CECS 524

Unit 6

Assignments

Read Chapter 6 in the textbook.

Program due Oct 11, 5 pm, in the Dropbox for Unit 6.

Using the programming language of your choice, implement the expression for computing array addresses for 2-dimensional arrays of any element size and any arbitrary lower and upper bounds. This is the expression

location(a[i, j]) = address of a[row\_lb, col\_lb] - (((row\_lb \* n) + col\_lb) \* element\_size) + (((i \* n) + j) \* element\_size)

where the first two terms are the constant part and the last is the variable part. n is the size of a row in the 2-D array.

Create a **subroutine** named **calcAddress** that accepts as parameters:

* base //address of a[row\_lb, col\_ob]
* rowlb //row lower bound
* collb //column lower bound
* rowub //row upper bound
* colub //column upper bound
* elementsize //size of the element

For this call:

**calcAddress(1200, 0, 0, 2, 2, 1);**

**base, rowlb, collb, rowub, colub, elementsize**

the output is

**For array a[0:2,0:2] with element size 1**

**a[0,0] address = 1200**

**a[0,1] address = 1201**

**a[0,2] address = 1202**

**a[1,0] address = 1203**

**a[1,1] address = 1204**

**a[1,2] address = 1205**

**a[2,0] address = 1206**

**a[2,1] address = 1207**

**a[2,2] address = 1208**

For this call:

**calcAddress(100, 1,1,2,2,2);**

the output is

**For array a[1:2,1:2] with element size 2**

**a[1,1] address = 100**

**a[1,2] address = 102**

**a[2,1] address = 104**

**a[2,2] address = 106**

For this call:

**calcAddress(100, 2, 3, 4,5,4);**

the output is

**For array a[2:4,3:5] with element size 4**

**a[2,3] address = 100**

**a[2,4] address = 104**

**a[2,5] address = 108**

**a[3,3] address = 112**

**a[3,4] address = 116**

**a[3,5] address = 120**

**a[4,3] address = 124**

**a[4,4] address = 128**

**a[4,5] address = 132**

For this call:

**calcAddress(100, -1, -1, 1, 2, 8);**

the output is

**For array a[-1:1,-1:2] with element size 8**

**a[-1,-1] address = 100**

**a[-1,0] address = 108**

**a[-1,1] address = 116**

**a[-1,2] address = 124**

**a[0,-1] address = 132**

**a[0,0] address = 140**

**a[0,1] address = 148**

**a[0,2] address = 156**

**a[1,-1] address = 164**

**a[1,0] address = 172**

**a[1,1] address = 180**

**a[1,2] address = 188**

**In your driver make all calls in sucsession with continuous output.**

**Like this:**

**myMain() {**

**calcAddress(1200, 0, 0, 2, 2, 1);**

**calcAddress(100, 1,1,2,2,2);**

**calcAddress(100, 2, 3, 4,5,4);**

**calcAddress(100, -1, -1, 1, 2, 8);**

**}**

**YOU MUST USE THE FORMULA. Any dope can just add the element size to the start address in a loop.**