UNIT I (GE8151 PROBLEM SOLVING AND PYTHON PROGRAMMING)

ALGORITHMIC PROBLEM SOLVING

Algorithms are procedural solutions to problems. Algorithmic problem solving is defined as the formulation and solution of problems where solution involves the principles and techniques to construct the correct algorithms. It means that solving problems that require formulation of an algorithm for their solution.

Steps in designing and analyzing an algorithm

The steps are

i) Understanding the problem

ii) Ascertaining the capabilities of a computational device

iii) Choosing between exact and approximate problem solving

iv) Deciding on appropriate data structures.

v) Algorithm design techniques

vi) Methods of specifying an algorithm.

vii) Proving an algorithm's correctness.

viii) Analyzing an algorithm.

ix) Coding an algorithm.

i) Understanding the problem

→Before designing an algorithm, you need to understand the problem completely.

→Read the problem description carefully.

⟶ Check if it is similar to some standard problems and if a known algorithm exists.

→Ask Questions if you have any doubts and do a few small examples by hand.

→Think about special cases.

→Ask Questions again if needed.

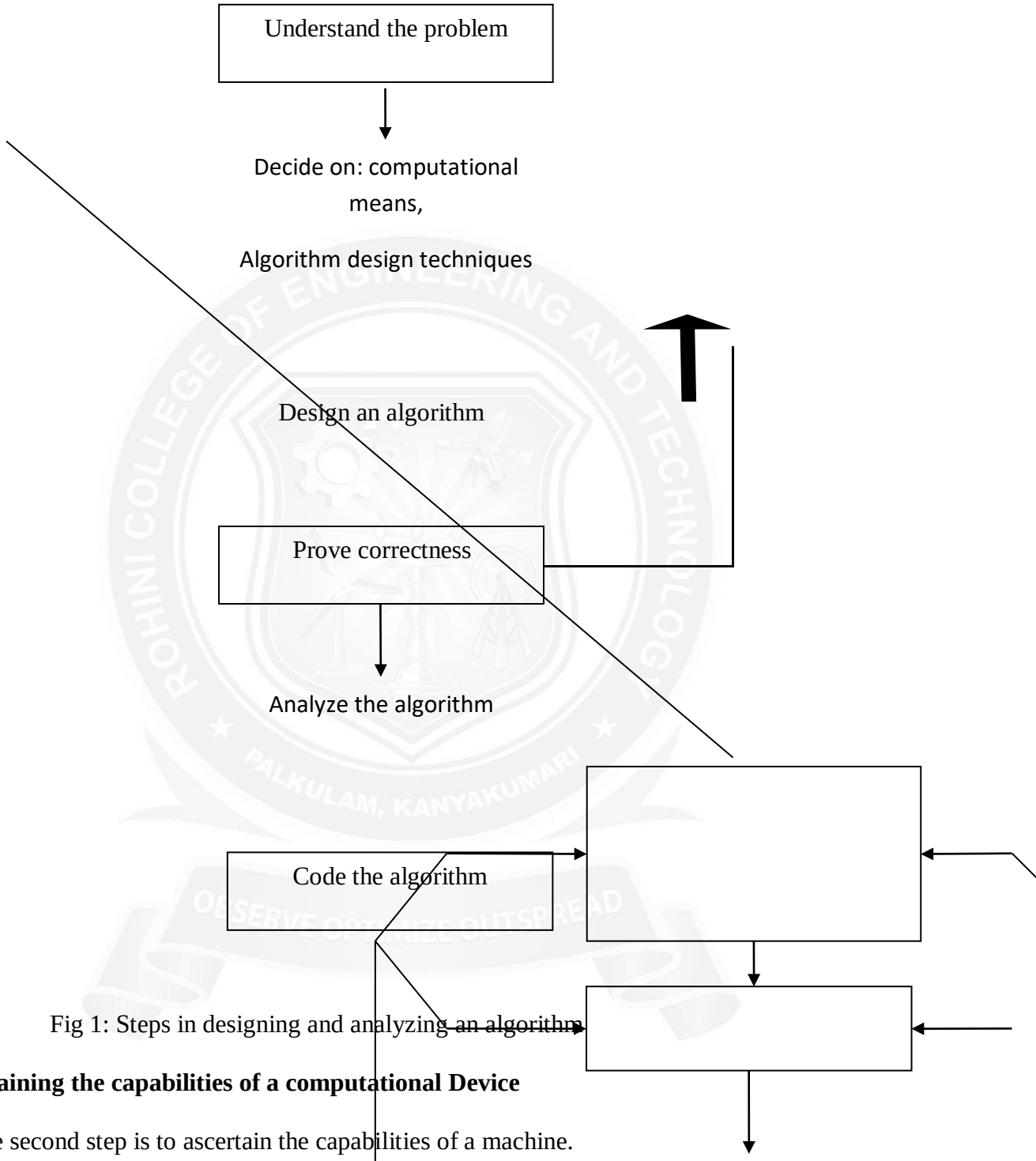

Fig 1: Steps in designing and analyzing an algorithm

ii) Ascertaining the capabilities of a computational Device

The second step is to ascertain the capabilities of a machine.

Sequential algorithms:

⟶ The essence of von-Neumann machines architecture is captured by RAM; here the instructions are executed one after another, one operation at a time.

→Algorithms designed to be executed on such machines are called sequential algorithms.

Parallel algorithms:

⟶ An algorithm which has the capability of executing the operations concurrently iscalled parallel algorithms.

⟶RAM model doesn't support this.

iii) Choosing between exact and approximate problem solving

→The next decision is to choose between solving the problem exactly or solving it approximately.

→Based on this, the algorithms are classified as exact and approximation algorithms.

Exact algorithms:

⟶ The algorithms that can solve the problem exactly are called exact algorithms.

Approximate algorithms:

→The algorithms that can solve the problem not exactly are called approximate algorithms.

→There are three issues to choose an approximation algorithm.

