"...It is unworthy of excellent men to loose hours like slaves in the labour of calculation which could safely be regulated to anyone else if machines were used.."

–Leibnitz in the beginning of seventeenth Century.

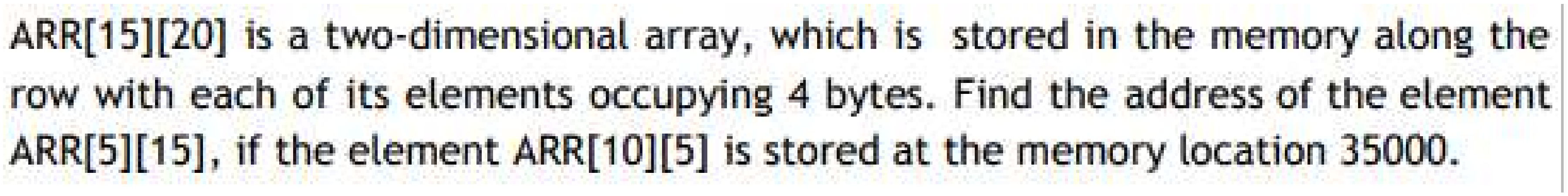
Programming

- Identifier
    - Naming Rules
- Token
- Macro
- Arrays
- Pointers

Finding memory address

T[20][50] is a two dimensional array, which is stored in the memory along the row with each of its element occupying 4 bytes, find the address of the element T[15][5], if the element T[10][8] is stored at the memory location 52000.

DIY



ARR[15][20] is a two-dimensional array, which is stored in the memory along the row with each of its elements occupying 4 bytes. Find the address of the element ARR[5][15], if the element ARR[10][5] is stored at the memory location 35000.

ARR[15][20] is a two-dimensional array, which is stored in the memory along the row with each of its elements occupying 4 bytes. Find the address of the element ARR[5][15], if the element ARR[10][5] is stored at the memory location 35000.

