**Pseudocode used in CST370**

01/15/2020

Dr. Byun

Pseudocode is an informal high-level description of an algorithm. It typically uses structural conventions of normal programming languages, but omits some statements for machine execution such as variable declarations, system-specific code and some subroutines.[<https://en.wikipedia.org/wiki/Pseudocode>]

There’s no standard notation for pseudocode syntax. But to make it easy to understand the algorithms described in our textbook, this document presents basic notations using a few examples.

Example 1: Calculate the sum of an array with *n* elements

|  |
| --- |
| 1. Algorithm *SequentialAdd* (A[0..n-1]) 2. // Input: An array A with *n* numbers from the index 0 to n-1 3. // Output: Sum of the numbers in the array A 4. sum ← 0 // Assign the value 0 to the variable “sum”   // Note that **←** symbol is used to indicate an **assignment** in the textbook.  // Meanwhile, **=** symbol is used as a **comparison** operator.   1. i ← 0 2. while ( i < n ) do // ‘while’ loop with the comparison operator <. 3. sum ← sum + A[i] // Body of the while loop. Indentation is important in our text. 4. i ← i + 1 // Body of the while loop. 5. return sum // Algorithm returns the sum of the array as a result. |

Example 2: Determine the max value in an array with *n* elements

|  |
| --- |
| 1. Algorithm *FindMax* (A[0..n-1]) 2. // Input: An array A with *n* numbers from the index 0 to n-1 3. // Output: Determine the max number in the array A 4. maxvalue ← A[0] 5. i ← 1 6. while ( i < n ) do 7. if ( A[i] > maxvalue ) //’if’ statement with the comparison operator 8. maxvalue ← A[i] 9. i ← i + 1 10. return maxvalue |

The following example calculates an average of *n* numbers in an array. Note that the pseudocode has a for loop.

Example 3: Determine the average of an array using the “for” loop

|  |
| --- |
| 1. Algorithm *Average* (A[0..n-1]) 2. // Input: An array A with *n* numbers from the index 0 to n-1 3. // Output: Average of the numbers in the array A 4. sum ← A[0] 5. for i ← 1 to n - 1 do // ‘for’ loop from the index 1 to n-1 (inclusive). 6. sum ← sum + A[i] 7. avg ← sum / n 8. return avg |

The for loop used in our textbook differ from common programming languages. For example, the C ++ language uses a for loop with a comparison operation (<=) and an index increment operation (++) as shown below

for (int i = 1; i <= n - 1; i++)

But, the for loop in the pseudocode in our textbook describes the index and its range as shown below.

for i ← 1 to n - 1 do

The following example presents pseudocode with three operations in a while loop.

Example 4: Search a value in an array with *n* elements

|  |
| --- |
| 1. Algorithm *SequentialSearch*(A[0..n − 1], K) 2. // Searches for a given value in a given array by sequential search 3. // Input: An array A[0..n − 1] and a search key K 4. // Output: The index of the first element in A that matches K or −1 if there are no 5. // matching elements in the array. 6. i ←0 7. while ((i < n) AND (A[i] K)) do // ‘while’ loop with ‘<’, ‘AND’, and ‘‘ operators 8. i ←i + 1 9. if ( i < n ) 10. return i 11. else 12. return −1 |

This is our last example to show a nested loop in the pseudocode for a sorting algorithm called selection sort.

Example 5: Selection sort for an array with *n* elements

|  |
| --- |
| 1. Algorithm *SelectionSort*(A[0..n − 1]) 2. // Input: An array A[0..n − 1] 3. // Output: Rearrange the input array A in the ascending order.. 5. i ←0 6. while ( i < n ) do 7. min ← i 8. j ← i + 1 9. while ( j < n ) do 10. if ( A[j] < A[min] ) 11. min ← j 12. j ← j + 1 13. // swap A[i] and A[min] 14. tmp ← A[i] 15. A[i] ← A[min] 16. A[min] ← tmp 17. i ←i + 1 19. return |