UNIT-1 INTRODUCTION.

Notion of an algorithm - Fundamentals of algorithmic Problem solving - Important problem types - Fundamentals of the analysis of Algorithm efficiency - Analysis Framework - Asymptotic Notations and its properties - Mathematical analysis for Recursive & Non recursive algorithms.

Algorithm - It is a sequence of steps to solve a problem. This design and analysis of algorithm is very important for designing algorithm to solve different types of problems in the branch of CSE. and IT.

Prerequisites:-

-> Basic knowledge of programming
-> Mathematics
-> Data structures.

An algorithm is the best way for solving a particular problem in very simple and an efficient way.

If we have perfect algorithm, we can implement it using any programming language. This means that, algorithm is independent from any programming language

Motivation:-

Advancement in scienc and technology enhances the Performance of processor, which proportionally effect the characteristics of computer security, scalability, reusuability.

There exists vital issues such as sorting, searching, graph problems, combinational problem which serves as motivation for algorithm designing.

Design Goal of DAA :-

→ Solving problem with multiple constraints including size, performance in terms of both space time.

→ Some problems are easy to solve and some are ha

→ Hence designing a new system requires need for ne technology and background of new technology.

Main concern of this course is

→ Correctness of solution

→ Decomposition of application to small and clear units which can be maintained precisely

→ improving the performance of application.

→ An algorithm is a set of rules for carrying out calculations either by hand or machine.

→ Sequential of computational steps

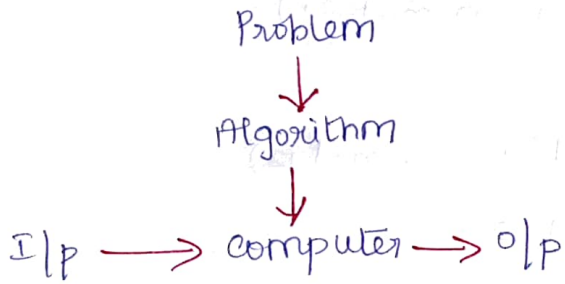
→ Sequence of operations to organize data in DS

→ finite set of instructions to specify operations to carry out some problems.

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 legitimating i/p for finite amount of data and time.



Example:

Gcd of two integer -> 3 ways to solve. Gcd can be denoted as gcd(m,n).

Euclidean algorithm:

$$gcd(m,n) = gcd(n, m \bmod n)$$

Repeated until $m \bmod n = 0$;

as $gcd(m,0) = m$.

Algorithm 1. If n=0, return the value of m as the answer and stop. else proceed with step 2.

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

3. Assign the value of n to m and value of r to n. Goto step 1.

I/p - two non-negative integers

o/p Greatest common divisor of m and n

```
while n=0 do
        r ← m
        mod n
        m ← n
        n ← r
        return m.
```

This algorithm stops when 2nd number becomes zero. For each iteration, the second number in the pair gets smaller and eventually becomes zero and algorithm stops

gcd(60, 24) m=60 24=n

gcd(m, n) = gcd(n, m mod n)

= gcd(24, 60 mod 24)

= gcd(24, 12).

gcd(24, 12) = gcd(12, 24 mod 12)

= gcd(12, 0)

gcd(12, 0) = 12

hence gcd(60, 24) = 12.

24)72
24)48

Algo 2:-

1. → Assign the value of min {m, n} to t
2. → Divide m by t. If the remainder of this division is 0, go to step 3, else to step 4.
3. → Divide n by t. If the remainder is 0, return the value of t as answer, else proceed to step 4.
4. → Decrease the value of t by 1. Go to step 2.

Algo 3:-

1. Finding prime factors of m
2. Finding prime factors of n.
3. Identify all the common factors in two primes expansion found in step 1 and 2.
4. Compute the product of all common factors and return it as gcd of numbers given.

[This algorithm is complex due to prime factorization]

gcd(60, 24) = 12 ⇒ 60 = 2·2·3·5

24 = 2·2·2·3

↓↓ 2·2·3

= 12.