Basics of Programming with Arrays

Larry Caretto
Computer Science 106

Computing in Engineering and Science

April 20, 2006

California State University Northridge

Outline

- Review introduction to arrays
  - Declaring and using arrays
  - Array size and maximum subscript
  - Using a variable subscript for arrays
- Writing code with arrays and for loops
  - Array sums
  - Finding maximum and minimum elements in an array
- Data processing with arrays

Representing Data

Run	Data	Math	C++
0	12.3	x_0	x[0]
1	14.4	x_1	x[1]
2	11.8	x_2	x[2]
3	12.5	x_3	x[3]
4	13.2	x_4	x[4]
5	14.1	x_5	x[5]

- C++ array, x[i] used to represent data for which x_i is used in mathe-matical notation

Using Arrays

- Declare arrays in typical way, but add maximum elements, e.g. int v[100];
- Refer to arrays as to any other variable using subscript v[3] or v[k]
  - Must assign value to k before using it as variable subscipt
  - Major tool in arrays is using variable subscript that is for loop index

```
const int N = 200; double a[N];
for ( int j = 0; j < N; j++) a[j] = 0;
```


Maximum Array Subscript

- Array subscripts start at zero
- A declaration double y[N] declares a y array with N elements numbered from y[0] to y[N-1]
  N must be a const
- For loop to handle all elements is

```
for ( int k = 0; k < N; k++ )
```

- C++ does not check to see if an array subscript is in bounds -- an incorrect subscript could affect some other memory location

General Array Processing

- To process each element in an array with N elements, starting with the initial element, use a for loop with index k
  - k starts at zero
  - The continuation condition, k < N, will process elements 0, 1, 2, … , N-1
  - Increment k by 1
- `for ( int k = 0; k < N; k++)`
- Will process elements 0 to N – 1 of array regardless of array size

General Array Processing II

- On the previous chart N means the number of elements defined, not the total number of elements that can be stored in the array
- Sometimes it is more convenient to refer to the subscripts than to the number of elements
- E. g., array whose first and last defined elements have subscripts F and L
- `for ( k = F, k <= L; k++ )`

General Array Processing III

- In the examples that follow, we will generally assume that an array has N elements, whose first subscript is zero
- The for loop command to process each element in such an array is
  `for ( k = 0; k < N; k++)`
- We can use different increments (e.g. k += 3) to skip elements

Setting Array to a Single Value

- Simplest array processing code

```
for (int k = 0; k < N; k++ )
    arr[k] = value;
```

- What is code to declare a double array, x, with 200 elements all set to zero?

```
const int MAX = 200;
double x[MAX]
for (int k = 0; k < MAX; k++ )
    x[k] = 0;
```


Keyboard Input, File Output

```
const int MAX_SIZE = 100;
double z[MAX_SIZE];
ofstream outFile( "array.out" );
for (int i = 0; i < MAX_SIZE; i++)
{
   cout << "Enter z[" << i << "]: ";
   cin >> z[i];
   outFile << i << "  " << z[i];
}
```


File Input, Screen Output

```
const int MAX_SIZE = 100;
double z[MAX_SIZE];
ifstream inFile( "array.dat" );
for (int i = 0; i < MAX_SIZE; i++)
{
   inFile >> z[i];
   cout << "z[" << i << "] = "
        << z[i];
}
```


Computing Array Sums

- Use previous tool of running sum variable along with for loop to include all array elements in the sum
- Will use this code later to compute mean

```
int i;   const int N = …;
double x[N], sum = 0;
// Code to input N values of x[k]
for( i = 0; i < N; i++ )  {} not
{    sum += x[i];      }  needed here
cout << "Array sum = " << sum
```

Trace Array Sum Code

k	x[k]
0	12
1	14
2	11
3	12
4	13
5	14

```
double x[6], sum = 0;
for( i = 0; i < 6; i++ )
{    sum += x[i]; }
```

Before loop sum = 0
i = 0: sum = 0 + 12 = 12
i = 1: sum = 12 + 14 = 26
i = 2: sum = 26 + 11 = 37
i = 3: sum = 37 + 12 = 49
i = 4: sum = 49 + 13 = 62
i = 5: sum = 62 + 14 = 76
i = 6: exit loop

Computing the Mean

$$\bar{x} = \frac{1}{N}\sum_{i=1}^{N} x_i = \frac{1}{N}\sum_{i=0}^{N-1} x_i$$

```
sum = 0;
for( i = 0; i < N; i++ )
{    sum += x[i];      }
average = sum / N;
```

- N data items to average
- Subscripts starts at 0
- Last data item is element N-1 in array
- {} not needed

Finding the Maximum

- How do you find the maximum or minimum in a set of numbers?
- *E. g.*: 13 74 -3 12 91 0 -17 88 -4
- Now that you found the maximum and minimum, how would you explain what you did so a computer can understand?
- Scan the list and remember the largest (smallest) number you have seen and replace if you find one larger (smaller)

