SNS COLLEGE OF ENGINEERING

Kurumbapalayam (Po), Coimbatore – 641 107

An Autonomous Institution

Accredited by NBA – AICTE and Accredited by NAAC – UGC with 'A' Grade
Approved by AICTE, New Delhi & Affiliated to Anna University, Chennai

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

COURSE NAME : 19IT405 DESIGN AND ANALYSIS OF ALGORITHMS

II YEAR /IV SEMESTER

Unit 1- INTRODUCTION

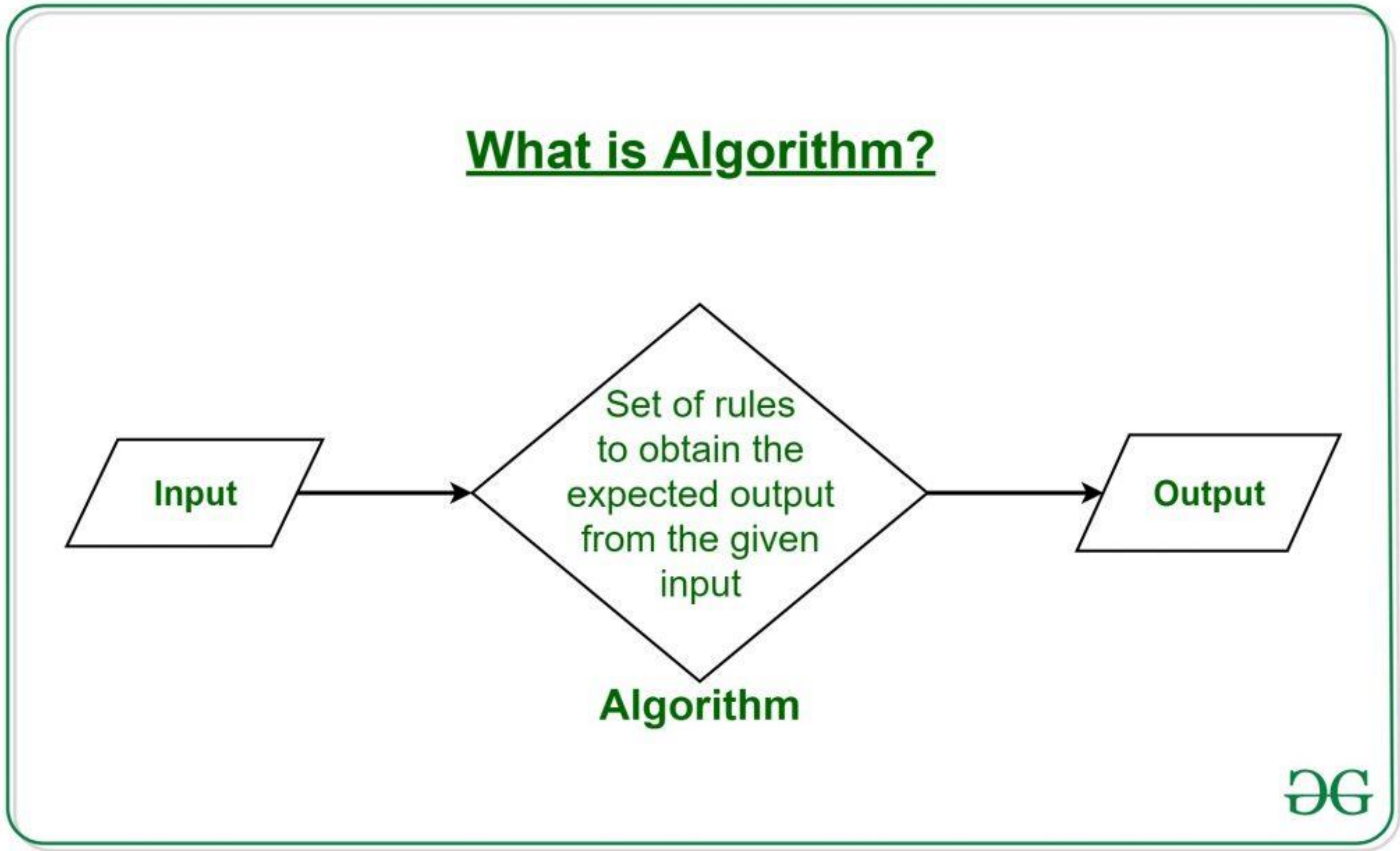
Topic 1: Notion of an Algorithm – Fundamentals of Algorithmic Problem Solving

Brain Storming

1. What is Algorithm?
2. Why it is important?

WHAT IS AN ALGORITHM??

