Ch 7. Arrays
Part 2

CS 1428
Fall 2011

Jill Seaman

Lecture 18

Operations over arrays

- Except for I/O for char arrays, array operations must be done one element at a time.
  - Input the 8 programming assignment grades for 1 student in CS1428

```
const int NUM_SCORES = 8;
int scores[NUM_SCORES];
cout << "Enter the " << NUM_SCORES
     << " programming assignment scores: " << endl;
cin >> scores[0] >> scores[1];
cin >> scores[2] >> scores[3];
cin >> scores[4] >> scores[5];
cin >> scores[6] >> scores[7];
```

  - Is there a better way?

Array input using a loop

- We can use a for loop to input into the array
  - the subscript can be a variable

```
const int NUM_SCORES = 8;
int scores[NUM_SCORES];
cout << "Enter the " << NUM_SCORES
     << " programming assignment scores: " << endl;

for (int i=0; i < NUM_SCORES; i++) {
   cin >> scores[i];
}
```


Array output using a loop

- We can use a for loop to output the elements of the array

```
const int NUM_SCORES = 8;
int scores[NUM_SCORES];
cout << "Enter the " << NUM_SCORES
     << " programming assignment scores: " << endl;

for (int i=0; i < NUM_SCORES; i++) {
   cin >> scores[i];
}

cout << "You entered these values: ";
for (int i=0; i < NUM_SCORES; i++) {
   cout << scores[i] << " ";
}
cout << endl;
```

Summing values in an array

- We can use a for loop to sum the elements of the array (running total)

```
const int NUM_SCORES = 8;
int scores[NUM_SCORES];
cout << "Enter the " << NUM_SCORES
     << " programming assignment scores: " << endl;

for (int i=0; i < NUM_SCORES; i++) {
   cin >> scores[i];
}

int total = 0;  //initialize accumulator
for (int i=0; i < NUM_SCORES; i++) {
   total = total + scores[i];
}
```


Computing the average for an array

- We can use a for loop to get the average of the values in the array

```
const int NUM_SCORES = 8;
int scores[NUM_SCORES];
cout << "Enter the " << NUM_SCORES
     << " programming assignment scores: " << endl;

for (int i=0; i < NUM_SCORES; i++) {
   cin >> scores[i];
}

int total = 0;  //initialize accumulator
for (int i=0; i < NUM_SCORES; i++) {
   total = total + scores[i];
}
double average =
       static_cast<double>(total) / NUM_SCORES;
```

Finding the maximum value in an array

- We can use a for loop to find the max value:
  Note: keep track of the max value so far

```
const int NUM_SCORES = 8;
int scores[NUM_SCORES];
cout << "Enter the " << NUM_SCORES
     << " programming assignment scores: " << endl;
for (int i=0; i < NUM_SCORES; i++) {
   cin >> scores[i];
}

int maximum = scores[0]; //init max to first elem
for (int i=1; i < NUM_SCORES; i++) { //start i at 1
   if (scores[i] > maximum)
      maximum = scores[i];
}
```


Finding the minimum value in an array

- We can use a for loop to find the min value:
  Note: keep track of the min value so far

```
const int NUM_SCORES = 8;
int scores[NUM_SCORES];
cout << "Enter the " << NUM_SCORES
     << " programming assignment scores: " << endl;
for (int i=0; i < NUM_SCORES; i++) {
   cin >> scores[i];
}

int minimum = scores[0]; //init min to first elem
for (int i=1; i < NUM_SCORES; i++) { //start i at 1
   if (scores[i] < minimum)
      minimum = scores[i];
}
```

Finding the maximum value in an array, **and its position**.

- Keep track of the minimum value, AND what its position is:

```
const int NUM_SCORES = 8;
int scores[NUM_SCORES];
// input code goes here

int indexOfMax = 0;   //init indexOfMax to first
int maximum = scores[0]; //init max to first elem
for (int i=1; i < NUM_SCORES; i++) { //start i at 1
   if (scores[i] > maximum) {
      maximum = scores[i];
      indexOfMax = i;
   }
}
cout << "The highest score was " << maximum
     << " and it was assignment " << indexOfMax+1
     << endl;
```


Counting values in an array that pass a test

- Use a for loop and a counter, incr counter for elements that pass the test (i.e. elem > 75)

```
const int NUM_SCORES = 8;
int scores[NUM_SCORES];
// input code goes here

int count = 0;   //init count to zero
for (int i=0; i < NUM_SCORES; i++) {
   if (scores[i] > 75) {
      count++;
   }
}
cout << "There were " << count
     << " scores above 75." << endl;
```

Array assignment

- Array assignment (a.k.a. array copy).

```
const int SIZE = 4;

int values1[SIZE] = {100, 200, 300, 400};
int values2[SIZE];

values2 = values1;  //WRONG, won't work right

for (int i = 0; i < SIZE; i++) {
   values2[i] = values1[i];
}
```


Array compare (for equality)

- Cannot use == on two arrays.

```
const int SIZE = 4;

int values1[SIZE] = {100, 200, 300, 400};
int values2[SIZE] = {100, 200, 300, 400};

if (values2 == values1) //WRONG, won't work right
  cout << "equal!" << endl;

bool arraysEqual = true;  //flag, assume true
int i = 0;
while (arraysEqual && i < SIZE) {
   if (values1[i] != values2[i])
      arraysEqual = false;
   i++;
}
if (arraysEqual)  cout << "equal!" << endl;
```

Watchout: increment operator

- What is output?

```
int numArray[5] = {6,7,8,9,0};
int count = 2;

numArray[count]++;
numArray[count++];

cout << count << endl;
for (int i=0; i<5; i++) {
   cout << numArray[i] << " ";
}
cout << endl;
```

- numArray[count]++ is (numArray[count])++
- numArray[count++] is numArray[(count++)]

Partially filled arrays

- The programmer does not always know ahead of time how many elements there will be in the array (ie reading from a file).

```
const int MAX_STUDENTS = 100;
int scores[MAX_STUDENTS];

ifstream infile;
infile.open("students.txt");

int count = 0;
while (count<MAX_STUDENTS && infile >> scores[count])
{
   count++;
}
for (int x = 0; x < count; i++)
   //do something with scores[x]
```