# Unit 10 Mathematical Functions and Notations

**Structure:**



## 10.1 Introduction

You know that number of mathematical and statistical tools, techniques and notations form an essential part in the concept of analysis of algorithms. In this unit, we will study a number of well known approximation functions. These functions will calculate approximate values of quantities, and these quantities are useful in many situations and among these some of the quantities are calculated just for comparison with each other.

**Objectives:**

After studying this unit, you should be able to:

- use various Mathematical Notations like $\Sigma$, $\Pi$, $\lfloor\ \rfloor$, $\lceil\ \rceil$, mod, log, *e* etc.
- apply concepts of Mathematical Expectation in Algorithms
- use of principle of Mathematical Induction in establishing truth of infinitely many Statements
- analyze the efficiency of an Algorithm

## 10.2 Functions & Notations

To start with, we recall the following notations and definitions.

Here,

$N = \{1, 2, 3, \ldots.\}$

$I = \{\ldots, -2, -1, 0, 1, 2, \ldots..\}$

$R$ = set of Real numbers.

**Notation:** If $a_1, a_2, \ldots a_n$ are $n$ real variables/ numbers then

(i) Summation

The expression $a_1 + a_2 + \ldots + a_i + \ldots + a_n$ may be denoted in shorthand as $\sum_{i=1}^{n} a_i$

(ii) Product

The expression $a_1 \times a_2 \times \ldots x\, a_i \times \ldots \times a_n$ is denoted as $\prod_{i=1}^{n} a_i$

**Function**

For two given sets $A$ and $B$ (which need not be different , i.e., $A$ may be the same as $B$) a rule $f$ which associates with each element of $A$, a unique element of $B$, is called a function from $A$ to $B$. If $f$ is a function from a set $A$ to a set $B$, then we denote it by $f: A \rightarrow B$. Also, for $x \in A$, $f(x)$ is called image of $x$ in $B$. Then, $A$ is called the domain of $f$ and $B$ is called the co domain of $f$.

***Example:***

Let $f: I \rightarrow I$ be defined such that $f(x) = x^2$ for all $x \in I$ then

| | | | |
|---|---|---|---|
| $f$ maps | $4$ | to | $16$ |
| $f$ maps | $0$ | to | $0$ |
| $f$ maps | $5$ | to | $25$ |

**Remark**

We may note the following:

i) If $f: x \rightarrow y$ is a function, then there may be more than one element, say $x_1$ and $x_2$ such that $f(x_1) = f(x_2)$

In the above example

$f(2) = f(-2) = 4$

By putting restriction that $f(x) \neq f(y)$ if $x \neq y$, we get special functions, called $1 - 1$ or injective functions.

ii) Though for each element $x \in X$, there must be at least one element $y \in Y$, such that $f(x) = y$. However it is not necessary that for each element $y \in Y$ there must be at least one element $x \in X$, such that $f(x) = y$, For example, for $y = -3 \in Y$ there is no $x \in X$ such that $f(x) = x^2 = -3$. By putting the restriction on the function $f$, such that for each $y \in Y$, there must be at least one element $x$ of $X$ such that $f(x) = y$, we get special functions called onto or surjective functions.

**Definition of some functions:**

***1 – 1* or Injective Function:** A function $f: A \to B$ is said to be $1 - 1$ or injective function if for $x, y \in A$, if $f(x) = f(y)$ then $x = y$

We have already seen that the function defined in above example is not $1 - 1$. However, by changing the domain, though defined by the same rule, $f$ becomes a $1 - 1$ function.

***Example:*** In this particular case, if we change the domain to
$N = \{1, 2, 3 \ldots..\}$, then we can easily check that function.

$f: N \to N$ defined as $f(x) = x^2$ for all $x \in N$ is $1 - 1$ because, in this case, for each $x \in N$ its negative $-x \notin N$. Hence for $f(x) = f(y)$ implies $x = y$. For example, If $f(x) = 4$ then there is only one value of $x = 2$ such that $f(2) = 4$.

**Onto/surjective function:** A function $f: X \to Y$ is said to be onto, or subjective if to every element of, the co domain of $f$, there is an element $x \in X$ such that $f(x) = y$.

We have already seen that the function defined in above example is not onto.

However, by changing the co domain y or changing the rule, (or both) we can make f as onto.

***Example:*** (Changing the domain)
Let $X = I = \{\ldots.. -3, -2, -1, 0, 1, 2, 3 \ldots\ldots\}$ but, if we change $Y$ as

$$Y = \{0, 1, 4, 9 \ldots..\} = \{y \mid y = n^2 \text{ for } n \in X\}$$

Then it can be seen that f: $X \to Y$ defined by $f(x) = x^2$ for all $x \in X$ is onto

***Example:*** (changing the rule)
Here, we change the rule so that $X = Y = \{ \ldots\ldots.-3, -2, -1, 0, 1, 2, 3 \ldots\ldots.\}$

But $f: X \to Y$ is defined as $f(x) = x + 3$ for $x \in X$.