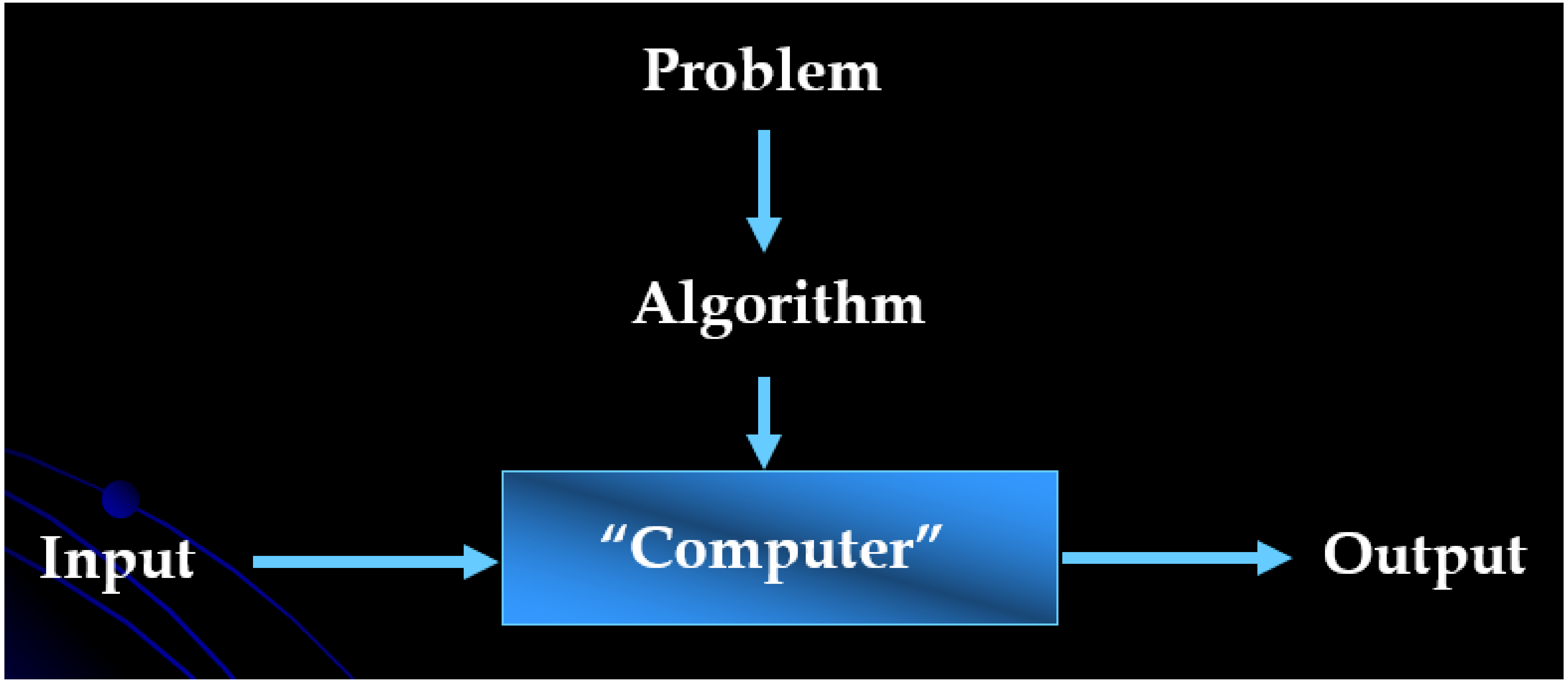
- An algorithm is a sequence of unambiguous instructions for solving a problem, i.e., for obtaining a required output for any legitimate input in a finite amount of time.


