EASTERN WASHINGTON UNIVERSITY
start something big

C Array

Computer Science Department
Eastern Washington University
Yun Tian (Tony) Ph.D.

Recall Last Class

- C functions
  - Why use functions?
  - How to write(define) functions?
  - How to use(call) a function?
  - Good practices

Today

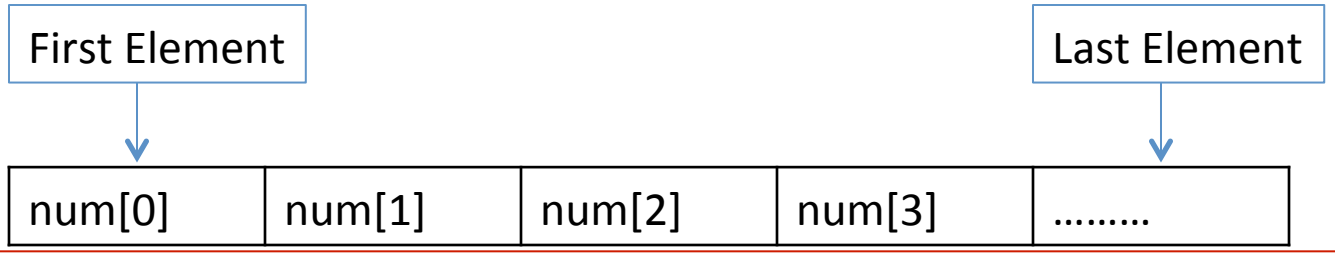
- C array

Concept of Array

- Array in C is a data structure.
  - An array is used to store a collection of data.
    - Arrays occupy space in memory.
  - It stores a **fixed-size** sequential collection of elements of the same type.
  - More useful to think of an array as a collection of variables of the same type.
    - Because we can assign/re-assign different values to each array element.

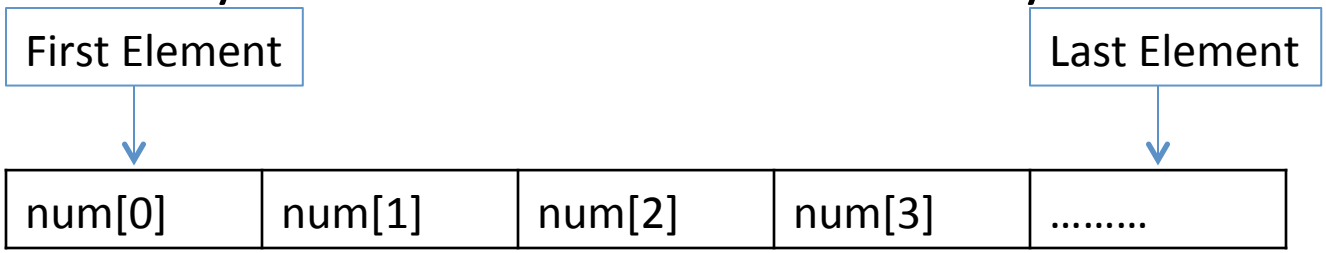
Concept of Array

- The entire array consist of **contiguous** memory locations.
  - Adjacent elements in array are stored in contiguous memory locations in memory.
  - The lowest address corresponds to the first element and the highest address to the last element.



Concept of Array

- The entire array consist of contiguous memory locations.
  - Interestingly, in C the **array name** is used as the address of the first element,
    - called array base address
    - &(num[0]) and **num** represent the same value, the array base address. **num** is the array name here.



Define Arrays

- To define an array in C, we specifies the type of the elements and the number of elements required by an array as follows:

  type arrayName [constantArraySize];

  Note here: the array size between bracket should be a integer constant greater than 0;

  It may work when put a variable there in the bracket on some platforms. But that is **discouraged**.

