DAA - Introduction

Previous Page Next Page

An algorithm is a set of steps of operations to solve a problem performing calculation, data processing, and automated reasoning tasks. An algorithm is an efficient method that can be expressed within finite amount of time and space.

An algorithm is the best way to represent the solution of a particular problem in a very simple and efficient way. If we have an algorithm for a specific problem, then we can implement it in any programming language, meaning that the **algorithm is independent from any programming languages**.

Algorithm Design

The important aspects of algorithm design include creating an efficient algorithm to solve a problem in an efficient way using minimum time and space.

To solve a problem, different approaches can be followed. Some of them can be efficient with respect to time consumption, whereas other approaches may be memory efficient. However, one has to keep in mind that both time consumption and memory usage cannot be

Problem definition

Development of a model

Specification of an Algorithm

Designing an Algorithm

Checking the correctness of an Algorithm

Analysis of an Algorithm

Implementation of an Algorithm

Program testing

Documentation

Characteristics of Algorithms

The main characteristics of algorithms are as follows −

Algorithms must have a unique name

Algorithms should have explicitly defined set of inputs and outputs

Algorithms are well-ordered with unambiguous operations

Algorithms halt in a finite amount of time. Algorithms should not run for infinity, i.e., an algorithm must end at some point

Pseudocode

Pseudocode gives a high-level description of an algorithm without the ambiguity associated with plain text but also without the need to know the syntax of a particular programming language.

The running time can be estimated in a more general manner by using Pseudocode to represent the algorithm as a set of fundamental operations which can then be counted.

On the other hand, pseudocode is an informal and (often rudimentary) human readable description of an algorithm leaving many granular details of it. Writing a pseudocode has no restriction of styles and its only objective is to describe the high level steps of algorithm in a much realistic manner in natural language.

For example, following is an algorithm for Insertion Sort.

```
Algorithm: Insertion-Sort
Input: A list L of integers of length n
Output: A sorted list L1 containing those integers present in L
Step 1: Keep a sorted list L1 which starts off empty
Step 2: Perform Step 3 for each element in the original list L
Step 3: Insert it into the correct position in the sorted list L1.
Step 4: Return the sorted list
Step 5: Stop
```

Here is a pseudocode which describes how the high level abstract process mentioned above in the algorithm Insertion-Sort could be described in a more realistic way.

```
for i <- 1 to length(A)
   x <- A[i]
   j <- i
   while j > 0 and A[j-1] > x
      A[j] <- A[j-1]
      j <- j - 1
   A[j] <- x
```

In this tutorial, algorithms will be presented in the form of pseudocode, that is similar in many respects to C, C++, Java, Python, and other programming languages.

Previous Page Next Page