What is Algorithm?
Input
Set of rules to obtain the expected output from the given input
Output
Algorithm
ƏG

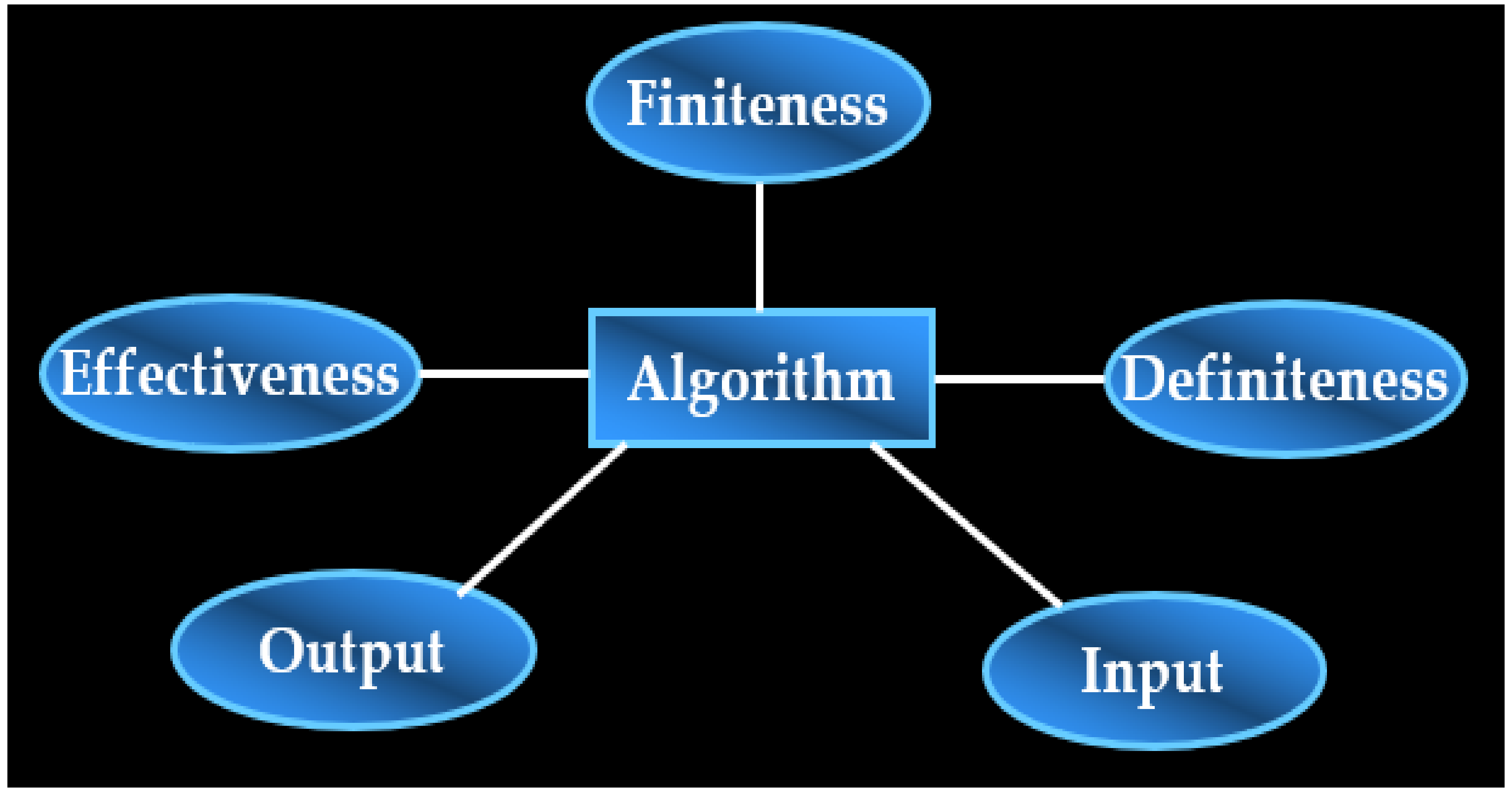
WHY TO STUDY ALGORITHMS ?

- Theoretical importance
  - The core of computer science
- Practical importance
  - A practitioner’s toolkit of known algorithms
  - Framework for designing and analyzing algorithms for new problems

NOTION OF ALGORITHM



PROPERTIES OF AN ALGORITHM:



Recipe, process, method, technique, procedure, routine,… with following requirements:

1. **Finiteness :** terminates after a finite number of steps.
2. **Definiteness:** unambiguously specified.
3. **Input:** valid inputs are clearly specified.
4. **Output:** can be proved to produce the correct output given a valid input.
5. **Effectiveness:** steps are sufficiently simple and basic.

PROPERTIES OF AN ALGORITHM:

Correctness

- Input conditions should be satisfied

Termination

- Algorithm must avoid infinite loop

Performance

- Quantification of the space and time complexities

EUCLID’S ALGORITHM

- **Problem:**

Find gcd(m,n), the greatest common divisor of two nonnegative, not both zero integers m and n

- **Examples:**

gcd(60,24) = 12

gcd(60,0) = 60

EUCLID’S ALGORITHM

- Euclid’s algorithm is based on repeated application of equality

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

  until the second number becomes 0, which makes the problem trivial.

