**Passing arrays to and from methods**

Note: An array has three parts

1) the address of the first indexed variable

2) the base type of the array

3) the size of the array (# of elements)

When an array is used as an array argument to a function, only the first of these three parts is given to the function so you have to pass the size of the array in as well.

An argument to a function may be an entire array, but an argument for an entire array is neither a call-by-value nor a call-by-reference argument. It is an array argument. When an array argument is plugged in for an array parameter, all that is given to the function is the address in memory of the first indexed variable of the array argument (the one indexed by 0).The array argument does not tell the function the size of the array. Therefore, when you have an array parameter to a function, you normally must also have another formal parameter of type int that gives the size of the array.

An array argument is like a call-by-reference argument in the following way: If the functionbody changes the array parameter, then that change is actually made to the array argument (the array sent to the function). If you do not want the function to modify the array argument, you should use the const parameter modifier to create a constant array parameter. eg. void showTheWorld(const int a[], int sizeOfa).

A function may not return an array in the same way that it returns a value of type int or double. You must return a pointer to the array instead.

When a multidemensional array parameter is given in a function heading or function declaration, the size of the first dimension is not given, but the remaining dimension sizes must be given in square brackets. Since the first dimension size is not given, you usually need an additional parameter of type int that gives the size of this first dimension. eg. void getPage(char p[][100], int sizeDimension1)

To calc size of array instead of hard coding it

int sizeOfNumArray = sizeof(numArray)/sizeof(numArray[0]);//calc size of array  
  
**(**Source: Absolute C++ Savitch 3rd edition)