LAB 3

ARRAY

Department of Electronics Engineering
National Yang Ming Chiao Tung University

Outline

- Array introduction
- Lab4 exercise

Array Declaration

- type array_name[size]
  - int score[5]

    5 variables of int named score, score[0]~score[4]

  - size must be an expression evaluating to integral CONSTANT

    ```
    int a_arr[10+20];// ok
    const int K = 100;
    int b_arr[K];// ok, b_arr[100];
    int n = 100;
    int c_arr[n];// error, n is not a constant
    ```

Array Initialization (1/3)

- An array can be initialized

```
int score[3] = {2, 12, 1};
```

–which is equivalent to following:

```
int score[3];
score[0] = 2;
score[1] = 12;
score[2] = 1;
```

Array Initialization (2/3)

- If fewer values than size
  - –fills from the beginning
  - –fills the remaining elements with 0 of array base type
  - –e.g., int a[6] = {10, 11, 12, 13}; // a[4] = 0, a[5] = 0

- If more values than size
  - –compilation error
  - –e.g., double b[3] = { 6.0, 6.5, 7.0, 8.0}; // error

Array Initialization (3/3)

- If size is unspecified

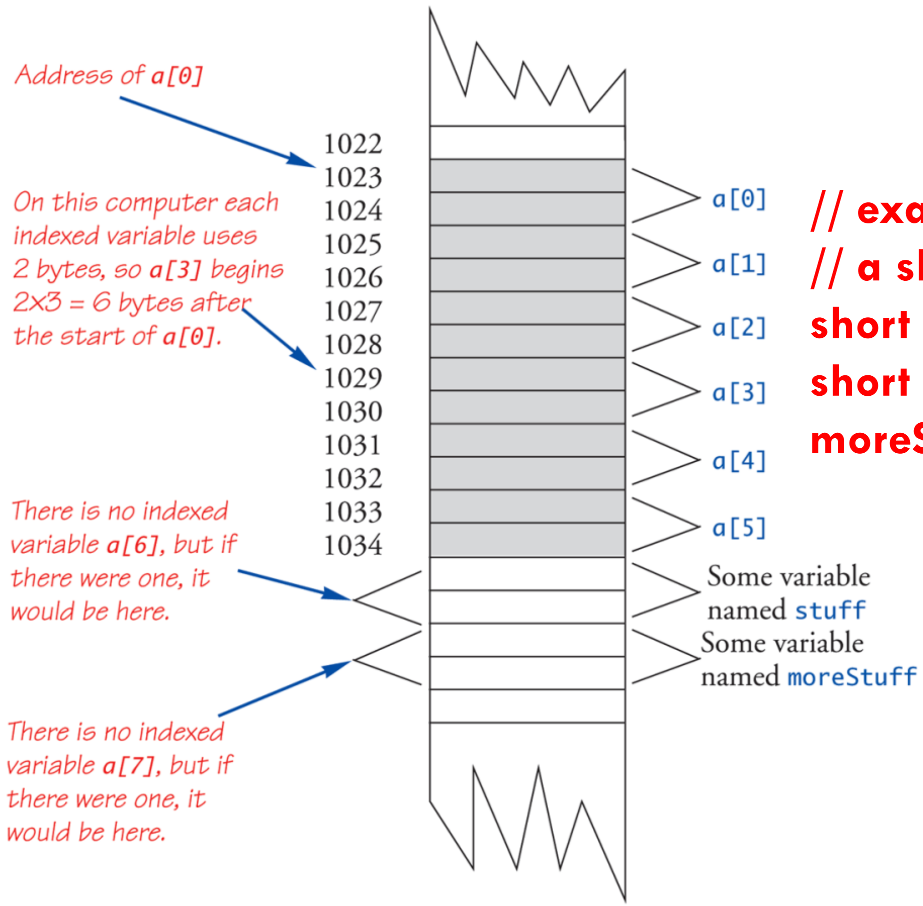
  –size is automatically determined based on number of initialization values

  –example:

  ```
  int b[]= {5, 12, 11}; // equivalent =>int b[3] = {5, 12, 11};
  int c[];// error, unknown size
  ```

