Dereferencing Arrays

You can also dereference an array, just like a regular pointer, using the regular **dereferencing operato**r. Here are some examples (note the parentheses):

```
int array[] = {1, 2, 3, 4, 5};
cout << *array;          // array[0]
cout << *(array + 2);    // array[2]
cout << *array + 2;      // array[0] + 2
```

Similarly, you can **combine dereferencing with address arithmetic**, by using the **subscript operator**; you can use the subscript operator on both pointers and arrays. All of these expressions are true.

```
int array[] = {1, 2, 3, 4, 5};
int *p = array;
*array == *p;              // true
array[0] == p[0];          // also true
array[3] = *(p + 3);       // true again
4[array] = *(4 + array);   // also true
```

In fact, in C++, the subscript operator is just shorthand for a combination of address arithmetic along with dereferencing the resulting address.

```
address[offset] -> offset[address] -> *(address + offset)
```