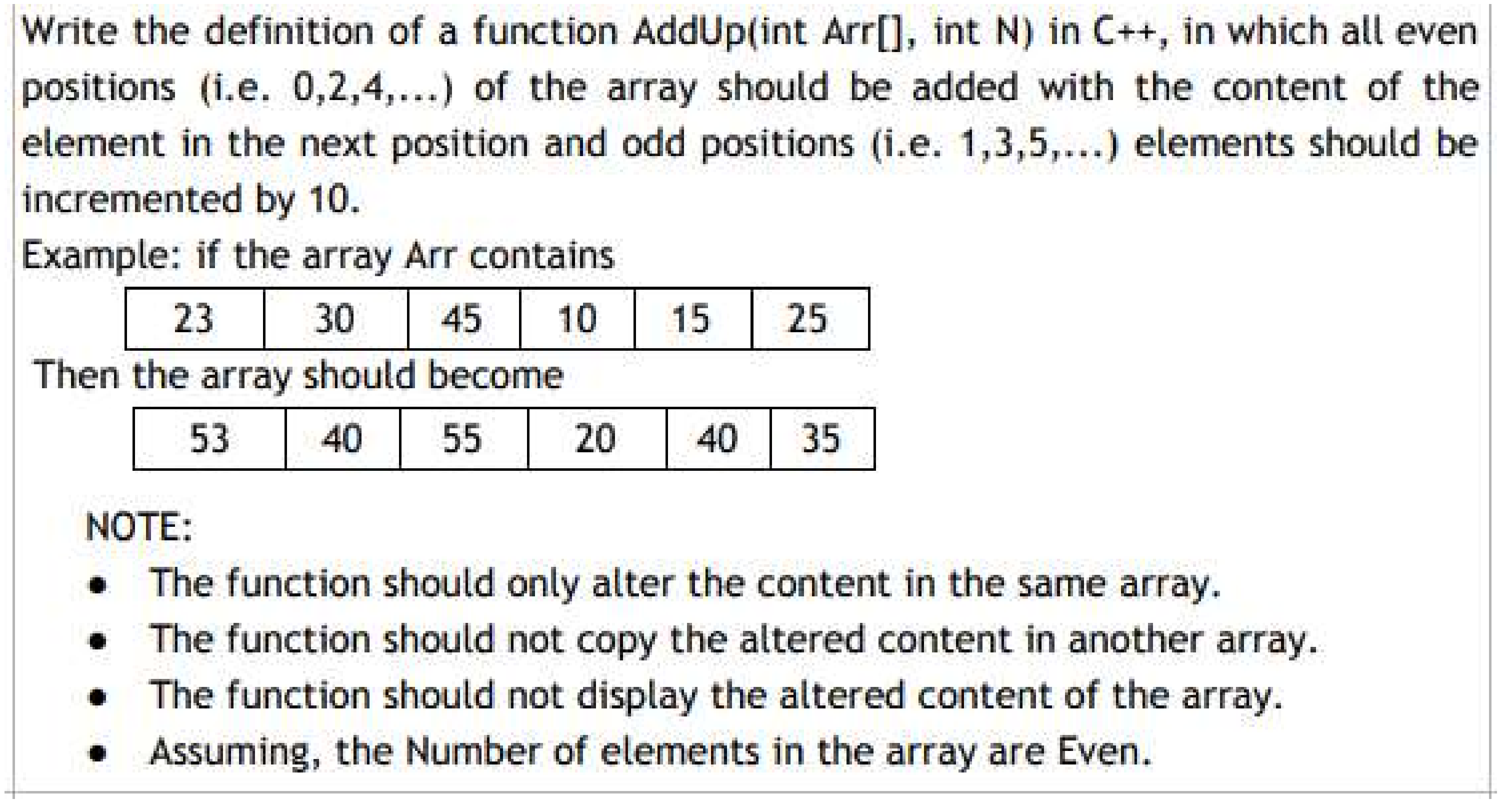
```
ROW MAJOR:
Loc(ARR[I][J]) =BaseAddress + W [( I - LBR)*C + (J - LBC)]

(where  W=size of each element = 4 bytes,  R=Number of
Rows=15, C=Number of Columns=20 )

Assuming LBR = LBC = 0
```

```
LOC(ARR[10][5])
35000          = BaseAddress + W(I*C + J)
35000          = BaseAddress + 4(10*20 + 5)
35000          = BaseAddress + 4(205)
35000          = BaseAddress + 820
BaseAddress    = 35000 - 820
               = 34180

LOC(ARR[5][15])= BaseAddress + W(I*C + J)
               = 34180       + 4(5*20 + 15)
               = 34180       + 4(100 + 15)
               = 34180       + 4 x 115
               = 34180       + 460
               = 34640
OR
Loc(ARR[I][J]) = Ref. Address + W (( I - LR)*C + (J - LC))
(where
W=size of each element = 4 bytes,
R=Number of Rows =15, C=Number of Columns=20
Reference Address= Address of given cell ARR[10][5]=35000
LR = Row value of given cell = 10
LC = Column value of given cell = 5

LOC(ARR[5][15]) = LOC(ARR[10][5]) + 4((5-10)*20 + (15-5))
LOC(ARR[5][15]) = 35000 + 4(-100 + 10)
                = 35000 + 4[-90]
                = 35000 -360
                = 34640
```

Functions

- Prototype
- Declaration
- Pass by reference
- Pass by value
- Arrays and functions
    - Most significant dimension

DIY



Write the definition of a function AddUp(int Arr[], int N) in C++, in which all even positions (i.e. 0,2,4,...) of the array should be added with the content of the element in the next position and odd positions (i.e. 1,3,5,...) elements should be incremented by 10.

Example: if the array Arr contains

23	30	45	10	15	25

Then the array should become

53	40	55	20	40	35

NOTE:

- The function should only alter the content in the same array.
- The function should not copy the altered content in another array.
- The function should not display the altered content of the array.
- Assuming, the Number of elements in the array are Even.

Object Oriented Programming Style

- Struct
- Classes
  - Access Specifier
- Default Methods
  - Ctor, Dtor, Assignment, Copy Ctor
- Polymorphism
- Inheritance
  - Order of Creation/Destruction

www.codingblocks.com
www.deepakaggarwal.me