Then we apply the definition to show that $f$ is onto

If $y \in Y$, then, by definition, for $f$ to be onto, there must exist an $x \in X$ such that $f(x) = y$. So the problem is to find out $x \in X$ such that $f(x) = y$ .
Let us assume that $x \in X$ exists such that $f(x) = y$
i.e., $x + 3 = y$
i.e., $x = y - 3$

But, as $y$ is given, $x$ is known through the above equation. Hence $f$ is onto.

**Monotonic Functions:** For the definition of monotonic functions, we consider only functions

$f: R \rightarrow R$ where, $R$ is the set of real numbers.

A function $f: R \rightarrow R$ is said to be monotonically increasing if for $x, y \in R$ and $x \leq y$ we have $f(x) \leq f(y)$

Further, $f$ is said to be strictly monotonically increasing if $x < y$ then $f(x) < f(y)$

***Example:***

Let f: $R \rightarrow R$ be defined as $f(x) = x + 3$, for $x \in R$

Then, for $x_1, x_2 \in R$, the domain, if $x_1 \geq x_2$ then $x_1 + 3 \geq x_2 + 3$, (by using monotone property of addition), which implies $f(x_1) \geq f(x_2)$. Hence, $f$ is monotonically increasing.

A function f: $R \rightarrow R$ is said to be monotonically decreasing if, for $x, y \in R$ and $x \leq y$ then

$f(x) \geq f(y)$

In other words, as $x$ increases, value of its image decreases.

Further, $f$ is said to be strictly monotonically decreasing, if $x < y$ then $f(x) > f(y)$.

***Example:***

Let $f: R \rightarrow R$ be defined as

$f(x) = -x + 3$

If $x_1 \geq x_2$ then $-x_1 \leq -x_2$ implying $-x_1 + 3 \leq -x_2 + 3$ which further implies that $f(x_1) \leq f(x_2)$ hence, $f$ is monotonically decreasing.

**Self Assessment Questions**

1. A function $f: R \rightarrow R$ is said to be monotonically increasing if for $x, y \in R$ and $x \leq y$ we have ————————.
2. ———————— maps each real number $x$ to the integer, which is the least of all integers greater than or equal to $x$.

## 10.3 Modular Arithmetic/ Mod Function

Consider the following

If, we are following 12 hour clock, and if it is 11 O'clock now then after 3 hours, it will be 2 O'clock and not 14 O'clock (whenever the number of o'clock exceeds 12, we subtract n = 12 from the number)

**Definitions**

***b* mod *n*:** If $n$ is a given positive integer and $b$ is any integer, then
$b$ mod $n = r$ where $0 \leq r < n$ and $b = k * n + r$

In other words, $r$ is obtained by subtracting multiplies of $n$ so that the remainder $r$ lies between $0$ and $(n - 1)$.

For example: If $b = 42$ and $n = 11$ then

$b$ mod $n = -42$ mod $11 = 2$ $(\because -42 = (-4) \times 11 + 2)$

Mod function can also expressed in terms of the floor function as follows:

$b(mod\ n) = b - \left\lfloor \frac{b}{n} \right\rfloor \times n$

**Factorial:** For $N = \{1, 2, 3, \ldots\}$, the factorial function

factorial: $N \cup \{0\} \rightarrow N \cup \{0\}$

given by $\angle n = n.\ \angle n - 1 = n \times$ factorial $(n - 1)$

**Exponentiation Function (Exp):** is a function of two variables $x$ and $n$ where $x$ is any non – negative real number and n is an integer (though $n$ can be taken as non integer also, but we restrict to integers only)

Exp $(x\ n)$ denoted by $x^n$, is defined recursively as follows:
For $n = 0$
Exp $(x, 0) = x^0 = 1$
For $n > 0$
Exp $(x, n) = x.\ Exp\ (x, n - 1)$
i.e.,
$x^n = x\ .\ x^{n-1}$
For $n < 0$, let $n = -m$ where $m > 0$

$$x^n = x^{-m} = \frac{1}{x^m}$$

In $x^n$, $n$ is also called the exponent/ power of $x$.

For example: If $x = 1.5$, $n = 3$, then

Also, Exp $(1.5, 3) = (1.5)^3 = (1.5) \times [(1.5)^2] = [(1.5) [1.5 \times (1.5)^1]$

$= 1.5 [(1.5 \times (1.5 \times (1.5)^0))]$

$= 1.5 [(1.5 \times (1.5 \times 1))] = 1.5 [(1.5 \times 1.5)]$

$1.5 [2.25] = 3.375$

Exp $(1.5, -3) = (1.5)^{-3} = \frac{1}{(1.5)^3} = \frac{1}{3.375}$

Further, the following rules apply to exponential function.

$$\left((b)^m\right)^n = b^{nm}$$

$$\left(b^m\right)^n = \left(b^n\right)^m$$

$$b^m . b^n = b^{m+n}$$

For $b \geq 1$ and for all $n$, the function $b^n$ is monotonically increasing in $n$. In other words, if $n_1 \geq n_2$ then $b^{n1} \geq b^{n2}$ $if\ b \geq 1.$

