AD3351- DESIGN AND ANALYSIS OF ALGORITHMS

UNIT-1

INTRODUCTION

Notion of an Algorithm – Fundamentals of Algorithmic Problem Solving – Important Problem Types – Fundamentals of the Analysis of Algorithmic Efficiency –Asymptotic Notations and their properties. Analysis Framework – Empirical analysis – Mathematical analysis for Recursive and Non-recursive algorithms – Visualization

1. NOTION OF AN ALGORITHM:

An ***algorithm*** is a sequence of unambiguous instructions for solving a problem, i.e., for obtaining a required output for any legitimate input in a finite amount of time.

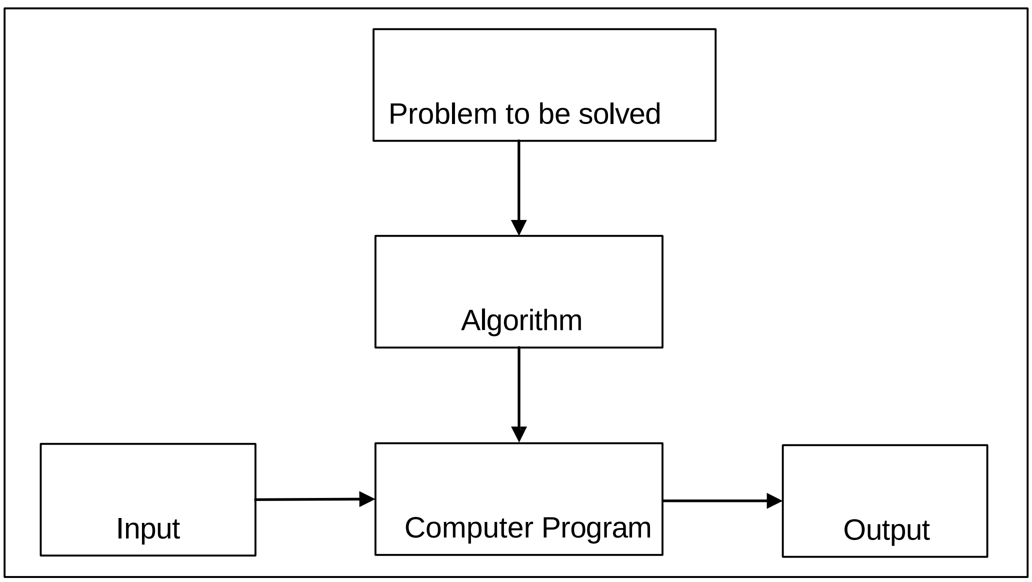

FIGURE 1.1 The notion of the algorithm.

It is a step by step procedure with the input to solve the problem in a finite amount of time to obtain the required output.

The notion of the algorithm illustrates some important points:

- The non-ambiguity requirement for each step of an algorithm cannot be compromised.
- The range of inputs for which an algorithm works has to be specified carefully.
- The same algorithm can be represented in several different ways.
- There may exist several algorithms for solving the same problem.
- Algorithms for the same problem can be based on very different ideas and can solve

the problem with dramatically different speeds.

Characteristics of an algorithm:

Input : Zero / more quantities are externally supplied**. Output**

: At least one quantity is produced.

Definiteness: Each instruction is clear and unambiguous.

Finiteness: If the instructions of an algorithm is traced then for all cases the algorithm must terminates after a finite number of steps.

Efficiency: Every instruction must be very basic and runs in short time.

Steps for writing an algorithm:

1. An algorithm is a procedure. It has two parts; the first part is **head** and the second part is body.
2. The Head section consists of keyword **Algorithm** and Name of the algorithm with parameter list. E.g. Algorithm name1(p1, p2,…,p3)

   The head section also has the following:

   //Problem Description:

   //Input:

   //Output:

3. In the body of an algorithm various programming constructs like **if, for, while** and some statements like assignments are used.
4. The compound statements may be enclosed with **{** and**}** brackets. **if**, **for**, **while** can be closed by **end if**, **end for**, **end while** respectively. Proper indention is must for block.
5. Comments are written using **//** at the beginning.
6. The **identifier** should begin by a letter and not by digit. It contains alpha numeric letters after first letter. No need to mention data types.
7. The left arrow **"←"** used as assignment operator. E.g.v←10
8. **Boolean**operators(TRUE,FALSE),**Logical**operators(AND,OR,NOT)and**Re lational**

operators (<,<=, >, >=,=, ≠, <>) are also used.

9. Input and Output can be done using **read** and **write**.
10. **Array []**, **if then else condition**, **branch** and **loop** can be also used in Algorithm.

Example:

The greatest common divisor(GCD) of two nonnegative integers *m* and *n* (not-both-zero), denoted gcd*(m, n)*, is defined as the largest integer that divides both *m* and *n* evenly, i.e., with a remainder of zero.

Euclid's algorithm is based on applying repeatedly the equality gcd*(m, n)* = gcd*(n, m* mod*n),*

where *m* mod *n* is the remainder of the division of *m* by *n*, until *m* mod *n* is equal to 0. Since gcd*(m,*
0*)* = *m*, the last value of *m* is also the greatest common divisor of the initial *m* and *n*.
gcd*(*60, 24*)* can be computed as follows:gcd*(*60, 24*)*
= gcd*(*24, 12*)* = gcd*(*12, 0*)* = 12.

Euclid's algorithm for computing gcd*(m, n)* in simple steps

Step 1 If *n* = 0, return the value of *m* as the answer and stop; otherwise, proceed to Step 2.

Step 2 Divide *m* by *n* and assign the value of the remainder to *r*.

Step 3 Assign the value of *n* to *m* and the value of *r* to *n*. Go to Step1.

Euclid's algorithm for computing gcd*(m, n)* expressed in pseudocode

ALGORITHM *Euclid_gcd(m, n)*

//Computes gcd*(m, n)* by Euclid's algorithm

//Input: Two nonnegative, not-both-zero integers *m* and *n*

//Output: Greatest common divisor of *m* and *n*

while *n* ≠ 0 **do**

$r \leftarrow m$ mod n

$m \leftarrow n$

$n \leftarrow r$

return *m*