- unlike Java:

  type [] arrayName; → does NOT work in C

Define Arrays

- double balance[10];
  - Now ***balance*** is available array which is sufficient to hold up to 10 double numbers.
  - The maximum number of elements in balance is fixed,
    - We have to know the maximum capacity before **compile**.
    - If more than 10 values in a file, then balance array cannot hold them all.
    - If only 5 values in a file, then half of array memory is wasted.

Memory Space for Arrays

- Define means create the array entirely, including allocating memory space.
- If you do **int arr[10];** inside a function or in main(), the memory space for 10 integers is allocated **automatically** when the function is invoked.
  - You **do not** worry about the memory allocation in this case.
- This memory is deallocated **automatically** when the function (in which the array is defined) returns.
  - Therefore, we **cannot return** this type of array from inside a function.

Define Arrays

- int arr[10]; we call this type of array, **'static'** array for two reasons,
  - Size of array has to be a constant, and has to be known before compile.
  - Memory for array is allocated and deallocated automatically,
    - OS takes care of that.
- On the contrary, we have **dynamic** array.

Define Arrays

- On the contrary, we have **dynamic** array.
  - Pointer in C corresponds to a **dynamic** array.
  - Programmer requests memory manually using function malloc() or calloc().
    - Also Programmer is responsible to manually deallocate the memory that is returned by malloc() or calloc().
  - Size of the dynamic array can be determined at run time.
- We have more information when study pointers. Today we focus on C 'static' array.

Initialize Array

- Today we focus on C 'static' array.
- double balance[5] = {1000.0, 2.0, 3.4, 17.0, 50.0};
  - We cannot put more than 5 values between {}.
  - This is compile error!
- double balance[] = {1000.0, 2.0, 3.4, 17.0, 50.0};
  - If you omit the size of the array, an array just big enough to hold the initialization is created.

Access Array Elements

- Same usage as in Java

  double salary = balance[9];

  – This access the tenth element in balance array.

  – Array index starts at 0 as we learned in Java.

Simple Demo

```
#include <stdio.h>
#define SIZE 10

int main ()
{
        int n[SIZE]; /* n is an array of 10 integers */
        int i, j;   /* initialize elements of array n to 0 */
        for ( i = 0; i < SIZE; i++ )
        {
                n[ i ] = i + 100; /* set element at location i to i + 100 */
        } /* output each array element's value */
        for (j = 0; j < SIZE; j++ )
        {
                printf("Element[%d] = %d\n", j, n[j] );
        }
        printf("value for n[11] is %d\n", n[11]);
        return 0;
}
//what if we access n[11] or n[12] in the program above.
```

Simple Demo

```
#include <stdio.h>
int update(int a[], int size, int except)
{
        int i;
        int count = 0;
        for(i = 0; i < size; i ++)
        {
                if( a[i] != except )
                {
                        a[i] = a[i] + 10;
                        count ++;
                }
        }
        return count;
}

int main()
{
    int grades[5] = {  80, 80, 90, 60, 40 };
    int ret = update( grades, 5, 90 );  // Array elements in grades can be modified in function update(). SAME as JAVA.
    //output array
}
```

Negative Example

```
int [] loadData() {
        int data[100]; //'static array', automatic memory allocation
        int i;
        for( i = 0; i < 100; i ++)
                data[i] = i * 2;
        return data;
}
//This will not work in C.
//From inside a function, you cannot return a 'static' array.
```

2D Array

- int arr2d[100][50];
  - 'static' 2D array, system reserves 100 rows of integers, each rows consist of 50 integer items.
    - Same as 1D 'static' array with regard to memory allocation and deallocation.
  - Use **arr2d[10][20]** to access the element at row 10 and column 20.
  - 100 * 50 integer items are contiguously stored in memory.
    - Row-major storage in memory.

Summary

- C Arrays
- Two types
  - 'static' Array
  - Dynamic Array
  - Their differences
- How to use 'static' array in C program

Next Class

- C pointer
- Ready to take the challenge