**Polynomial:** A polynomial in $n$ of degree $k$, where $k$ is a non – negative integer, over $R$, the set of real numbers, denoted by $P(n)$, is of the form

$$P_k(n) = a_k n^k + a_{k-1} n^{k-1} + ...... + a_1\ n + a_0$$

Where, $a_k \neq 0$ and $a_i \in R,$ i = $0, 1, ……..,K$

Using the summation notation

$$P_k(n) = \sum_{i=0}^{k} a_i\ n^i \qquad a_k \neq 0, a_i \in R$$

Each of $\left(a_i\ n^i\right)$ is called a term.

Generally, the suffix $k$ in $P_k$ $(n)$ is dropped and instead of $P_k(n)$ we write $P(n)$ only.

We may note that $P(n) = n^k = 1.\ n^k$ for any $k$, is a single term polynomial. If $k \geq 0$ then $P(n) = n^k$ is monotonically increasing. Further, if $k \leq 0$ then $p(n) = n^k$ is monotonically decreasing.

***Result:*** Though $0^\circ = 1$. The following is a very useful result relating the exponentials and polynomials.

For any constants $b$ and $c$ with $b > 1$

$$\lim_{n\to\infty} \frac{n^c}{b^n} = 0$$

The result, in non – mathematical terms, states that for any given constants $b$ and $c$, but with

$b > 1$, the terms in sequence $\frac{1^c}{b^1}, \frac{2^c}{b^2}, \frac{3^c}{b^3}, \ldots\ldots, \frac{k^c}{b^k}, \ldots\ldots\ldots$

gradually decrease and approaches zero. Which further means that for constants $b$ and $c$, and integer variable $n$, the exponential term $b^n$, for $b > 1$, increases at a much faster rate than the polynomial term $n^c$.

**Exponential Function:** The letter $e$ is used to denote the quantity

$$1 + \frac{1}{1!} + \frac{1}{2!} + \frac{1}{3!} + \ldots\ldots$$

and is taken as the base of natural logarithm function, then for all real numbers $x$, we define the exponential function

$$e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \ldots\ldots\ldots = \sum_{i=0}^{\infty} \frac{x^i}{i!}$$

For all real numbers, we have

Further, if $|\,x\,| \leq 1$ then $1 + x \leq e^x \leq 1 + x + x^2$

The following is another useful result, which we state without proof:

***Result:*** $\lim_{n\to\infty}\left(1 + \frac{x}{n}\right)^n = e^x$

**Logarithm:** The concept of logarithm is defined indirectly by the definition of exponential defined earlier. If $a > 0$, $b > 0$ and $c > 0$ are three real numbers, such that

$$c = a^b$$

Then $b = \log_a c$ (read as log of $c$ to the base $a$)

Then *a* is called the base of the logarithm.

For example: if $2^6 = 64$, then $log_2\ 64 = 6$,
i.e., *2* raised to power *6* gives *64*.

Generally, two bases, viz, *2* and *3* are very common in scientific and computing fields and hence, the following special notations for these bases are used.

(i) *lg n* denotes $log_2 n$ *n* (base *2*)
(ii) *ln n* denoted $log_e n$ (base *e*);

Where the letter $\ell n$ denotes logarithm and the letter *n* denotes natural.

***Result:*** Given below are some important properties of logarithms without proof.

For *n*, a natural number and real numbers *a*, *b* and *c* all greater than *0*, the following identities are true

i) $log_a(bc) = log_a b + log_a c$

ii) $log_a(b^n) = n\ log_a b$

iii) $log_a\left(\frac{1}{b}\right) = -log_a b$

iv) $log_a b = \frac{1}{log_b a}$

v) $a^{log_b c} = c^{log_b a}$

**Self Assessment Questions**

3. ______________ is a function of two variables *x* and *n* where *x* is any non – negative real number and n is an integer.

4. $---- = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + ......... = \sum_{i=0}^{\infty} \frac{x^i}{i!}$

## 10.4 Mathematical Expectation in Average Case Analysis

The concept of mathematical expectation is needed in average – case analysis of algorithms. In order to understand the concept better, consider the following example.

**Example:** Suppose, the students of MCA, who completed all the courses in the year 2005, had the following distribution of marks:

| Range of marks | Percentage of students who scored in the range |
|---|---|
| 0% to 20% | 08 |
| 20% to 40% | 20 |
| 40% to 60% | 57 |
| 60% to 80% | 09 |
| 80% to 100% | 06 |

If a student is picked up randomly from the set of students under consideration, what is the % of marks expected of such a student? After scanning the table given above, we intuitively expect the student to score around the 40% to 60% class, because, more than half of the students have scored marks in and around this class.

