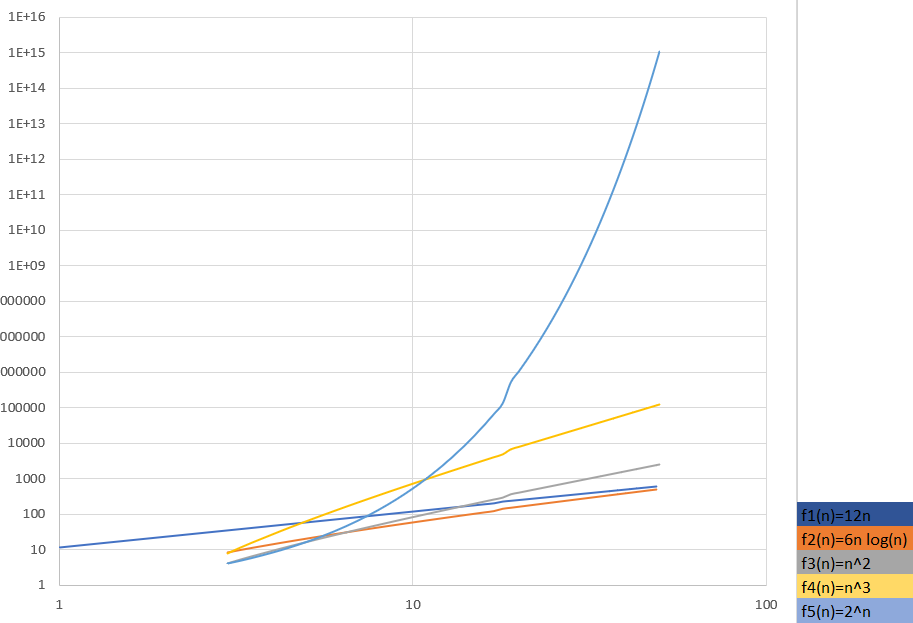
Assignment 1

R-1.1 Graph the functions 12*n*, 6*n* log *n*, *n*2, *n*3, and 2n using logarithmic scale for the x- and y-axes; that is, if the function value *f(n)* is *y*, plot this as a point with *x*-coordinate at log *n* and *y*-coordinate at log *y*.



R-1.2 Algorithm A uses 10*n* log *n* operations, while algorithm B uses *n*2 operations. Determine the value *n*0 such that A is better than B for *n* ≥ *n*0.

10nlogn<=n2 -----------------------🡪 10logn<=n --------------🡪 n0>=10

R-1.6 Order the following list of functions by the big-O notation.

|  |  |  |  |
| --- | --- | --- | --- |
| *n* log *n* | log log *n* | 1/*n* | 4*n*3/2 |
| 5n | 2*n* log2 *n* | 2n | 4n |
| N3 | N2log n | 4log n | √*n* |

|  |  |
| --- | --- |
| 1. 1/n 2. Log log n 3. √*n* 4. 4log n 5. 5n 6. *n* log *n* | 1. 2*n* log2 *n* 2. 4*n*3/2 3. N2log n 4. N3 5. 2n 6. 4n |

R-1.10 Give a big-O characterization, in terms of *n*, of the running time of the Loop1 method below:

|  |  |
| --- | --- |
| **Algorithm** Loop1(n) |  |
| s ← 0 | O(1) |
| **for** *i* ← 1 **to** *n* **do** | O(n) |
| *s* ← *s* + *i* | O(n) |
| *Tottal* | O(1)+2O(n) =O(n) |

R-1.14 Perform a similar analysis for method Loop5 below:

|  |  |
| --- | --- |
| **Algorithm** Loop5(n) |  |
| s ← 0 | O(1) |
| **for** *i* ← 1 **to** *n*2 **do** | O(n2) |
| **for** *j* ← 1 **to** i **do** | O(n4) |
| *s* ← *s* + *i* | O(n4) |
| *Tottal* | O(1)+O(n2) +2O(n4) =O(n4) |