Chapter 2: Arrays and Structures

2.1 Arrays

2.1.1 The Abstract Data Type

- Although an array is usually implemented as a consecutive set of memory locations, that is not always the case.
- Intuitively an array is a set of pairs, $\langle$index, value$\rangle$, such that each index that is defined has a value associated with it. In mathematical terms, we call this a correspondence or a mapping.
- Aside from creating a new array, most languages provide only two standard operations for arrays, one that retrieves a value, and a second that stores a value.

ADT Array is

objects: A set of pairs $\langle$index, value$\rangle$ where for each value of index there is a value from the set item. Index is a finite ordered set of one or more dimensions, for example, $\{0, \ldots, n-1\}$ for one dimension, $\{(0,0), (0,1), (0,2), (1,0), (1,1), (1,2), (2,0), (2,1), (2,2)\}$ for two dimensions, etc.

functions: for all $A \in$ Array, $i \in$ index, $x \in$ item, j, size $\in$ integer.

Array Create(j, list) ::= return an array of j dimensions where list is a j tuple whose ith element is the size of the ith dimension. Items are undefined

Item Retrieve(A, i) ::= if (i $\in$ index) return the item associated with index value i in array A else return error.

Array Store(A, i, x) ::= if (i in index)
return an array that is identical to array A except the new pair $\langle i, x \rangle$ has been inserted else return error.

end Array