CS 102
Introduction to Programming Using C++

Chapter 6

Arrays and Vectors

Homework

- Written homework
- p. 292, R6.1, 2, 6, 10, 11, 13, 14, 23, 25

- Programs
- p. 295, choose one of P6.1, 2, 9
  - If you choose P6.2, choose four parts
- Also choose one of P6.12, 14

An Array We Already Know

- This is something good to remember as we learn about arrays
  - A string is a type of array
- We have already coded a number of array ideas when we used strings
- See if you can find the parallels to strings as we discuss arrays

Why Do We Need Arrays?

- Collecting temperatures
- Your job is to write a program to track today's high and low temperature
- You do this by checking the temperature once per hour
- You keep track of the high and low
- See the program GetTemps.cpp

Program Details

- You create variables temp1, temp2, …, temp24
- Of course, you can't use "…"
  - You have to type all 24 of those
- You have to type all those variable names
- You then type lots of cin statements to read the data

The World Is Always Changing

- The boss comes to you and tells you to change your program
- You need to check the temperature every half hour
- You start by creating temp25, temp26, …, temp48
- You continue to modify your entire program
- Also, the boss wants the average temperature
- The formula is: (temp1 + … + temp48) / 48
- This is getting quite tedious!

The Solution to the Problem

- The solution is the new data type: arrays
- Now look at GetTemps-Array.cpp
- An array is a group of variables that all
  - Have the same name
  - Have the same type
  - Are accessed by their positions
  - Store similar data
- In this case, we could call the array temp

More Array Details

- An array has a definite size
  - The size does not change
  - It cannot be changed
- Items in an array are called elements
- Elements are referred to by their locations
  - The first location is 0, the next is 1, etc.
- The location of an element is called a subscript or an index

Defining an Array

- We have to define an array before we use it
  - This is not anything new
  - We always define variables before we use them
- There are two ways to define them
  - int temps [24];
  - int temps[] = {80, 65, 70, 78, 75};
- There is also a sort-of combination
  - int temps [5] = {70, 80, 85};

The Elements of an Array

- An array can contain anything
- An array of double precision numbers is defined as
  - double lengths [] = {35.25, 34, 35.5, 36, 36,75, 36};
- You can have float, boolean, char, int, double, even string arrays
- There are even more advanced types of arrays

Using the Array

- Let's create a new double array, called costs, that can hold 100 elements: _____
- To print the third element
  - cout << costs [2];
- To set the tenth element to $50.00
  - costs [9] = 50.0;
- Testing if the 17th element is less than 500
  - if (costs [16] < 500.0)
- Anywhere you can use a variable, you can now also use an array element
- Generally, you cannot use the entire array in one statement

What Is Wrong with This Code?

- What is wrong with this code?

```
char *a = malloc(5);  //  Reserve 5 spaces for the array a
a[5] = '\0';
while (i < 5) a[i++] = ' ';
printf  ("'%s'\n", a);
free(a);
return 0;
```

- I read somewhere that when you malloc a pointer, you must free it somewhere, or it would take memory on your computer until you restart it. But this code is still crashing :/

Bounds Errors

- Suppose you declared an int array

```
int number_of_parts = {50, 55, 52, 60, 58, 62};
```

- You can print them using

```
for (int i=0;  i<6; i++)
    cout  <<  number_of_parts [i]  <<  endl;
```

- If you accidentally miss the end, and code

```
for (int i=0;  i<16; i++)
    cout  <<  number_of_parts [i]  <<  endl;
```

  - you will get a bounds error

The Consequences of Bounds Errors

- A bounds error can go unnoticed for a while
- The program will probably crash
- At some point, you may refer to memory outside of your program
- The operating system will notice and abort the program

Using Part of an Array

- Suppose you have the array int scores[10];
- What if you are recording your scores and you only have three scores?
- You can use the first three locations in the array
  - You don't have to use all 10
- The problem is that you have to keep track of how many actual scores you have
- You could create an int called, for example, number_of_scores
  - Start it at 0, indicating that there are 0 elements in the array
  - Increment it as you read a score