18ES611 Embedded System Programming

Sarath tv

Data Types

Basic – integer, float, char, double

Derived – Array, structures ,unions

Array

An array is collection of items stored at continuous memory locations. The idea is to declare multiple items of same type together.

In C, declare an array by specifying its type and size or by initializing it or by both.

```
// Array declaration by specifying size

int arr[10];

// Array declaration by initializing elements

int arr[] = {10, 20, 30, 40}

// Array declaration by specifying size and initializing  elements

int arr[6] = {10, 20, 30, 40}
```

Accessing array elements

Through index

Through pointer (will be dealt later).

Array elements are accessed by using an integer index. Array index starts with 0 and goes till size of array minus 1.

Multidimensional Arrays

Multi-dimensional arrays are declared by **providing more** than **one set** of **square** [ ] brackets after the **variable name** in the declaration statement.

For two dimensional arrays, the **first dimension** is commonly considered to be the **number** of **rows**, and the **second dimension** the number of columns.

```
int a[ 2 ][ 3 ] = { { 5, 6, 7 }, { 10, 20, 30 } };
```

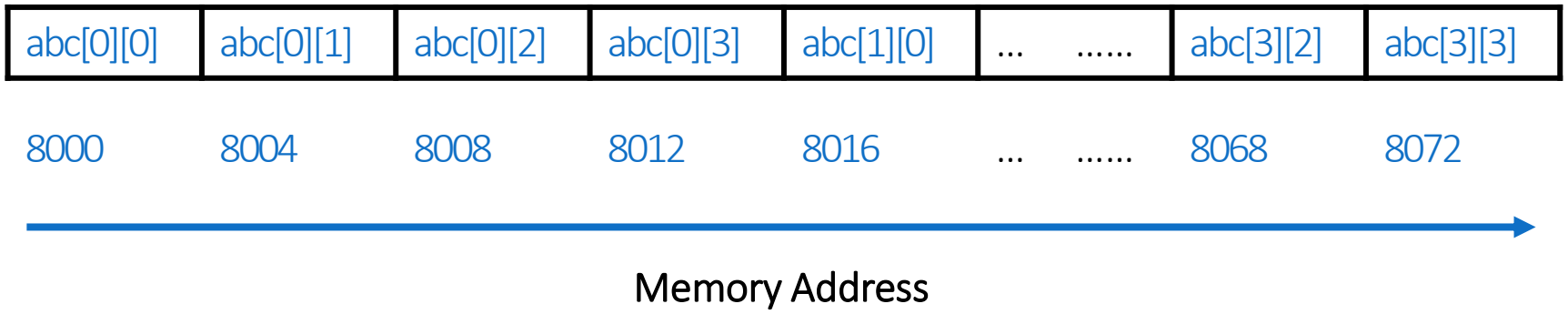
	Column 0	Column 1	Column 2	Column 3
Row 0	a[0][0]	a[0][1]	a[0][2]	a[0][3]
Row 1	a[1][0]	a[1][1]	a[1][2]	a[1][3]
Row 2	a[2][0]	a[2][1]	a[2][2]	a[2][3]

Two dimensional Array

abc[0][0]	abc[0][1]	abc[0][2]	abc[0][3]
abc[1][0]	abc[1][1]	abc[1][2]	abc[1][3]
abc[2][0]	abc[2][1]	abc[2][2]	abc[2][3]
abc[3][0]	abc[3][1]	abc[3][2]	abc[3][3]

The array abc is a 2 Dimensional array having 4 rows 4 columns
Indexing starts with zero.
A particular element can be accessed by using the index for row and column.

Actual representation of this array in memory



Normal Representation of addresses – Hex., Here in decimal.

Elements are stored in contiguous locations.

Address difference between the element = Size of one element.

- A **2D array** is **stored** in the computer's memory **one row following another**.
- The **address of the first byte of memory** is considered as the memory **location** of the **entire 2D array**.
- **Knowing** the **address** of the **first byte** of memory, the **compiler** can easily **compute** to find the **memory** location of any **other elements** in the 2D array provided the **number** of **columns** in the **array** is known.
- If each data value of the array requires **B bytes** of memory, and if the array has **C columns**, then the memory location of an element such as score[m][n] is

(m*c+n)*B

from the address of the first byte.

- Note that **to find the memory location of any element**, **there is no need to know the total number of rows in the array,** i.e. **the size of the first dimension**. Of course the size of the first dimension is needed to prevent reading or storing data that is out of bounds.
- Again one should not think of a 2D array as just an array with two indexes. You should think of it as an **array of arrays.**

Higher dimensional arrays should be similarly **interpreted**. For example a **3D array should be thought of as an array of arrays of arrays**. To find the **memory location of any element in the array relative to the address of the first byte**, the **sizes of all dimensions other than the first must be known.**

Knowledge of how multidimensional arrays are stored in memory helps one understand how they can be initialized, and how they can be passed as function arguments.

Stack Implementation using array

Functions for stack operations!!!

THANK YOU!!!!!