Assuming that marks within a class are uniformly scored by the students in the class, the above table may be approximated by the following more compact table.

| % marks | Percentage of students scoring the marks |
|---|---|
| 10* | 08 |
| 30 | 20 |
| 50 | 57 |
| 70 | 09 |
| 90 | 06 |

As explained earlier, we expect a student picked up randomly, to score around 50% because more than half of the students have scored marks around 50%

This informal idea of expectation may be formalized by giving to each percentage of marks, weight in proportion to the number of students scoring the particular percentage of marks in the above table.

Thus, we assign weight $\left(\frac{8}{100}\right)$ to the score 10% ($\because$ 8, out of 100 students score on the average 10% marks); $\left(\frac{20}{100}\right)$ to the score 30% and so on.

Thus,

Expected % of marks

$$= 10 \times \frac{8}{100} + 30 \times \frac{20}{100} + 50 \times \frac{57}{100} + 70 \times \frac{9}{100} + 90 \times \frac{6}{100} = 47$$

The final calculation of expected marks of 47 is roughly equal to our intuition of the expected marks, according to our intuition, to be around 50.

We generalize and formalize these ideas in the form of the following definition.

**Mathematical Expectation**

For a given set *S* of items, let to each item, one of the *n* values, say $v_1, v_2 \ldots.., v_n$, be associated. Let the probability of occurrence of an item with value $v_i$ be $p_i$. If an item is picked up at random, then its expected value *E(v)* is given by

$$E(v) = \sum_{i=1}^{n} p_i v_i = p_1 v_1 + p_2 v_2 + \ldots.. + p_n v_n$$

**Self Assessment Question**

5. The concept of ———— is needed in average – case analysis of algorithms

## 10.5 Efficiency of an Algorithm

If a problem is algorithmically solvable then it may have more than one algorithmic solution. Mainly, the two computer resources taken into consideration for efficiency measures are time and space requirements for executing the program corresponding to the solution/algorithm. We will restrict to only time complexities of algorithms of the problems.

It is easy to realize that given an algorithm for multiplying two *n x n* matrices, the time required by the algorithm for finding the product of two *2 x 2* matrices, is expected to take much less time than the time taken by the same algorithm for multiplying, say, two *100 x 100* matrices. This explains intuitively the notion of the size and instances of a problem and also the role of size in determining the (time) complexity of algorithm. If the size of general instance is *n* then time complexity of the algorithm solving the problem under consideration is some function of *n*.

However, for all types of problems, this does not serve properly the purpose for which the notion of size is taken into consideration. Hence, different measures of size of an instance of a problem are used for different types of problem. For example,

i) In sorting and searching problems, the number of elements, which are to be sorted or are considered for searching, is taken as the size of the instance of the problem of sorting/ searching.

ii) In the case of solving polynomial equations i.e., dealing with the algebra of polynomials, the degrees of polynomial instances, may be taken as the sizes of the corresponding instances.

There are two approaches for determining complexity (or time required) for executing an algorithm, viz.,

i) empirical (or a posterior)
ii) theoretical (or priori)

In the empirical approach (the programmed) algorithm is actually executed on various instances of the problem and the size *(s)* and time *(t)* of execution for each instance is noted. And then by some numerical or other technique, *t* is determined as a function of *s*. This function then, is taken as complexity of the algorithm under consideration.

In the theoretical approach, we mathematically determine the time needed by the algorithm, for a general instance of size, say, *n* of the problem under consideration. In this approach, generally, each of the basic instructions like assignment, read and write and each of the basic operation like *'+',* comparison of pair of integers etc. is assumed to take one or more, but some constant number of, (basic) units of time for execution. Time for execution for each structuring rule is assumed to be some function of the time required for constituent of the structure.

Thus starting from the basic instructions and operations and using structuring rules, we can calculate the time complexity of a problem or an algorithm.

The theoretical approach has a number of advantages over the empirical approach as listed below:

i) The approach does not depend on the programming language in which the algorithm is coded and on how it is coded in the language.
ii) The approach does not depend on the computer system used for executing (a programmed version of) the algorithm.
iii) In case of a comparatively inefficient algorithm, which ultimately is to be rejected, the computer resources and programming efforts which otherwise would have been required and wasted, will be saved.
iv) Instead of applying the algorithm to many different – sized instances, the approach can be applied for a general size, say, *n* of an arbitrary instance of the problem under consideration. In the case of theoretical approach, the size *n* may be arbitrary large. However, in empirical approach, because of practical considerations, only the instances of moderate sizes may be considered.

**Self Assessment Question**

6. Starting from the basic instructions and operations and using structuring rules, we can calculate the ————— of a problem.

## 10.6 Well known Asymptotic Function & Notations

The purpose of these asymptotic growth rate functions to be introduced to facilitate the recognition of essential character of a complexity function through some simpler functions delivered by these notations. For example a complexity function $f(n) = 5004\ n^3 + 83\ n^2 + 19\ n + 408$, has essentially same behavior as that of $g(n) = n^3$ as the problem size *n* becomes larger and larger. But $g(n) = n^3$ is much more understtandable and its value easier to compute than the function *f(n)*

Given below are the five well-known approximation functions.