Finding the Maximum

- Store the current maximum in a variable (e.g., `max`) and compare `max` to new array values
- If a new value is greater than `max` replace `max` by that value

```
double max = x[0];// initialize max
for ( int i = 1; i < N; i++ )
{    if ( x[i] > max )
        { max = x[i]; }
}
```

- Can omit both sets of braces

Finding the Minimum

- Store the current maximum in a variable (e.g., `min`) and compare `min` to new array values
- If a new value is less than `min` replace `min` by that value

```
double min = x[0];// initialize min
for ( int i = 1; i < N; i++ )
{    if ( x[i] < min )
        { min = x[i]; }
}
```

- Can omit both sets of braces

Initializing Arrays

- We can initialize an array by placing all the data values in braces following the array declaration

```
int x[5] = { 12, 17, -22, 4, 12 };
const int NX = 5;
int x[NX] = { 12, 17, -22, 4, 12 };
int x[] = { 12, 17, -22, 4, 12 };
```

- Note that the maximum size is not required when we initialize an array

Data Processing with Arrays

- In a typical experimental situation we can have fixed conditions (e.g. ambient pressure and temperature)
  - These are scalar (non-array) variables
- We can have several data sets in which two or more variables are measured
  - Each variable measured in each run is an array variable
  - There is one array element for each run

Array Processing Example

- You have taken current and voltage data from a circuit
- There are N pairs of data
- Current is stored as the amps[k] array and voltage as the volts[k] array
- Write the code to compute the average power if arrays are already declared and data on N, volts[] and amps[] are already input

Average Power One

```
double sum = 0;
for ( int k = 0; k < N; k++ )
{
     power[k] = amps[k] * volts[k];
     sum += power[k];
}
double averagePower = sum / N;
cout << "Power = " << averagePower
     << " watts";
```


Average Power Two

```
double sum = 0
for ( int k = 0; k < N; k++ )
{
     power = amps[k] * volts[k];
     sum += power;
}
double averagePower = sum / N;
cout << "Power = " << averagePower
     << " watts";
```


Average Power Three

```
double sum = 0
for ( int k = 0; k < N; k++ )
{
     sum += amps[k] * volts[k];
}
double averagePower = sum / N;
cout << "Power = " << averagePower
     << " watts";
```


Differences in Power Codes

- Used three ways to compute power
- Only one used a power[k] array
- Code works with power not an array or not even a variable
- Usually define arrays when we want to save results of a computation for use in subsequent computations

Average Power One

```
double sum = 0;
for ( int k = 0; k < N; k++ )
{
    power[k] = amps[k] * volts[k];
    sum += power[k];
}
double averagePower = sum / N;
cout << "Power = " << averagePower
  << " watts";
```

The power array could be used in subsequent (or the same) loops

California State University Northridge 25

Average Power Two

```
double sum = 0
for ( int k = 0; k < N; k++ )
{
    power = amps[k] * volts[k];
    sum += power;
}
double averagePower = sum / N;
cout << "Power = " << averagePower
  << " watts";
```

The power variable can be used in the same loop only

California State University Northridge 26

Average Power Three

```
double sum = 0
for ( int k = 0; k < N; k++ )
{
    sum += amps[k] * volts[k];
}
double averagePower = sum / N;
cout << "Power = " << averagePower
  << " watts";
```

The power calculation has to be redone if we want to use power[k] subsequently

California State University Northridge 27

Calculating Tables with Arrays

- We have previously calculated tables where the step between variables was the same
- How can we construct a table with uneven step sizes in the variables?
  - We can define one or two arrays for a one-way or two-way table
  - Each array has all the values that we want in the independent value of the table

California State University Northridge 28

Exercise

- The volume occupied by an idea gas is found from the equation V = nRT/P
  - V is the volume in m^3
  - n is the number of kmols (kgmoles)
  - R = 8.3144 kPa· m^3/kmol·K is the (universal) gas constant
  - T is the temperature in kelvins
  - P is the pressure in kPa
- Write a program that computes V for fixed n and T and several P values

California State University Northridge 29

Exercise II

- Write a program that computes and prints V = nRT/P for a series of pressures using n = 1 kmol and T = 273.15 K
  - P = 1, 2, 5, 10, 20, 50, 100, 200, 500, 1000, 2000, 5000, and 10000 kPa
  - R = 8.3144 kPa· m^3/kmol·K

California State University Northridge 30

Exercise Solution

```
const double R = 8.3144;
const int NP = 13;
double n = 1, T = 273.15, P[NP] =
  {1, 2, 5, 10, 20, 50, 100, 200,
  500, 1000, 2000, 5000, 10000 };
for ( int k = 0; k < NP; k++ )
{
    cout << "\nFor P = " << P[k]
         << " kPa, V = " << n * R
            * T / P[k] << " m^3";
}
```

California State University Northridge

31