Faculty of Engineering & Technology

University of Lucknow
Lucknow

Data Structure Primer Using C

Subject Code: CS-301

Topic: Memory Address Calculation in an Array

Dr. Himanshu Pandey

Assistant Professor (CSE)

FoET- University of Lucknow.

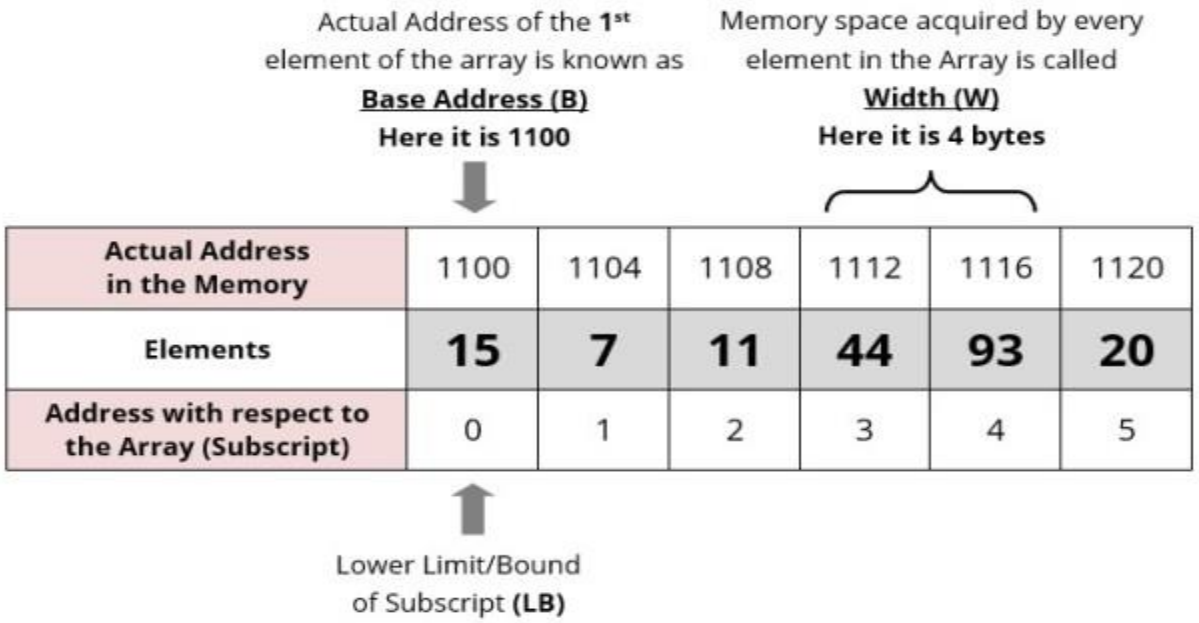
REFERENCES :

1. Aaron M. Tenenbaum, YedidyahLangsam and Moshe J. Augenstein "Data Structures Using C and C++", PHI Learning Private Limited, Delhi India.
2. Horowitz and Sahani, "Fundamentals of Data Structures", Galgotia Publications Pvt Ltd Delhi India.
3. A.K. Sharma ,Data Structure Using C, Pearson Education India.
4. Rajesh K. Shukla, "Data Structure Using C and C++" Wiley Dreamtech Publication.
5. Lipschutz, "Data Structures" Schaum's Outline Series, Tata Mcgraw-hill Education (India) Pvt. Ltd .
6. Michael T. Goodrich, Roberto Tamassia, David M. Mount "Data Structures and Algorithms in C++", Wiley India.
7. P.S. Deshpandey, "C and Datastructure", Wiley Dreamtech Publication.
8. R. Kruse etal, "Data Structures and Program Design in C", Pearson Education.
9. Berztiss, A.T.: Data structures, Theory and Practice :, Academic Press.

Disclaimer: The e-content is exclusively meant for academic purposes and for enhancing teaching and learning. Any other use for economic/commercial purpose is strictly prohibited. The users of the content shall not distribute, disseminate or share it with anyone else and its use is restricted to advancement of individual knowledge. The information provided in this e-content is developed from authentic references, to the best of my knowledge.

Memory Address Calculation in an Array

Address Calculation in single (one) Dimension Array:



Array of an element of an array say “A[I]” is calculated using the following formula:

Address of A [I] = B + W * (I – LB)