1) $O : (O\ (n^2)$ is pronounced as big-oh of $n^2$, or sometimes just as oh of $n^2$)
2) $\Omega : (\Omega\ (n^2)$ is pronounced as ‘big-omega of $n^2$ or sometimes just as omega of $n^2$)
3) $\Theta : (\Theta\ (n^2)$ is pronounced as ‘theta of $n^2$)

4) $o$ : ($o$ ($n^2$) is pronounced as 'little-oh $h^2$)
5) $\omega$ : ($\omega$ ($n^2$) is pronounced as 'little-omega of $n^2$)

These approximations denote relations from functions to functions.
$f, g: N \rightarrow N$ are given by
$f(n) = n^2 - 5n$ and $g(n) = n^2$ then $O(f(n)) = g(n)$ or $O(n^2 - 5n) = n^2$

To be more precise, each of these notations is a mapping that associates a set of functions to each function under consideration. For example, if $f(n)$ is a polynomial of degree $k$ then the set $O(f(n))$ includes all polynomials of degree less than or equal to $k$.

**Remarks**
In the discussion of any one of the five notations, generally two functions, say, $f$ and $g$ are involved. The functions have their domains and co domains as $N$, the set of natural numbers, i.e., $f : N \rightarrow N$

$g : N \rightarrow N$

These functions may also be considered as having domain and co domain as $R$.

**1) The notation *O***
Provides asymptotic upper bound for a given function. Let $f(x)$ and $g(x)$ be two functions each from the set of natural numbers or set of positive real numbers.

Then $f(x)$ is said to be $O(g(x))$ (pronounced as big-oh of $g$ of $x$) if there exists two positive integers/real number constants $C$ and $K$ such that

$$f(x) \leq C\, g(x) \quad \text{for all } x \geq k \quad \text{……….(A)}$$

**Note:** (The restriction of being positive on integers/ real is justified as all complexities are positive numbers)

***Example:*** For the function defined by
$f(x) = 2x^3 + 3x^2 + 1$ show that
i) $F(x) = O(x^3)$
ii) $f(x) = O(x^4)$
iii) $x^3 = O(f(x))$
iv) $x^4 \neq O(f(x))$
v) $F(x) \neq O(x^2)$

**Solution:**

i) Consider

$f(x) = 2x^2 + 3x^2 + 1$

$\leq 2x^3 + 3x^3 + 1x^3 = 6x^3$ for all $x \geq 1$

(by replacing each term $x^i$ by the highest degree term $x^3$)

$\therefore$ there exist $C = 6$ and $k = 1$ such that

$f(x) \leq C, x^3$ for all $x \geq k$

Thus we have found the required constants $C$ and $k$. Hence $f(x)$ is $O(x^3)$

ii) As, above, we can show that

$f(x) \leq 6x^4$ for all $x \geq 1$

However, we may also, by computing some value of $f(x)$ and $x^4$, find $C$ and $k$ as follows

$f(1) = 2 + 3 + 1 = 6$ ; $(1)^4 = 1$

$f(2) = 2.2^3 + 3.2^2 + 1 = 29$ ; $(2)^4 = 16$

$f(3) = 2.3^3 + 3.3^2 + 1 = 82$ ; $(3)^4 = 81$

for $C = 2$ and $k = 3$ we have

$f(x) \leq 2.\ x^4$ for all $x \geq k$

Hence $f(x)$ is $O(x4)$

iii) for $C = 1$ and $k = 1$ we get

$x^3 \leq C\ (2x^3 + 3x^2 + 1)$ for all $x \geq k$

iv) We prove the result by contradiction. Let there exist positive constants C and k such that

$x^4 \leq C\ (2x^3 + 3x^2 + 1)$ for all $x \geq k$

$\therefore x^4 \leq C\ (2x^3 + 3x^3 + x^3) = 6\ Cx^3$ for $x \geq k$

$\therefore\ x^4 \leq 6\ Cx^3$ for all $x \geq k$

Implying $x \leq 6C$ for all $x \geq k$

But for $x = max\ \{6C + 1, k\}$, the previous statement is not true.

Hence the proof.

v) Again we establish the result by contradiction.

Let $O(2x^3 + 3x^2 + 1) = x^2$

Then for some positive numbers $C$ and $k$

$2x^3 + 3x^2 + 1 \leq Cx^2$ for all $x \geq k$,

Implying

$x^3 \leq C\ x^2$ for all $x \geq k$ ($\because x^3 \leq 2x^3 + 3x^2 + 1$ for all $x \geq 1$)

implying

$x \leq C$ for $x \geq k$

Again for $x = \max \{C + 1, k\}$

The last inequality does not hold. Hence the result.

**2) The Notation Ω**

The function f(n)= Ω (g(n)) (read as "f of n is omega of g of n") if there exist positive constants c and $n_o$ such that f(n) ≥ =c g(n) for all n, n ≥ $n_o$.

