DESIGN AND ANALYSIS OF ALGORITHMS
LAB 01: Complexity notation, Bruce-force Algorithms

I. Complexity notation

$T(n), f(n)$ - monotonically increasing functions, nonnegative, n is the input size of algorithms.

1. Big-O notation
   a. Definition: $T(n) \in O(f(n)) \Leftrightarrow \exists c > 0, \exists n_0 \geq 0: T(n) \leq c.f(n), \forall n \geq n_0$
   b. For example,
      if $T(n) = 32n^2 + 17n + 32$
      $f(n) = n^2$
      , i.e.
      $$T(n) \in O(f(n))$$
      $$32n^2 + 17n + 32 \in O(n^2)$$
      Because.
      $$32n^2 + 17n + 32 \leq 81 * n^2, \forall n \geq 2 \ (c = 81, n_0 = 2)$$
   c. Big-O is upper bounds on computational complexity of algorithms.
2. Big-Omega notation
   a. Definition:$T(n) \in \Omega(f(n)) \Leftrightarrow \exists c > 0, \exists n_0 \geq 0: T(n) \geq c.f(n), \forall n \geq n_0$
   b. For example, if $T(n) = 32n^2 + 17n + 32$
      $f(n) = n^2$
      , i.e.$T(n) \in \Omega(f(n))$
      $$32n^2 + 17n + 32 \in \Omega(n^2)$$
      Because.$32n^2 + 17n + 32 \geq 1 * n^2, \forall n \geq 2 \ (c = 1, n_0 = 1)$
   c. Big-Omega Is the lower bounds on computational complexity of the algorithm.
3. Big-Theta Notation:
   a. Definition:$T(n) \in O(f(n)) \Leftrightarrow \exists c_1 > 0, \exists c_2 > 0, \exists n_0 \geq 0:$
      $$c_1 * f(n) \leq T(n) \leq c_2.f(n), \forall n \geq n_0$$
   b. For example, if $T(n) = 32n^2 + 17n + 32$
      $f(n) = n^2$

, i.e.

$$T(n) \in \Theta(f(n))$$

$$32n^2 + 17n + 32 \in \Theta(n^2)$$

Because $32n^2 + 17n + 32 \le 81 * n^2, \forall n \ge 2\ (c = 81, n_0 = 2)$

(We have:)

$$T(n) \in O(f(n))$$

And

$$32n^2 + 17n + 32 \ge 1 * n^2, \forall n \ge 2\ (c_1 = 1, n_0 = 1)$$

(We have:)$T(n) \in \Omega(f(n))$

c. Big-Theta: Is the tight bound on computational complexity of algorithms.

II. Sample exercises with solutions on complexity notations

1. Sample

The ascending order of speed of functions (as n increases) is given in the following table from left to right:

c	$O(lgn)$	$O(n)$	$O(nlgn)$	$O(n^2)$	$O(n^3)$	$O(a^n)$ $(a > 1)$	$O(n!)$

Let's sort the following functions in ascending order of the speed:

$f_1(n) = 10^n$
$f_2(n) = n^{1/3}$
$f_3(n) = n^n$
$f_4(n) = \log_2 n$
$f_5(n) = 2^{\sqrt{\log_2 n}}$

Solution:

Notice, the functions f_1, f_2, f_4 are the basic functions that have been provided in the given table. Therefore, we have:

$$f_4(n) = \log_2 n << f_2(n) = n^{1/3} << f_1(n) = 10^n \text{ (*)}$$

a. Find the relationship between $\boldsymbol{f_1(n) = 10^n}$ **and** $\boldsymbol{f_3(n) = n^n}$

We have:$10^n \le 1 * n^n, \forall n \ge 10 = n_0 (c = 1)$

According to the Big-O definition, we have:$10^n \in O(n^n)$

So: $f_1(n) = 10^n \ll f_3(n) = n^n$ (**)

b. Find the relationship between

$\boldsymbol{f_2(n) = n^{\frac{1}{3}}}$, and $\boldsymbol{f_4(n) = \log_2 n}$ and $\boldsymbol{f_5(n) = 2^{\sqrt{\log_2 n}}}$

Taking logarithm with the base of two of these three functions, we are:

$$\log_2 f_2(n) = \frac{1}{3}\log_2 n$$

$$\log_2 f_4(n) = \log_2 \log_2 n$$

$$\log_2 f_5 = \log_2 2^{\sqrt{\log_2 n}} = \sqrt{\log_2 n} = (\log_2 n)^{1/2}$$

Set $z = log_2 n$, the above expressions become the following basic functions:

$$\log_2 f_2(n) = \frac{1}{3}\log_2 n = \frac{1}{3}z$$

$$\log_2 f_4(n) = \log_2 \log_2 n = \log_2 z$$

$$\log_2 f_5(n) = z^{1/2}$$

We have:

