

Module - 1

Introduction to Algorithms

Course Code & Name : CSE270 Design and Analysis of Algorithms

DAY-1 : 19-10-2020

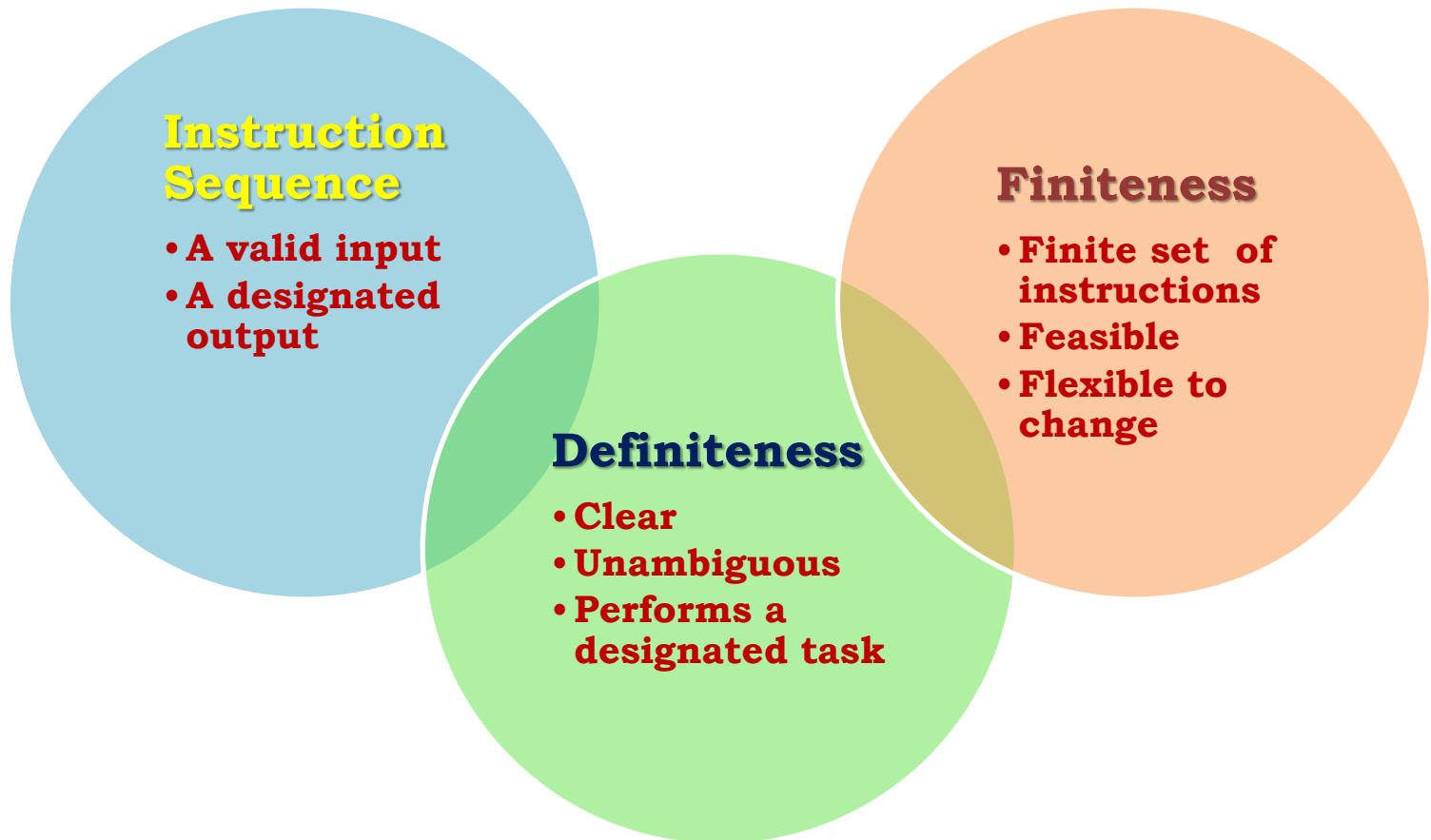
Forenoon : 09:00 AM – 01:00 PM

Contents

- Define the term Algorithm
- Characteristics
- The pros and cons
- The need for an algorithm
- Algorithm development : Step by step
- Important problem types

Algorithm

A finite set of sequence of unambiguous instructions that performs a designated task



“A sequence of unambiguous instructions for solving a problem, i.e. for obtaining the required output for any legitimate input in a finite amount of time” by Any Levintin

Characteristics of Algorithms

- **Input:** It should externally supply zero or more quantities.
- **Output:** It results in at least one quantity.
- **Definiteness:** Each instruction should *be clear and unambiguous*.
- **Finiteness:** An algorithm should terminate after executing a *finite number of steps*.
- **Effectiveness:** Every instruction should be fundamental to be carried out, in principle, by a person using only pen and paper.
- **Feasible:** It must be feasible enough to produce each instruction.
- **Flexibility:** It must be flexible enough to carry out desired changes with no efforts.
- **Efficient:** The term *efficiency is measured in terms of time and space required by an algorithm to implement*. Thus, an algorithm must ensure that it takes little time and less memory space meeting the acceptable limit of development time.
- **Independent:** *An algorithm must be language independent*, which means that it should mainly focus on the input and the procedure required to derive the output instead of depending upon the language.

Comparison

Pros

- **Effective Communication:**
 - written in a natural language like English,
 - it becomes easy to understand the step-by-step delineation of a solution.
- **Easy Debugging:**
 - A well-designed algorithm facilitates easy debugging to detect the logical errors that occurred inside the program.
- **Easy and Efficient Coding:**
 - An algorithm is nothing but a blueprint of a program that helps develop a program.
- **Independent of Programming Language:**
 - It is a language-independent, it can be easily coded by incorporating any high-level language.