Eg: The function 3n+2= Ω (n) as 3n+2 ≥ 3n for n ≥ 1

The function 3n+3= Ω (n) as 3n+3 ≥ 3n for n ≥ 1

The function $10n^2+4n+2= \Omega (n^2)$ as $10n^2+4n+2 \geq n^2$ for n ≥ 1

The statement f(n)= Ω (g(n)) states that g(n) is only a lower bound on f(n).

**3) The Notation Θ**

This notation provides simultaneously both asymptotic lower bound and asymptotic upper bound for a given function.

Let *f(x)* and *g(x)* be two functions, each from the set of natural numbers or positive real numbers to positive real numbers. Then, *f(x)* is said to be *Θ (g(x))* (pronounced as big-theta of *g* of *x*) if, there exists positive constants $C_1, C_2$ and *k* such that $C_2\, g(x) \leq f(x) \leq C_1 g(x)$ for all $x \geq k$.

(Note the last inequalities represent two conditions to be satisfied simultaneously viz.,

$C_2\, g(x) \leq f(x)$ and $f(x) \leq C_1 g(x)$)

Now we state the following theorem without proof which relates the three functions *O Ω Θ*

**Theorem:** For any two functions *f(x)* and *g(x), f(x) = Θ (g(x))* if and only if *f (x) = 0* (*g(x)*) and *f(x) = Ω (g(x))* where *f(x)* and *g(x)* are non-negative.

***Example:*** Let $f(n)\colon 1 + 2 + .... + n,\ n \geq 1$. Show that $f(n) = \theta (n^2)$.

***Solution:*** First, we find an upper bound for the sum. For $n \geq 1$, consider

$1 + 2 + \ldots\ldots + n \leq n + n + \ldots.. + n = n.\ n = n^2$

This implies that $f(n) = O(n^2)$. Next, we obtain a lower bound for the sum.

We have

$$1+2+...+n = 1+2+...+\left\lceil \frac{n}{2} \right\rceil + ..... + n +$$

$$\geq \left\lceil \frac{n}{2} \right\rceil + ..... + n$$

$$\geq \left\lceil \frac{n}{2} \right\rceil + \left\lceil \frac{n}{2} \right\rceil + .. + \left\lceil \frac{n}{2} \right\rceil$$

$$\geq \frac{n}{2} \cdot \frac{n}{2}$$

$$= \frac{n^2}{4}.$$

This proves that $f(n) = \Omega(n^2)$. Thus by the above. Theorem $f(n) = \theta(n^2)$.

**4) The Notation $o$**

The asymptotic upper bound provided by big-oh notation may or may not be tight in the sense that if $f(x) = 2x^3 + 3x^2 + 1$. Then for $f(x) = O(x^3)$, though there exist $C$ and $k$ such that $f(x) \leq C(x^3)$ for all $x \geq k$ yet there may also be some values for which the following equality also holds

$f(x) = C(x^3)$ for $x \geq k$ However, if we consider $f(x) = O(x^4)$ then there can not exist positive integer $C$ such that $f(x) = Cx^4$ for all $x \geq k$.

The case of $f(x) = O(x^4)$ provides an example for the notation of small-oh i.e., notation o

Let $f(x)$ and $g(x)$ be two functions, each from the set of natural numbers or positive real numbers to positive real numbers.

Further, let $C > 0$ be any number, then $f(x) = o(g(x))$ (pronounced as little oh of g of x) if there exists natural number $k$ satisfying

$f(x) < Cg(x)$ for all $x \geq k \geq 1$ ………..(B)

Here we get the following points

i) In the case of little-oh the constant $C$ does not depend on the two functions $f(x)$ and $g(x)$. instead, we can arbitrarily choose $C > 0$

ii) The inequality $(B)$ is strict whereas the inequality $(A)$ of big — oh is not necessarily strict.

***Examples:***

For $f(x) = 2x^3 + 3x^2 + 1$, we have

i) $f(x) = o(x^n)$ for any $n \geq 4$

ii) $f(x) \neq o(x^n)$ for any $n \leq 3$

**Solutions:**

i) Let C > 0 be given and to find out $k$ satisfying the requirement of little-oh.

Consider

$$2x^3 + 3x^2 + 1 < C\,x^n = 2 + \frac{3}{x} + \frac{1}{x^3} < Cx^{n-3}$$

When $n = 4$

Then, above inequality becomes

$$= 2 + \frac{3}{x} + \frac{1}{x^3} < Cx$$

If we take $k = max\left\{\frac{7}{C}, 1\right\}$

Then,

$2x^3 + 3x^2 + 1 < C\,x^4$ for $n \geq k$. In general, as $x^n > x^4$ for $n \geq 4$, therefore, $2x^3 + 3x^2 + 1 < C\,x^n$ for $n \geq 4$.For all $x \geq k$ with $k = max\left\{\frac{7}{C}, 1\right\}$

ii) We prove the result by contradiction Let, if possible, $f(x) = 0\ (x)$ for $n \leq 3$. Then there exist positive constants $C$ and $k$ such that $2x^3 + 3x^2 + I < C\,x^n$ for all $x \geq k$.