- Example:

  gcd(60,24) = gcd(24,12) = gcd(12,0) = 12

EUCLID’S ALGORITHM

ALGORITHM Euclid(m, n)

Step 1 If n = 0, return m 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 Step 1.

PSEUDOCODE:

ALGORITHM Euclid(m, n)

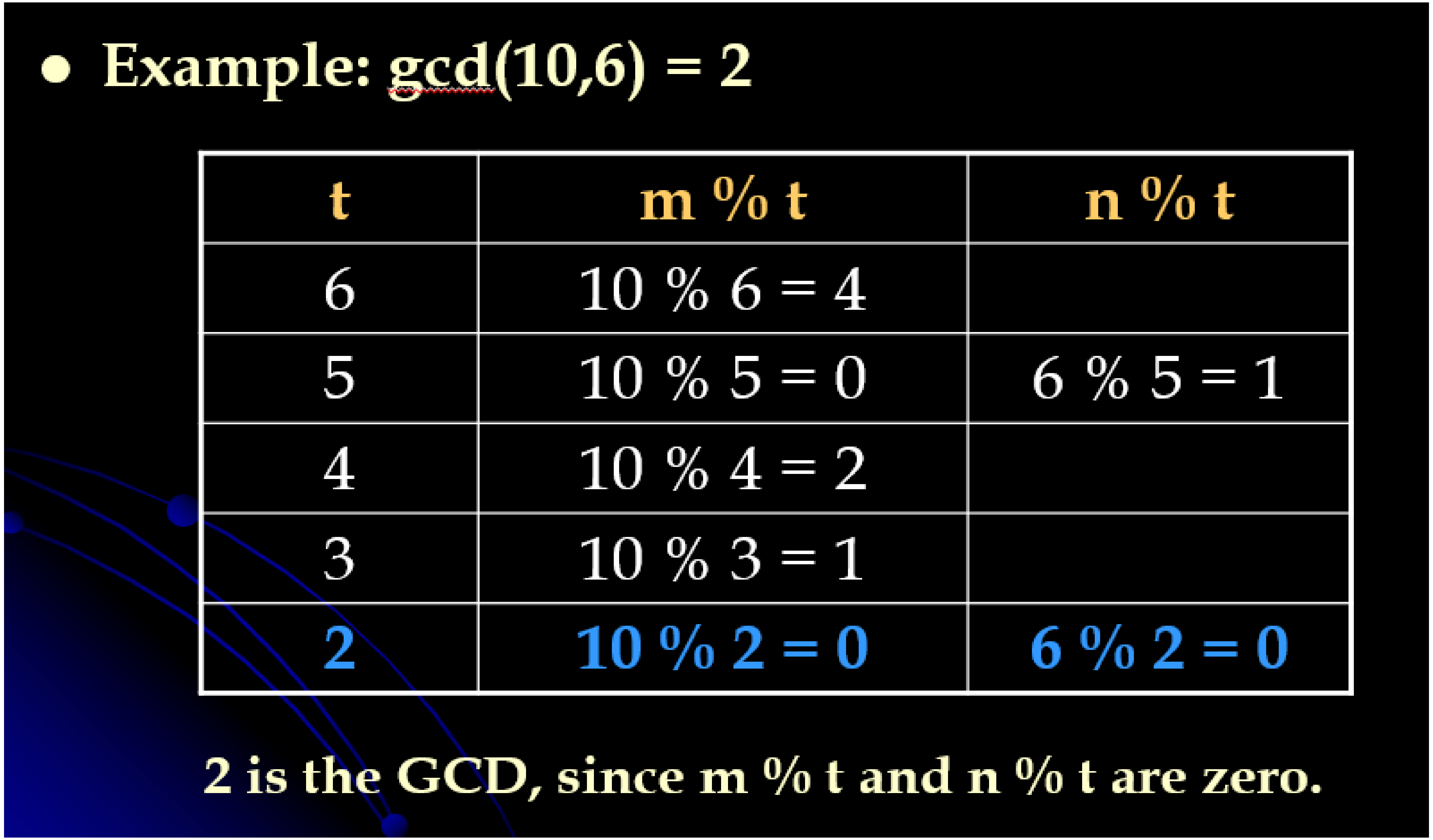
```
// computes gcd(m, n) by Euclid’s algorithm
// Input: Two nonnegative, not-both-zero intgers m and n
// Output: Greatest common divisor of m & n
while n ≠ 0  do
        r ← m mod n
          m← n
          n ← r
return m
```