→First, there are certain problems like extracting square roots, solving non-linear equations which cannot be solved exactly.

→Secondly, if the problem is complicated it slows the operations. E.g. Traveling salesman problem.

→Third, this algorithm can be a part of a more sophisticated algorithm that solves aproblem exactly.

iv) Deciding on appropriate data structures

→Data structures play a vital role in designing and analyzing the algorithms.

→Some of the algorithm design techniques also depend on the structuring data specifying a problem's instance.

Algorithm + Data structure = Programs

v) Algorithm design techniques

→An algorithm design technique is a general approach to solving problems algorithmically that is applicable to a variety of problems from different areas of computing.

→Learning these techniques is important for two reasons.

i) First, they provide guidance for designing for new problems.

ii) Second, algorithms are the cornerstones of computer science.

vi) Methods of specifying an algorithm

Once you have designed an algorithm, you need to specify it in some fashion. There are two options for specifying algorithms.

⟶ Pseudo code

⟶ Flowchart

Pseudo code is a mixture of a natural language and programming language. It is more precise.

Flowchart is a method of expressing an algorithm by a collection of connected geometric shapes containing descriptions of the algorithm's steps.

vii) Proving an algorithm's correctness

→Once an algorithm has been specified, you have to prove its correctness.

→You have to prove that the algorithm yields a required result for every legitimate input in a finite amount of time.

→A common technique for proving correctness is to use mathematical induction.

viii) Analyzing an algorithm

⟶ Our algorithm should possess several qualities.

⟶ Analysis of algorithms and performance analysis refers to the task of determining how much computing time and storage an algorithm requires.

→After correctness, the most important quality of an algorithm is Efficiency.

→There are two kinds of algorithm Efficiency.

i) **Time efficiency**: It indicates how fast the algorithm runs.

ii) **Space efficiency:** Indicates how much extra memory the algorithm needs.

ix) Coding an algorithm

⟶ Algorithms are implemented as computer programs.

⟶ Verification and validation is done for error correction.

ROHINI COLLEGE OF ENGINEERING AND TECHNOLOGY
PALKULAM, KANYAKUMARI
OBSERVE OPTIMIZE OUTSPREAD