Dividing by $x^3$ throughout, we get

$$2 + \frac{3}{x} + \frac{1}{x^2} < C\,x^{n-3}$$

As $C$ is arbitrary, we take $C = 1$, then the above inequality reduces to

$$2 + \frac{3}{x} + \frac{1}{x^2} < C\,x^{n-3} \text{ for } n \leq 3 \text{ and } x \geq k \geq 1$$

Also, it can be easily seen that $x^{n-3} \leq 1$ for $n \leq 3$ and $x \geq k \geq 1$.

$$\therefore 2+\frac{3}{x}+\frac{1}{x^2}\leq 1 \qquad \text{for} \quad n \leq 3$$

However, the last inequality is not true. Therefore, the proof by contradiction. Generalizing the above example, we get the results

*f(x)* is a polynomial of degree *m* and *g(x)* is a polynomial of degree *n*. Then *f(x) = o(g(x))* if and only if $n > m$

We state below the two results which can be useful in finding small-oh upper bound for a given function (without proof).

More generally, we have

**Theorem:** Let *f(x)* and *g (x)* be functions in definition of small — oh notation. Then *f(x) = o(g (x))* if and only if

$$\lim_{x\to\infty}\frac{f(x)}{g(x)}=0$$

Next, we introduce the last asymptotic notation, namely, small-omega. The relation of small -omega to big-omega is similar to what is the relation of small-oh to big oh.

**5) The Notation $\omega$**

Again the asymptotic lower bound $\Omega$ may or may not be tight. However, the asymptotic bound $\omega$ cannot be tight. The definition of $\omega$ is as follows:

Let *f(x)* and *g(x)* be two functions each from the set of natural numbers or the set of positive real numbers to set of positive real numbers.

Further, let $C > 0$ be any number, then, $f(x) = \omega(g(x))$ if there exists a positive integer *k* Such that $f(x) > C\,g(x)$ for all $x \geq k$.

***Example:***

Lf $f(x) = 2x^3 + 3x^2 + 1$ then $f(x) = \omega$ and also $f(x) = \omega(x^2)$

**Solution:**

Let *C* be any positive constant

Consider $2x^3 + 3x^2 + 1 > Cx$

To find out $k \geq 1$ satisfying the Conditions of the bound $\omega$.

$2x^2 + 3x + \frac{1}{x} > C$ (dividing through out by $x$)

Let $k$ be integer with $k \geq C + 1$ then for all $x \geq k$

$$2x^2 + 3x + \frac{1}{x} \geq 2x^2 + 3x > 2k^2 + 3K > 2C^2 + 3C > C \qquad \left(\because k \geq C^1 - 1\right)$$

$\therefore f(x) = \omega(x)$

Again, consider for any $C > 0,\ 2x^3 + 3x^2 + I > C > x^2$ then

$2x + 3 + \frac{1}{x^2} > C$ Let $k$ be integer with $k \geq C + 1$

Then, for $x \geq k$ we have $2x + 3\frac{1}{x^2} \geq 2x + 3 > 2k + 3 > 2C + 3 > C$

Hence

$f(x) = \omega(x^2)$

In general, we have the following two theorems

**Theorem:** If $f(x)$ is a Polynomial of degree $n$, and $g(x)$ is a polynomial of degree $n$, then

$f(x) = \omega\,(g(x))$ if and only if $m > n.$

**Theorem:** Let $f(x)$ and $g(x)$ be functions in the definitions of little-omega.

Then $f(x) = \omega\,(g(x))$ if and only if

$$\lim_{x\to\infty} \frac{f(x)}{g(x)} = \infty$$

$$\lim_{x\to\infty} \frac{g(x)}{f(x)} = 0$$

**Self Assessment Questions**

7. The ———— provides asymptotic upper bound for a given function.

8. Let $f(x)$ and $g\,(x)$ be functions in definition of small-oh notation. Then $f(x) = o(g\,(x))$ if and only if ————.

## 10.7 Analysis of Algorithms — Simple Examples

Now we study the different types of analysis of Algorithms with simple illustrations.

**Compute Prefix Average:** For a given array *A[1 …n]* of numbers, the problem is concerned with finding an array *B[1…n]* such that

*B [1] = A[1]*

*B [2]* = average of first two entries = $\frac{(A[1]+A[2])}{2}$

*B [3]*=average of first *3* entries=$\frac{(A[1]+A[2]+A[3])}{3}$ and in general for $1 \le i \le n$,

*B[ i ]* = average of first *i* entries in the array *A [1…n]* = $\frac{A[1]+A[2]+....A[i]}{i}$

```
Algorithm first-prefix-Average (A [1…n])
begin (of algorithm)
for i ← 1 to n do
begin (first for-loop)
Sum ← 0;
{sum stores the sum of first i terms, obtained in different iterations of for-loop}
for j ← 1 to i do
begin {of second for-loop}
Sum ← Sum + A [ j ];
end {of second for loop}
B [ i ] ← Sum / i
End {of the first for — loop}
end {of algorithm}
```

**Analysis of First-Averages**

The different steps involved in the analysis of first averages is discussed below

**Step 1:** Initialization step for setting up of the array *A [1. .n]* takes constant time, say, $C_1$, in view of the fact that for the purpose, only address of *A* (or of

*A[1]* is to be passed. Also after all the values of *B [1...n]* are computed, then returning the array *B [1…n]* also takes constant time say $C_2$, again for the same reason.