OTHER METHODS FOR COMPUTING gcd(m,n)

➢Consecutive Integer Checking Algorithm

```
Step 1   Assign the value of min{m,n} to t
Step 2   Divide m by t. If the remainder of this division is 0,
            goto Step 3;
         otherwise, go to Step 4
Step 3   Divide n by t.  If the remainder of this division is 0,
            return the value of t as answer and stop;
         otherwise, proceed to Step 4
Step 4   Decrease the value of t by 1 and go to Step 2
```

Consecutive Integer Checking Algorithm



- Example: gcd(10,6) = 2

t	m % t	n % t
6	10 % 6 = 4	
5	10 % 5 = 0	6 % 5 = 1
4	10 % 4 = 2	
3	10 % 3 = 1	
2	10 % 2 = 0	6 % 2 = 0

2 is the GCD, since m % t and n % t are zero.

OTHER METHODS FOR COMPUTING gcd(m, n) (CONT…)

➢Middle - school procedure

Step 1 Find the prime factors of m.

Step 2 Find the prime factors of n.

Step 3 Identify all the common factors in the two prime expansions found in step1 and step2 (If P is a common factor occuring P_m and P_n times in m and n respectively, it should be repeated min$\{P_m, P_n\}$ times).

Step 4 Compute the product of all the common factors and return it as gcd(m,n)

