A Compendium of Basic Algorithms

Melvin Cabatuan
Electronics and Computer Engineering Department
De La Salle University
Manila, Philippines
Email: melvin.cabatuan@dlsu.edu.ph

I. Introduction

This lecture notes includes a collection of commonly used algorithms in an introductory course for a computational background. An algorithm is a finite set of precise instructions for performing a computation or for solving a problem. An algorithm has input values from a specified set. From each set of input values, an algorithm produces output values. An algorithm should yield the correct output values for each set of input values. Furthermore, the steps of an algorithm must be defined precisely and the desired output must be produced after a finite number of steps. Finally, the procedure should be applicable for all problems of the desired form, not just for a particular set of input.

An algorithm is often described using a pseudocode which is a high-level description of an algorithm that uses the structural conventions of a standard programming language, but is intended for human reading. The following sections describe an algorithm in terms of a pseudocode with actual implementation.

II. Finding the Maximum

Algorithm 1 Calculate *maximum* element in an integer set

Require: $\{a_1, a_2, \ldots, a_i, \ldots, a_n\} \in \mathbb{Z}$
Ensure: $result = max(a_1, a_2, \ldots, a_i, \ldots, a_n)$
  $result \leftarrow a_1$
  for $i = 2$ to n **do**
    if $result < a_i$ **then**
      $result \leftarrow a_i$
    end if
  end for

III. Finding the Minimum

Algorithm 2 Calculate *minimum* element in an integer set

Require: $\{a_1, a_2, \ldots, a_i, \ldots, a_n\} \in \mathbb{Z}$
Ensure: $result = min(a_1, a_2, \ldots, a_i, \ldots, a_n)$
  $result \leftarrow a_1$
  for $i = 2$ to n **do**
    if $result > a_i$ **then**
      $result \leftarrow a_i$
    end if
  end for

IV. Linear Search (Using a For-Loop)

Algorithm 3 Locate an element x in a list of distinct values or determine that it is not in the list.

Require: $\{a_1, a_2, \ldots, a_i, \ldots, a_n\}_{\neq} \in \mathbb{Z}$; $x \in \mathbb{Z}$
Ensure: $result = k$, where $(a_k = x)$ and $k \in \{1, \ldots, n\}$ if the element is found; otherwise $k = -1$
  $result \leftarrow -1$
  for $i = 1$ to n **do**
    if $result == a_i$ **then**
      $result \leftarrow i$
    end if
  end for

V. Linear Search (Using a While-Loop)

Algorithm 4 Locate an element x in a list of distinct values or determine that it is not in the list.

Require: $\{a_1, a_2, \ldots, a_i, \ldots, a_n\}_{\neq} \in \mathbb{Z}$; $x \in \mathbb{Z}$
Ensure: $result = k$, where $(a_k = x)$ and $k \in \{1, \ldots, n\}$ if the element is found; otherwise $k = -1$
  $i \leftarrow 1$
  while $(i \leq n) \wedge (x \neq a_i)$ **do**
    $i \leftarrow i + 1$
  end while
  if $i \leq n$ **then**
    $result \leftarrow i$
  else
    $result \leftarrow -1$
  end if

VI. Binary Search

Algorithm 5 Locate an element x in a list of distinct and sorted values or determine that it is not in the list.

Require: $\{a_1, a_2, \ldots, a_i, \ldots, a_n\}_{\neq} \in \mathbb{Z}$, where $a_1 < a_2 < \ldots < a_n$; $x \in \mathbb{Z}$

Ensure: $result = k$, where $(a_k = x)$ and $k \in \{1, \ldots, n\}$ if the element is found; otherwise $k = -1$

$i \leftarrow 1$
$j \leftarrow n$
while $i < j$ **do**
  $mid \leftarrow \left\lfloor \frac{i+j}{2} \right\rfloor$
  if $x > a_{mid}$ **then**
    $i \leftarrow mid + 1$
  else
    $j \leftarrow mid$
  end if
end while
if $x == a_i$ **then**
  $result \leftarrow i$
else
  $result \leftarrow -1$
end if