Arrays of Structs

Introduction

- A common way to hold data is in an array.
- A common way to represent data is with a struct.
- Many times you will need to put them both together.
- Accessing the array of structs is fairly easy, once you get used to it.
Example

struct Record
{
    string ID;
    int Value;
    string Description;
};

Record MyArray[10]; // declares an array of // 10 structs of type Record
 // set up a blank record for initialization
Record Blank;
Blank.ID = "";
Blank.Value = 0;
Blank.Description = "";
 // initialize the array to blank
for ( int i = 0; i < 10; i++ )
    MyArray[i] = Blank;
/read in some values that need to be stored
in ID Value Description;
/Store them in an array location
MyArray[0].ID = ID;
MyArray[0].Value = Value;
MyArray[0].Description = Description;
/You can fill the entire array this way, like in a loop or something.
/Now you can do things like sort or print
/them

## Sorting

```c
void Sort ( Record Array[], int size )
{
  Record TempRecord;
  int smallest;
  for ( int i=0; i < size-1; i++ )
  { 
    smallest = i;
    for ( int j = i+1; j < size; j++ )
    {
```
{ 
    if ( Array[smallest].ID > Array[j].ID )
        smallest = j;
}
TempRecord = Array[i];
Array[i] = Array[smallest];
Array[smallest] = TempRecord;
}

**Printing**

```cpp
void Print(Record Array[], int size, ofstream& out)
{
    for ( int i=0; i<size; i++ )
        out << setw(15) << Array[i].ID << setw(15) << Array[i].Value << setw(40) << Array[i].Description << endl;
}
```