Lecture 12.1

Topic

1. Arrays and Functions – Continued

1. Arrays and Functions – Continued

Recall that as an array is passed to a function then the array address is automatically sent to the function; and with this address, the function will be able to access the array elements.

Also it was mentioned that the array size must be explicitly sent to the function as argument unless otherwise implied in the logic and function interface.

And again,

> **When sending array arguments to functions, it is important to remember that the array itself (that means the values of array members/elements) will NOT be copied over to the function.**

The function will work through the array argument name in order to have access to the values as well as addresses of every member.

Because of having these addresses, inside the function, one may also be able to change the values of the array elements.

1.1 Function Prototypes

A simple function may be given as follows,

```
void printArrayVerionA( int arg[], int size );
```

In this prototype, an argument name is used with the square brackets to indicate an array and the size is followed.

And the above prototype can also be written as

```
void printArrayVerionA( int [], int );
```

So for prototypes involved an array, there is no specific name of any array being mentioned or required.

1.2 Examples

Example 1 – **`for()`** Loop

```
void printArray( int arg[], int size ) {
  int i;

  for ( i = 0; i < size; i++ ) {
    printf( "\narray[ %d ] : %d", i, arg[ i ] );
  }

  return;
}
```

Example 2 – **`while()`** Loop

```
int sumArrayWhile( int arg[], int size ) {
  int total;
  int i;

  i = 0;
  total = 0;
  while ( i < size ) {
    total += arg[ i ];
    i++;
  }
  return total;
}
```

Example 3 – **`do-while()`** Loop

```
int sumArrayDoWhile( int arg[], int size ) {
  int total;
  int i;

  i = 0;
  total = 0;
  do {
    total += arg[ i ];
    i++;
  } while ( i < size );

  return total;
}
```

Example 4 – **Updating Array Elements**

```
void scaleArray( int arg[], int size, int scaleFactor ) {

  int i;

  for ( i = 0; i < size; i++ ) {
    arg[ i ] *= scaleFactor;
  }

  return;
}
```