Arrays & Functions

You cannot pass an array by value to a function as you can a vector. While you can pass an array to a function by reference, you'll almost never do that. Instead, we need to learn about a new way of passing parameters: **pass by address**.

Recall that an array name **is an address**, which you may store inside a pointer.

```
int array[5];
int *p = array;
```

This is the secret to writing functions that process arrays:

- Create a function **with a pointer as a parameter**. You **may** declare this pointer as `int a[]`, indicating that you **intend** to initialize it with the address of the first element in an array.
- **Call the function**, **supplying the name of an array** as the argument.

Here are two prototypes. The first uses the square brackets to declare the pointer variable `a`. The second uses the normal pointer parameter syntax. Both have identical meaning **when as a parameter declaration**.

```
int aSum(const int a[], size_t size);
int aSum(const int *a, size_t size);
```

With "pointer notation", the star comes **before** the name, while with "array notation", the **brackets come after**. A common error, for Java programmers moving to C++, is to write the prototype like this,

```
int aSum(const int[] a, size_t size);
```

which is a syntax error.