$\log_2 z \le 1 * z^{1/2}, \forall z \ge 16$(i.e. or)

$$\log_2 16 \le \ 16 \le 2^n = 65536$$

According to the big-O definition:

$\log_2 z \in O\left(z^{1/2}\right)$

$\Leftrightarrow \log_2 f_4(n) \in O(\log_2 f_5\,(n)$

$\Leftrightarrow f_4(n) \in O(f_5(n))$

Similarly, we have:

$z^{1/2} \le 1 * \frac{1}{3}z, \forall z \ge 9$(i.e. or)$\log_2 n \ge 9n \ge 2^9 = 512$

According to the big-O definition:

$$z^{1/2} \in O(\frac{1}{3} * z)$$

$\Leftrightarrow \log_2 f_5(n) \in O(\log_2 f_2\,(n))$

$\Leftrightarrow f_5(n) \in O(f_2(n))$

So: $f_4(n) \ll f_5(n) \ll f_2(n)$ (* * *)

From (*), (* *), (* *) we have the arrangement:

$$f_4(n) = \log_2 n \ll f_5(n) = 2^{\sqrt{\log_2 n}} \ll f_2(n) = n^{\frac{1}{3}} \ll$$

$$f_1(n) = 10^n \ll f_1(n) = 10^n \ll f_3(n) = n^n$$

2. Exercises

1. The following conclusions are true or FALSE, if correct, prove it, if wrong, give a counter-example.
   a. $32n^2 + 17n + 32 \in O(n)$
   $$\lim_{n\to\infty} \frac{32n^2 + 17n + 32}{n} = \lim_{n\to\infty} 32n = \infty$$
   b. $32n^2 + 17n + 32 \in O(n^3)$
   c. $32n^2 + 17n + 32 \in \Omega(n^3)$
   d. $32n^2 + 17n + 32 \in \Omega(n)$
   e. $2^{n+1} \in \mathrm{O}(2^n)$?
   f. $2^{2n} = 2^{\mathrm{n}}2^{\mathrm{n}} \in \mathrm{O}(2^n)$?
   g. If $f(n) \in O(g(n))$ and $g(n) \in O(f(n))$ then$f(n) = g(n)$
   $$f(n) \in O\big(g(n)\big) \equiv \exists c, n_0\ f(n) \le cg(n) \forall n \ge n_0$$
   $$g(n) \in O\big(f(n)\big) \equiv \exists c_1, n_1\ g(n) \le c_1 f(n) \forall n \ge n_1$$
   $$f(n) \ne g(n)$$
   $$counterexample:$$
   $$f(n) = 3n, g(n) = n$$
   $$f(n) \in O\big(g(n)\big)$$
   $$g(n) \in O\big(f(n)\big)$$
   $$f(n) = 3n \ne g(n) = n$$

2. Given two functions $f(n), g(n)$: $f(n) \in O(g(n))$. Let's tell if the following conclusions are right or wrong, if true, prove them, if wrong, give examples:
   a. $\log_2 f(n) \in O(\log_2 g(n))$
   b. $2^{f(n)} \in O\big(2^{g(n)}\big)$
   c. $f(n)^2 \in O(g(n)^2)$
3. Prove the following conclusions:
   A. $f(n) \in \mathrm{O}\big(g(n)\big) \Leftrightarrow \mathrm{g}(n) \in \Omega\big(f(n)\big)$
   B. $\forall \mathrm{a}, \mathrm{b} > 0$, $(\mathrm{n} + \mathrm{a})^{\mathrm{b}} \in \Theta\big(\mathrm{n}^{\mathrm{b}}\big)$
   C. $f(n) = \Theta(g(n)) \Leftrightarrow \begin{cases} f(n) \in O(g(n)) \\ f(n) \in \Omega(g(n)) \end{cases}$
   D. $\left.\begin{matrix} f_1(n) \in O(g_1(n)) \\ f_2(n) \in O(g_2(n)) \end{matrix}\right\} \to f_1(n) + f_2(n) \in O(max\{g_1(n), g_2(n)\})$
4. Arrange the following functions in ascending order of speed:
   $f_1(n) = n^{2.5}$ $\qquad$ $f_2(n) = \sqrt{2n}$

$f_3(n) = n + 10$ $f_5(n) = 100^n$

$f_4(n) = 10^n$ $f_6(n) = n^2 \log n$

5. Arranges the following functions in ascending order of speed:

$g(n) = 2^{\sqrt{\log n}}$ $g_4(n) = n(\log n)^3$ $g_7(n) = 2^{n^2}$

$g_2(n) = 2^n$ $g_5(n) = n^{\log n}$

$g_3(n) = n^{4/3}$ $g_6(n) = 2^{2^n}$