Example: If m = 60 and n = 24 then

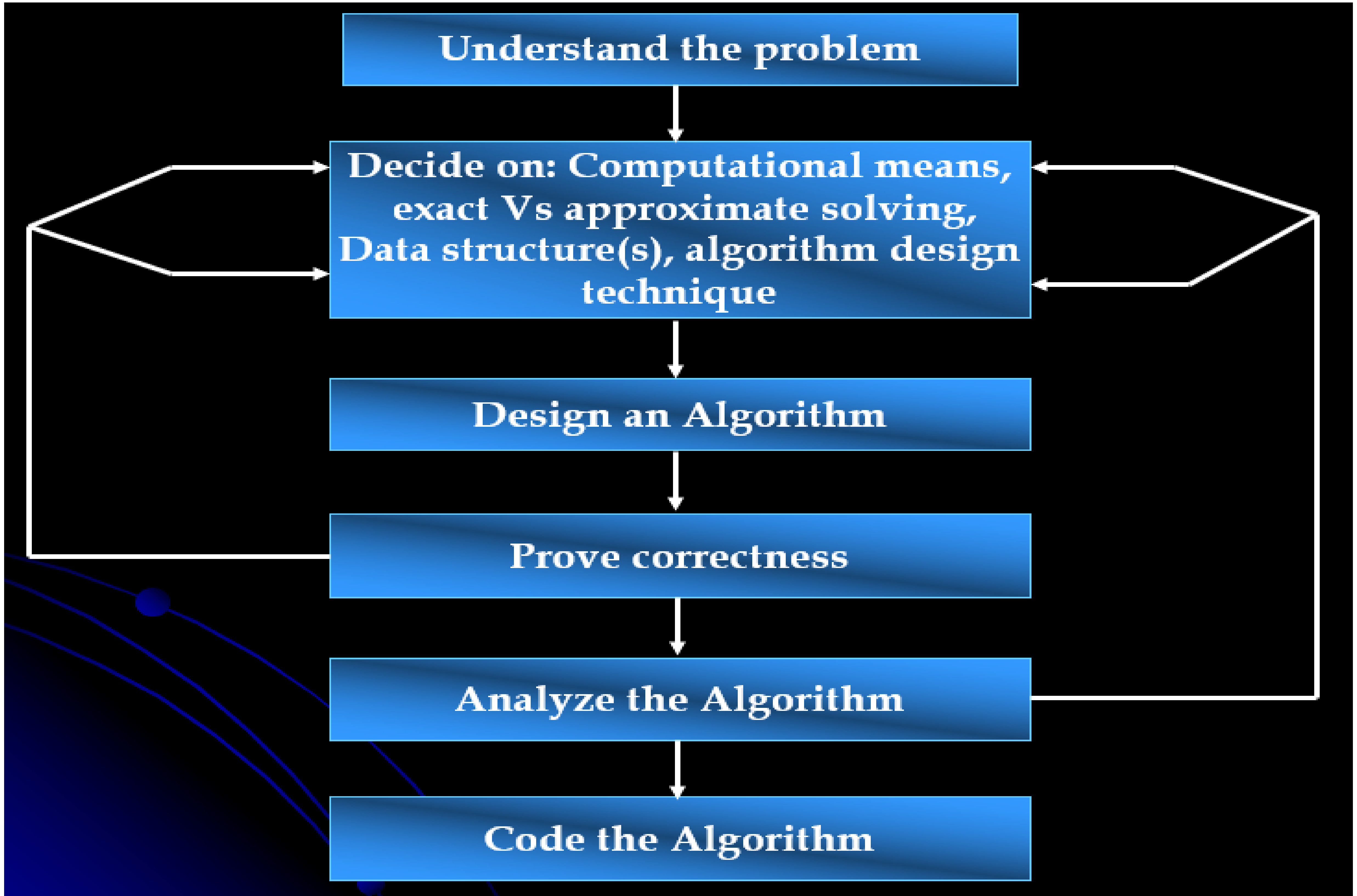
60 = 2 . 2 . 3 . 5

24 = 2 . 2 . 2 . 3

gcd(60,24) = 2 . 2 . 3 =12

Is this an algorithm?

FUNDAMENTALS OF ALGORITHMIC PROBLEM SOLVING



Understand the problem

Decide on: Computational means, exact Vs approximate solving, Data structure(s), algorithm design technique

Design an Algorithm

Prove correctness

Analyze the Algorithm

Code the Algorithm

Fig: Algorithm Design and Analysis Process.

- **Understanding the problem:**

  - What is the range of inputs that should be provided?
  - What is the output expected?

- Activities involved before going for designing the algorithm :

➢ **Ascertaining the capabilities of a Computational Device:**

- make sure about the capabilities of a computational device before designing so as to choose among ***sequential*** algorithms or ***parallel*** algorithms.

Conti…

- **Choosing between Exact and Approximate problem solving:**

  - make decision to choose between solving the problem exactly (***Exact algorithm***) or solving it approximately (***approximation algorithm***).

- **Deciding on Appropriate Data Structure:**

  Algorithms + Data Structures = Programs

- **Algorithm Design Techniques:**

  - Helps you in devising the algorithm.
  - Provide guidance for designing algorithms for new problems.

Conti…

- Designing an Algorithm:

Methods of Specifying an Algorithm:

- **Natural Language.**
- **Pseudocode:** a mixture of a natural language and programming language-like constructs.
- **Flowchart:** a method of expressing an algorithm by a collection of connected geometric shapes containing descriptions of the algorithm's steps.

Conti…..

- **Proving an Algorithm’s Correctness:**

- Prove that the algorithm yields a required result for every legitimate input in a finite amount of time.
- Mathematical Induction, a common technique for proving correctness.

Conti…

- Analyzing an Algorithm:

➢ Efficiency

- *Time efficiency* indicates how fast the algorithm runs

- *Space efficiency* indicates how much memory the algorithm needs.

➢ Simplicity

This theoritical analysis gives the approximate amount of resources required.

Conti…..

- Coding an Algorithm:

  - Algorithm is coded using suitable data structure in programming language.

  - can be tested to know the actual statistics about the algorithm’s consumption of time and space requirements.

  - If less efficient then you can fine tune the code to improve the speed or you can go for better algorithm.

Assessment 1

1. What is algorithm?

Ans : ____

2. Why algorithm effectiveness is important?

Ans : ____

References

TEXT BOOKS

1. Anany Levitin, “Introduction to the Design and Analysis of Algorithms”, Third Edition, Pearson Education, 2012.

REFERENCES

1.Thomas H.Cormen, Charles E.Leiserson, Ronald L. Rivest and Clifford Stein, “Introduction to Algorithms”, Third Edition, PHI Learning Private Limited, 2012.

2. Alfred V. Aho, John E. Hopcroft and Jeffrey D. Ullman, “Data Structures and Algorithms”, Pearson Education, Reprint 2006.

3. Donald E. Knuth, “The Art of Computer Programming”, Volumes 1& 3 Pearson Education, 2009.

4. Steven S. Skiena, “The Algorithm Design Manual”, Second Edition, Springer, 2008.

Thank You