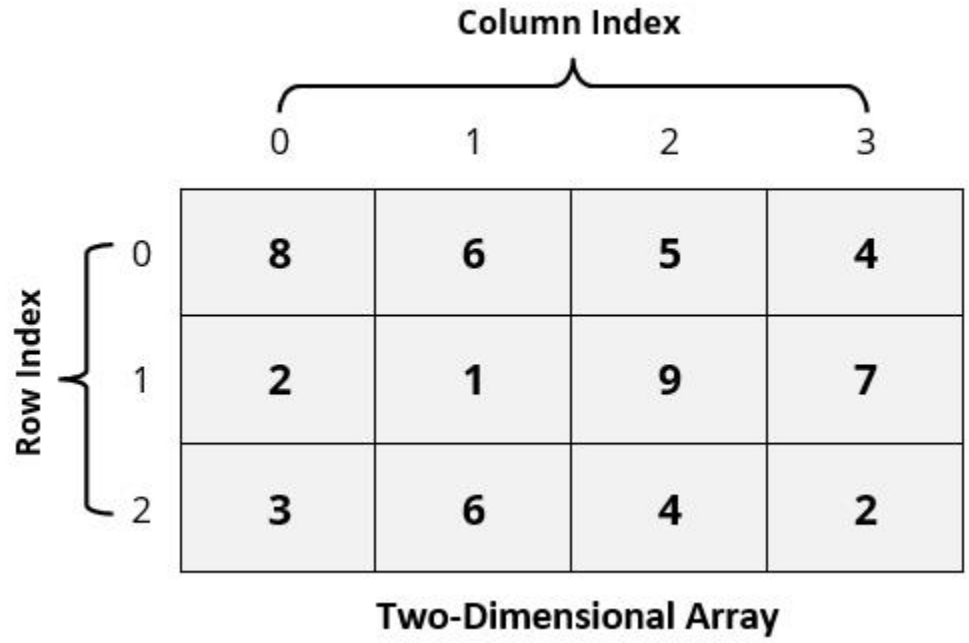
Where,
B = Base address
W = Storage Size of one element stored in the array (in byte)
I = Subscript of element whose address is to be found
LB = Lower limit / Lower Bound of subscript, if not specified assume 0 (zero)

Example:
Given the base address of an array **B[1300…..1900]** as 1020 and size of each element is 2 bytes in the memory. Find the address of **B[1700]**.
Solution:
The given values are: B = 1020, LB = 1300, W = 2, I = 1700

Address of A [I] = B + W * (I – LB)
= 1020 + 2 * (1700 – 1300)
= 1020 + 2 * 400
= 1020 + 800
= 1820 **[Ans]**

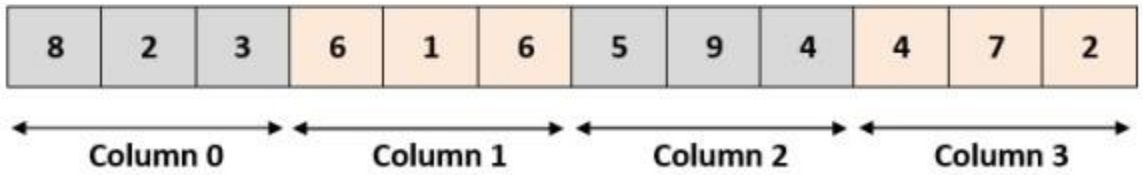
Address Calculation in Double (Two) Dimensional Array:

While storing the elements of a 2-D array in memory, these are allocated contiguous memory locations. Therefore, a 2-D array must be linearized so as to enable their storage. There are two alternatives to achieve linearization: Row-Major and Column-Major.



Row-Major (Row Wise Arrangement)

Column-Major (Column Wise Arrangement)



Address of an element of any array say "**A[I][J]**" is calculated in two forms as given:
(1) Row Major System (2) Column Major System

Row Major System:
The address of a location in Row Major System is calculated using the following formula:

Address of A [I][J] = B + W * [N * (I – Lr) + (J – Lc)]

Column Major System:

The address of a location in Column Major System is calculated using the following formula:

Address of A [I][J] Column Major Wise = B + W * [(I – Lr) + M * (J – Lc)]

Where,
B = Base address
I = Row subscript of element whose address is to be found
J = Column subscript of element whose address is to be found
W = Storage Size of one element stored in the array (in byte)
Lr = Lower limit of row/start row index of matrix, if not given assume 0 (zero)
Lc = Lower limit of column/start column index of matrix, if not given assume 0 (zero)
M = Number of row of the given matrix
N = Number of column of the given matrix

Important: Usually number of rows and columns of a matrix are given (like A[20][30] or A[40][60]) but if it is given as **A[Lr- – – – – Ur, Lc- – – – – Uc]**. In this case number of rows and columns are calculated using the following methods:

Number of rows (**M**) will be calculated as = **(Ur – Lr) + 1**
Number of columns (**N**) will be calculated as = **(Uc – Lc) + 1**
And rest of the process will remain same as per requirement (Row Major Wise or Column Major Wise).

Examples:

Q 1. An array X [-15…….10, 15……40] requires one byte of storage. If beginning location is 1500 determine the location of X [15][20].

Solution:
As you see here the number of rows and columns are not given in the question. So they are calculated as:

Number or rows say **M = (Ur – Lr) + 1** = [10 – (- 15)] +1 = 26
Number or columns say **N = (Uc – Lc) + 1** = [40 – 15)] +1 = 26

(i) Column Major Wise Calculation of above equation

The given values are: B = 1500, W = 1 byte, I = 15, J = 20, Lr = -15, Lc = 15, M = 26

Address of A [I][J] = B + W * [(I – Lr) + M * (J – Lc)]

= 1500 + 1 * [(15 – (-15)) + 26 * (20 – 15)] = 1500 + 1 * [30 + 26 * 5] = 1500 + 1 * [160] = 1660 **[Ans]**

(ii) Row Major Wise Calculation of above equation

The given values are: B = 1500, W = 1 byte, I = 15, J = 20, Lr = -15, Lc = 15, N = 26

Address of A [I][J] = B + W * [N * (I – Lr) + (J – Lc)]

= 1500 + 1* [26 * (15 – (-15))) + (20 – 15)] = 1500 + 1 * [26 * 30 + 5] = 1500 + 1 * [780 + 5] = 1500 + 785
= 2285 **[Ans]**