Array in Memory

Display 5.2 **An Array in Memory**

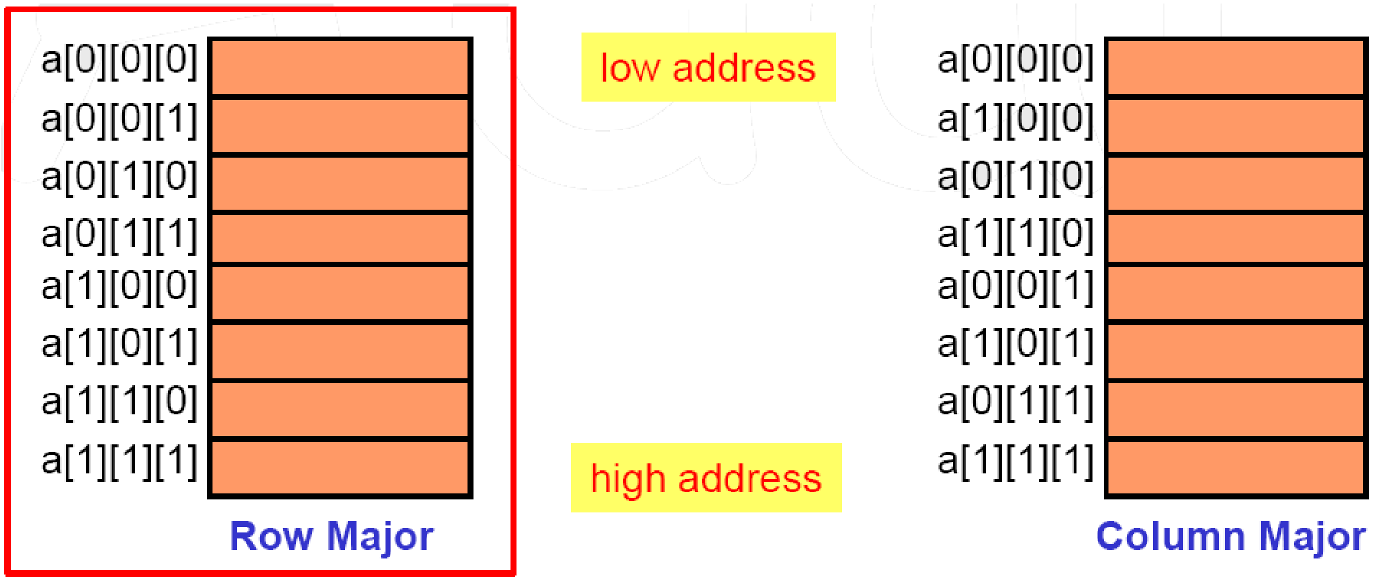


// example
// a short int takes 2bytes
short a[6]; // start @ 1023
short stuff; // start @ 1035
moreStuff; // start @ 1037

address of a[4]
= 1023+ 4* 2=1031

Multidimensional Array(1/2)

- How to put a multidimensional array into linearly addressed memory?
  - –row major (used in C/C++, …)
  - –column major (used in age-old FORTRAN, …)
- An array, a[2][2][2], has 8 elements
  - –a[0][0][0], a[0][0][1], a[0][1][0], a[0][1][1],a[1][0][0], a[1][0][1], a[1][1][0], a[1][1][1]



Multidimensional Array(2/2)

- For an array a[u1][u2]…[un] starting at the address A,what is the address of a[i1][i2]…[in] ?

  int a[6][7][8]; // assume starting from address 1000, sizeof(int) = 4

  a[1][2][3] = 10; // what is the address of a[1][2][3]?

  address = 1000+ ((1* 7* 8) + (2* 8) + 3) * 4= 1300

  address= A+ i1u2u3…un

  + i2u3u4…un

  ...

  + i(n-1)un

  + in

Array in Function (1/2)

- In function declaration and definition

  void f1(char arr[ ]); // just use empty brackets

  void f2(char arr[10]); // **still ok, compiler simply ignores what's inside [ ]**

- In function call

  –use array name as actual argument

  void f() {

  char table[10];

  f1(table);// ok

  f2(table);// **still ok**

  }

- Need another parameter for array sizeif required

  void f3(char arr[ ], int size);

Array in Function (2/2)

- In function declaration and definition

  –MUST specify sizes for ALL dimensions except for the first one
- void f1(char arr[ ][6][7][8]); // a 4-dimentional array

  void f2(char arr[9][6][7][8]); // **still ok, compiler ignores what's inside [ ]**
- In function call

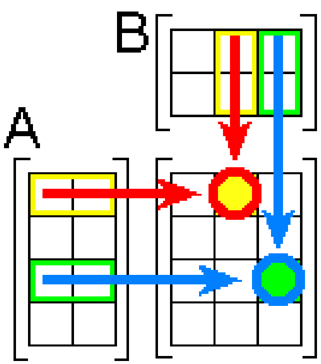
  –use array name as actual argument; sameway as for 1D array

```
void f() {
char table1[5][6][7][8];
char table2[5][5][7][8];
f1(table1);// ok
f2(table1);// still ok
f1(table2);// compilation error, array size not matched!
}
```

Exercise (1/2)

- Exercise 1
  - input 3x3 Matrix A
  - operation =1 ➔ output A^T & continue
  - operation =2 ➔ input 3x2 Matrix B & multiply two Matrix

由定義直接計算



左邊的圖表示出要如何計算AB的(1,2)和(3,3)元素，當A是個4×2矩陣和B是個2×3矩陣時

$$(AB)_{1,2} = \sum_{r=1}^{2} a_{1,r} b_{r,2} = a_{1,1} b_{1,2} + a_{1,2} b_{2,2}$$

$$(AB)_{3,3} = \sum_{r=1}^{2} a_{3,r} b_{r,3} = a_{3,1} b_{1,3} + a_{3,2} b_{2,3}$$

係數－向量方法

Exercise (2/2)

```
Input Matrix A:
1 2 3
4 5 6
7 8 9
Matrix A:
   1   2   3
   4   5   6
   7   8   9
Operation: 2
Input Matrix B:
1 2
3 4
5 6
Matrix B:
   1   2
   3   4
   5   6
Matrix A X Matrix B:
  22  28
  49  64
  76 100
```

```
Input Matrix A:
1 2 3
4 5 6
7 8 9
Matrix A:
   1   2   3
   4   5   6
   7   8   9
Operation: 1
Matrix A:
   1   4   7
   2   5   8
   3   6   9
Operation: 2
Input Matrix B:
1 2
3 4
5 6
Matrix B:
   1   2
   3   4
   5   6
Matrix A X Matrix B:
  48  60
  57  72
  66  84
```