Based on K. H. Rosen: Discrete Mathematics and its Applications.

Lecture 12: Algorithms and pseudocode. Section 3.1

1 Algorithms and pseudocode

Definition 1. An **algorithm** is a **finite** sequence of precise instructions for performing a computation or for solving a problem.

PROPERTIES OF ALGORITHMS are several properties that algorithms generally share. They are useful to keep in mind when algorithms are described. These properties are:

1. **Input.** An algorithm has input values from a specified set.
2. **Output.** From each set of input values an algorithm produces output values from a specified set. The output values are the solution to the problem.
3. **Definiteness.** The steps of an algorithm must be defined precisely.
4. **Correctness** An algorithm should produce the correct output values for each set of input values.
5. **Finiteness.** An algorithm should produce the desired output after a finite (but perhaps large) number of steps for any input in the set.
6. **Effectiveness.** It must be possible to perform each step of an algorithm exactly and in a finite amount of time.
7. **Generality.** The procedure should be applicable for all problems of the desired form, not just for a particular set of input values.

Since many programming languages are in use, we do not want to use a particular language to write our algorithms Instead we use a form of **pseudocode**. We proceed to illustrate with the pseudocode associated to several algorithms.

1.1 Maximum value

To find the maximum element in an ordered list of numbers we can proceed like this:

1. Set the temporary maximum equal to the first integer in the sequence. (The temporary maximum will be the largest integer examined at any stage of the procedure.)
2. Compare the next integer in the sequence to the temporary maximum, and if it is larger than the temporary maximum, set the temporary maximum equal to this number.

3. Repeat the previous step if there are more numbers in the sequence.

4. Stop when there are no numbers left in the sequence. The temporary maximum at this point is the largest integer in the sequence.

A pseudocode description of the algorithm for finding the maximum element in a finite sequence follows:

procedure max($a_1, a_2, \ldots, a_n$: numbers)
max $= a_1$
for $i = 2$ **to** n:
    if max $< a[i]$ **then** max $= a_i$
return max (max is the largest element)

1.2 Binary search

Binary search is an efficient algorithm for finding an item from a sorted list of items. The steps of the binary search algorithm are as follows: To search for the integer x in the list $a_1, a_2, \ldots, a_n$, where $a_1 \leq a_2 \leq \cdots \leq a_n$ begin by comparing x with the **middle term** a_m of the list, where $m = \lfloor (n+1)/2 \rfloor$. If $x > a_m$, the search for x is restricted to **the second half** of the list, which is $a_{m+1}, a_{m+2}, \ldots, a_n$. On the other hand if x is not greater than a_m, the search for x is restricted to **the first half** of the list, which is $a_1, a_2, \ldots, a_m$. The search has now been restricted to a list with no more than $\lfloor n/2 \rfloor$ elements. Repeat this process until a list with one term is obtained. Then determine whether this term is x. Pseudocode for the binary search algorithm can be presented as:

procedure binary search(x : integer, $a_1, a_2, \ldots, a_n$: increasing integers)
$i := 1$, (i is left endpoint of search interval)
$j := n$, (j is right endpoint of search interval)
while $i < j$
    $m := \lfloor (i+j)/2 \rfloor$
    if $x > a_m$ **then** $i := m + 1$
        else $j := m$
if $x = a_j$ **then** location is i
        else location is 0
return location (location is the subscript i of the term a_i equal to x, or 0 if x is not found)