Cons

- Developing algorithms for complex problems would be time-consuming and difficult to understand.
- It is a challenging task to understand complex logic through algorithms.

The need for an algorithm

Data

- To understand the basic idea of the problem.
- To find an approach to solve the problem.

Design

- To improve the efficiency of existing techniques.
- To understand the basic principles of designing the algorithms
- To measure the behavior (or performance) of the methods in all cases (best cases, worst cases, average cases)
- To measure and analyze the complexity (time and space) of the problems concerning input size without implementing and running it;
- To reduce the cost of design.

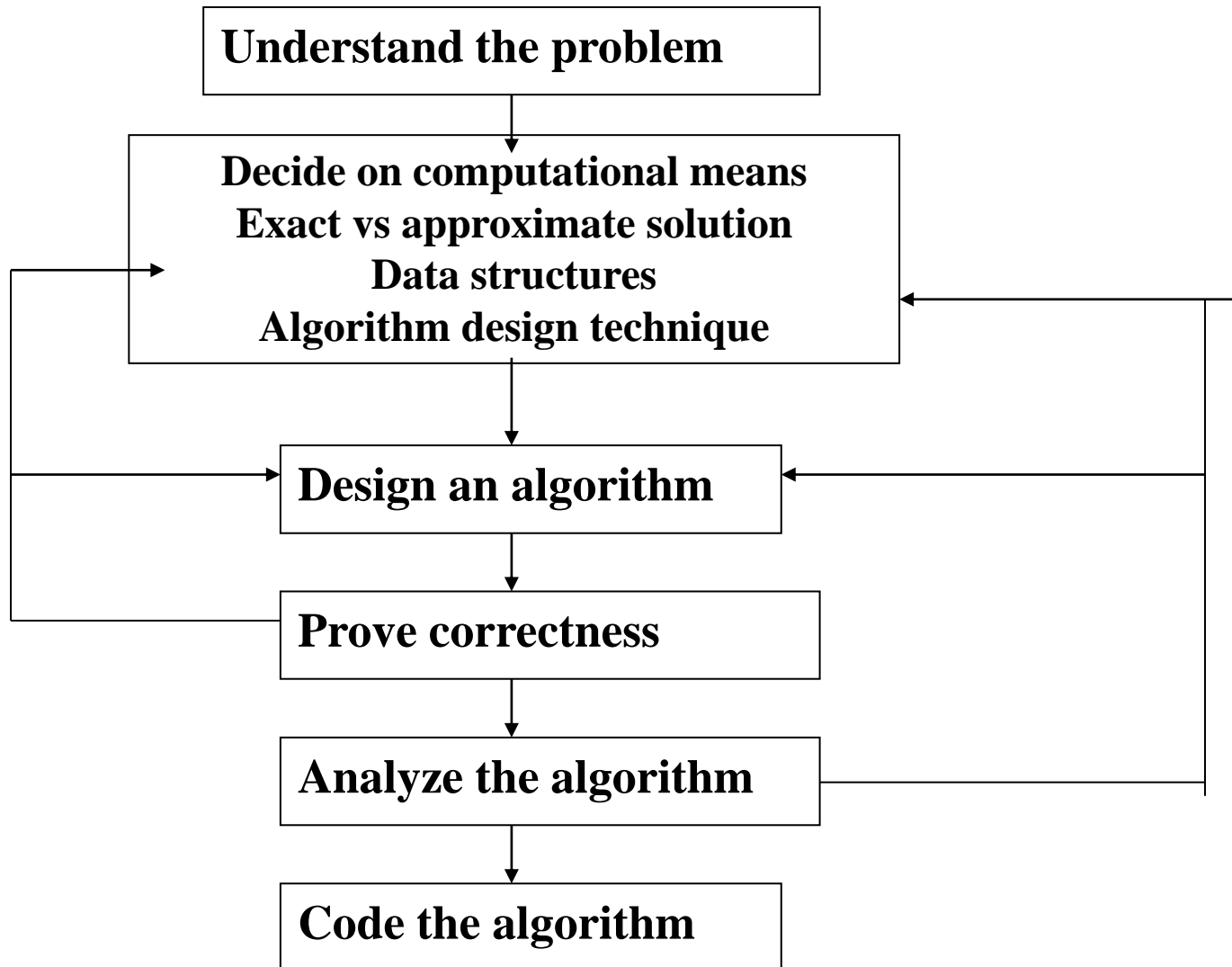
Analyse

- To compare the performance of the algorithm with respect to other techniques.
- It is the best method of description without describing the implementation detail.
- identify the resources (memory, input-output) cycles required by the algorithm.

Goal

- The Algorithm gives a clear description of requirements and goal of the problem to the designer.
- To understand the flow of the problem.

Algorithm development : Step by step method



What does it mean to understand the problem?

- What are the problem objects?
- What are the operations applied to the objects?

Deciding on computational means

- How the objects would be represented?
- How the operations would be implemented?

Design an algorithm

- Build a computational model of the solving process

Prove correctness

- Correct output for every legitimate input in finite time
- Based on correct math formula
- By induction

Analyze the algorithm

Efficiency: time and space

Simplicity

Generality: range of inputs, special cases

Optimality:

no other algorithm can do better

Coding

How the objects and operations in the algorithm are represented in the chosen programming language?

Important problem types

- Sorting
- Searching
- String processing
- Graph problems
- Combinatorial problems
- Geometric problems
- Cryptographic problems
- Developing algorithms for parallel and distributed computing based applications