Multi-dimension Arrays

Feb. 29, Ch 7
Highlights

- 2D arrays

```c
int box[3][4];
// 3 rows, 4 columns
```
Multidimensional Arrays

So far we have dealt with simple (one dimensional) arrays

We have represented this as all the data being stored in a line

(See: lineWorld.cpp)
Multidimensional Arrays

```c
int matrix[3][5];
matrix[1][3] = 50;
```

- Each row element has a matrix of length 5 associated with it.
- The matrix's length = 3 (number of rows).

```
[0,0] [0,1] [0,2] [0,3] [0,4]
[1,0] [1,1] [1,2] [1,3] [1,4]
[2,0] [2,1] [2,2] [2,3] [2,4]
```
Multidimensional Arrays

If we think of a couple simple (one dimensional) arrays on top of each other...

(See: gridWorld.cpp)
Multidimensional Arrays

Recreate:

![Multidimensional Array Diagram](image)

(See: oneToAHundred.cpp)
Multidimensional Arrays

You should always loop over the left most index (then to the right) when looping (if given the choice)

This is due to how the computer stores the arrays, it is much for efficient to do it in this manner
Arrays can have any number of dimensions

```cpp
int x[3][5][3][2][3][2][5][7][5][4][5][6][4];
x[1][3][2][0][1][1][3][5][3][2][0][2][1] = -7;
```

One annoying thing is that you have to give the length of the array in every dimension (except the last) to functions

(See: functionArrayLength.cpp)