Shana L Moore

PID: 011780377

HW 02 – Array Mapping Functions Homework

**KEY:**

|  |
| --- |
| L1 = lower bound of row  U1 = upper bound or row  L2 = lower bound of column  U2 = upper bound of column  i1 = row of element to map  i2 = column of element to map  Base = base address of each array  Esize = size of each array element |

**Major-column Formula:**

|  |
| --- |
| Base + [(i1-L1) \* (U2 – L2 + 1) + (i2 – L2)] \* esize |

1. Assume each element of an array *list*, stored in row-major order, occupies 4 bytes of storage, base address is 100…
   1. list[10] = 136
      1. 100 + [(10 - 1) \* (0 – 0 + 1) + (0 – 0)] \* 4
   2. list[5,10] = 560
      1. 100 + [(5 - 0) \* (20 - 0 + 1) + (10 - 0)] \* 4
      2. 100 + 115 \* 4
   3. list[3,8] = 584
      1. 100 + [(3 - -5) \* (12 - -1 + 1) + (8 - -1)] \* 4
      2. 100 + [(8)\*(14)+(9)] \* 4
      3. 100 + 121 \* 4
2. Develop a formula to access an array element if the two-dimensional array is stored in column-major order.
3. Number of columns we skip to get to the correct column in which our element if to be found:
   1. i2 – L2
4. Number of rows in each column?
   1. U1 – L1 + 1
5. Number of bytes each column occupy in memory:
   1. (U1 – L1 + 1) \* esize
6. Number of bytes we skip in memory to get to the first memeory location of the column in which our element is to be found:
   1. (i2 – L2) \* (U1 – L1 + 1) \* esize
7. Starting location in memory for the column in which our element is to be found:
   1. Base + (i2 – L2) \* (U1 – L1 + 1) \* esize
8. Number of elements we skip in the correct column to get to the element whose memory location we’re trying to locate?
   1. i1 – L1
9. How many bytes do these elements occupy in memory?
   1. (i1 – L1) \* esize
10. What is the memory location our element can be found at?
    1. Base + (i2 – L2) \* (U1 – L1 + 1) \* esize + (i1 – L1) \* esize
11. Simplifying the expression produces the following mapping function:
    1. Base + [(i1 + L1) + (U1 – L1 + 1) \* (i2 – L2)] \* esize