Arrays

Sometimes we want to store lists of data together in one place. For example, it might be handy to have a list of all your grades from science class. You could go back later and average all the grades in the list together to find out your final grade in science. In C++, lists are called arrays, and they have some special properties.

You can think of an array like a motel. Let's begin by imagining a single-story motel complex. The rooms are laid out in a row right next to each other. If you stand in front of room 7, you know that room 8 will be right next door, and room 9 will be after room 8. Arrays work in the same way. We can put values inside each of the "rooms" in the array, and since they are numbered logically, we can quickly find any "room" we need.

When you build a motel, you have to decide ahead of time how many rooms you want there to be, and you can't change your mind after the motel is built (without paying a lot of construction costs). In a C++ array, at the beginning of the function you declare the type, name, and size of your array. Going back to the science grade example, we would declare a science grade array like this:

```
float gradeArray[5];
```

The brackets with the number mean that we are constructing an array that is five units in length named *gradeArray*. We declare it as a float because we plan on storing floating point numbers in each of the "rooms" in the array. We can substitute the number 5 with any length of array we want, but once we declare our array to be a certain size, we can't change it.

Now that we've declared an array, we need to put things inside it. Inside our main function, we'll enter five grades into the array. When you put something into an array, you have to tell it where it should go, just like in a motel, you don't get to randomly choose a room. We talk about a specific location in an array by writing the name of the array and then putting the location number in brackets, like this: gradeArray[0] . As with all counting in computer science, we have to begin at 0, so the first "room" in the array motel is room 0, then room 1, etc. In *gradeArray*, there are five locations, which are numbered 0, 1, 2, 3, and 4. Let's put some values into them in the following code:

```
gradeArray[0] = 98.3;
gradeArray[1] = 91.1;
gradeArray[2] = 89.8;
gradeArray[3] = 90.5;
gradeArray[4] = 95.0;
```

Now that we have our grades in an array, we want to average them all together to calculate our final grade. To do this, we need to visit each location in the array (imagine going from door to door in a motel) and sum up the values in each position, and then divide them by the number of values we added. In order to do this systematically, we can use a for-loop. We will loop through each element, or location, in the array and add its value to a variable called *gradeSum* that we initialize ahead of time. After we finish the loop, we will divide *gradeSum* by 5 to get the average. Here's the code for this program:

```
#include <iostream>
using namespace std;

int main(){
        //declare the array and the other variables
```

```
        float gradeArray[5];
        float gradeSum, gradeAverage;
        int i;

        //put some grades into the array
        gradeArray[0] = 98.3;
        gradeArray[1] = 91.1;
        gradeArray[2] = 89.8;
        gradeArray[3] = 90.5;
        gradeArray[4] = 95.0;

        //initialize the gradeSum variable to be 0
        gradeSum = 0;

        //loop through each location in gradeArray (there are 5)
        for (i = 0; i < 5; i++){
                //add the value of the thing stored at location
                //i in gradeArray to gradeSum
                gradeSum += gradeArray[i];
        }

        //to finish the average, divide gradeSum by 5
        gradeAverage = gradeSum / 5;

        //print out the average
        cout << "Your science class average is " << gradeAverage << ".\n";
}
```

Challenge 1:

Can you change the program above so that it takes 7 grades instead of 5 to calculate the average?

Challenge 2:

Can you write a program that asks a user to enter 10 numbers, puts those numbers in an array, and then prints out the numbers in the array backwards?

Hint: use a loop that counts backwards.

Source