Array ADT

So far we have looked at Integer and String ADTs.

ADT Array:

Elements: A component data type is defined and all elements are of that type (homogeneous).

Structure: A linear index type is specified and a 1-1 correspondence exists between the index type and component type.
Array ADT Continued

Domain: All possible index values with all combinations of associated component values.

Operations:
1) Copy array element value (e.g. value = a[i])
   results: The $i^{th}$ component of a is copied into value
Array ADT Continued

2) Update array element (e.g. \( a[i] = \text{value} \))
   results: The \( i^{\text{th}} \) component of \( a \) is assigned value

3) Array copy (e.g. \( a = b \))
   results: All elements from \( b \) are copied into their respective positions in \( a \)
Multi-dimensional Arrays

- Obviously, we can extend the array ADT to include multidimensional arrays. The only real change is the structure which becomes something like:

  component-type array[index1, index2]
Array Mapping Function (AMF)

• The only real challenge in implementing arrays is how to map a multi-dimensional array into linear space.

• Two-dimensional array AMF by rows:

Consider: int a[10][5];

\[
a[i][j] = \text{base}(a) + (i * 5 + j) * \text{sizeof (int)};
\]
More AMF

- What is the AMF for each of the following assuming a row-major mapping?

1) double a[100];

2) float b[5][10][15];