**Step 2:** The body of the algorithm has two nested for-loops, the outer one, called the first for loop is controlled by *i* and is executed *n* times. Hence, the second for-loop along with its body, which form a part of the first for-loop, is executed *n* times. Further, each construct within second for-loop controlled by *j*, is executed *i* times just because of the iteration of the second for-loop. However, the second for-loop itself is being executed *n* times because of the first for-loop. Hence, each instruction within the second for-loop is executed *(n, i)* times for each values of

*i = 1, 2, . . .,n.*

**Step 3:** In addition, each controlling variable *i* and *j* is incremented by *1* after each iteration of *i* or *j* as the case may be. Also, after each increment in the control variable, it is compared with the (upper limit + *1*) of the loop to stop the further execution of the for-loop.

Thus, the first-for-loop makes *n* additions (to reach *(n + 1)* and n comparisons with (*n* + *1*)).

The second for-loop makes, for each value of *i = 1, 2,……n*, one addition and one comparison. Thus, total number of each of additions and comparisons done just for controlling variable *j*

$$= (1 + 2 + ...... + n) = \frac{n(n+1)}{2}$$

**Step 4:** Using the explanation of steps *2*, we count below the number of times the various operations are executed.

i) Sum ← *0* is executed *n* times, once for each value of *i* from *1* to *n*.

ii) On the similar lines of how we counted the number of additions and comparisons for the control variable *j*, it can be seen that the number of additions (sum ← sum + *A [ j ]* and divisions (*B [ i ]* ← sum / *i*) is $\frac{n\,(n+1)}{2}$

**Self Assessment Question**

9. 1 + 2 + ……….. + n = ________

## 10.8 Summary

- We Studied the different types of functions and Notations with well illustrated example.
- The concept of Modular arithmetic/ Mod function is well defined with simple examples.
- The concept of Mathematical Expectation and its use in average case analysis of algorithms is well illustrated.
- In this unit, we discussed that if a problem is algorithmically solvable then it may have more than one algorithmic solutions. In order to choose the best out of the available solutions, there are criteria for making such a choice. The complexity/efficiency measures or criteria are based on requirement of computer resources by each of the available solutions.
- In this unit, we also studied the different types of Asymptotic functions and their Notations.

## 10.9 Terminal Questions

1. Define an injective function
2. Define monotonic functions
3. Define exponential function with an example
4. Write the advantages of theoretical approach over the empirical approach
5. For the function $f(x) = 2x^3 + 3x^2 + 1$,
   show that

   i) $f(x) = \Theta(x^3)$

   ii) $f(x) \neq \Theta(x^2)$

   iii) $f(x) \neq \Theta(x^4)$

## 10.10 Answers

**Self Assessment Questions**

1. $f(x) \leq f(y)$
2. Ceiling function
3. Exponentiation function
4. $e^x$
5. Mathematical Expectation

6. time complexity
7. Notation $O$
8. $\lim\limits_{x \to \infty} \frac{f(x)}{g(x)} = 0$
9. $\frac{n(n+1)}{2}$

**Terminal Questions**

1. Refer Section 10.2
2. Refer Section 10.2
3. Refer Section 10.3
4. Refer Section 10.6
5. i) for $C_1 = 3$, $C_2 = 1$ and $k = 4$ *i.e. 1.* $C_2\, x^3 \le f(x) \le C_1\, x^3$ for all $x \ge k$

   ii) We can show by contradiction that no $C_1$ exists. Let, if possible for some positive integers $k$ and $C_1$, we have $2x^3 + 3x^2 + 1 \le C_1 .\ x^2$ for all $x \ge k$ hence $x^3 \le C_1\, x^2$ for all $x \ge k$

   i.e.,$x \le C_1$ for all $x \ge k$

   But for $x = \max \{C_1 + 1, k\}$

   The last inequality is not true.

   *Therefore,* $x = max \{C_1 + 1, k\}$

   iii) $f(x) \ne (x^4)$

   we can show by contradiction that there does not exist $C_2$

   such that

   $C_2\, x^4 \le (2x^3 + 3x^2 + 1)$

   If such a $C_2$ exists for some $k$ then $C_2\, x^4 \le 2x^3 + 3x^2 + 1 \le 6x^3$ for all $x \ge 1$, $k \ge 1$, implying $C_2\, x \le 6$ for all $x \ge k$

   But for $x = \left(\frac{6}{C_2} + 1\right)$

   The above inequality is false. Hence, proof by contradiction.