Today - Lecture 6 - CS162

1) Topic 2 - Structures
   - what are they
   - how to create them
   - passing structures as arguments
   - arrays of structures

2) Experience structures in a program

Announcements

* Program 1 is due
* Make sure to hit the "submit" button in D2L after uploading files into the dropbox!
* Quiz #3 is on functions. Don't forget to do it before Monday at 11:59pm!

Structures

1) Allows us to group data together!
2) Remember an array requires all elements to be the same data type

so what happens when we want to represent an inventory of information:

- product name
- barcode
- description
- price
- distributor

Think about how hard it would be to represent this with just arrays

```
char name[41];
char description[131];
float price;
char distributor[113];
```

How could you have more than 1 product

Structures can be used to GROUP different data types

1) Start with the keyword struct

2) Follow this with a "tag" name

```
struct product
{
    char name[41];
    char description[131];
    float price;
    char distributor[113];
};
```

This is your choice "tag" name

members

This semicolon is required

3) This is a "specification" that can be used to create variables

4) There is NO memory allocated yet

Using Structures

1. Create structure definitions GLOBALLY

2. Is that OK?

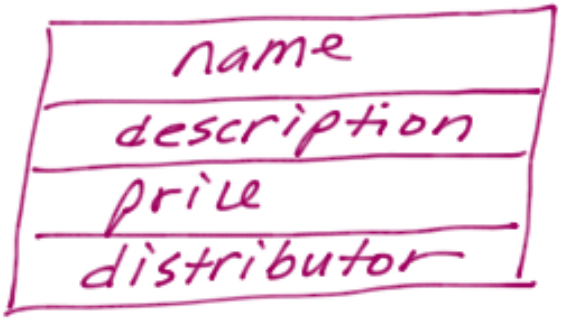
   yes - because a structure declaration is a "specification" and no memory is allocated. So there are no side effects

3. How do we use them?

```
int main()
{
    product item;
    int i;
```

data type — variable

← A local variable



Accessing Members

```
product another;
product item;
```

```
cin.get(item.name, 41, '\n');
```

object . member

variable or object of type struct

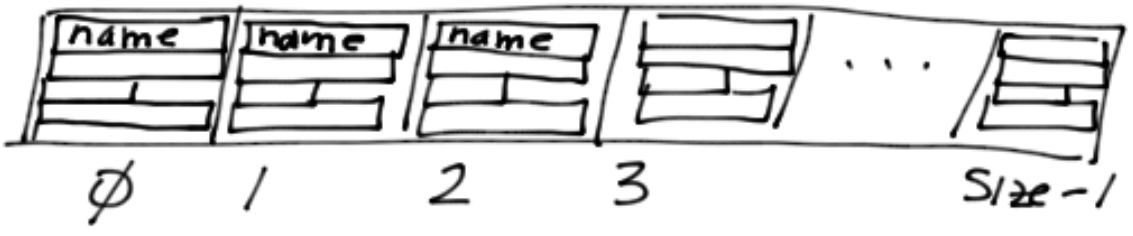
Direct member access operator

Variable . Member

Creating an Array of Structures

1. We can now create multiples of these groupings

```
product inventory[100];
```



2. Accessing Members:

```
cout << inventory[i].price;
```

Direct Member Access operator

A struct object (variable)

Member

PASSING Structures as Arguments

1. Prototype :

```
void inputinventory ( product & an_item);
```

Never pass a struct by value.

2. Function Call :

```
product item;
input_inventory ( item);
```

an object or variable of type struct product

3. Function Implementation:

```
void input_inventory(product & object)
{
    cout << "Enter a name: ";
    cin.get(object.name, 41);
    cin.ignore(100, '\n');
```

Note

Notice we can use this...

```
product inventory[100];
int num_items = 0;

for(int i = 0; i<100; ++i)
    input_inventory(inventory[i]);
```

Passing one object by Ref.

one structure instance

To Pass an entire array:

prototype: `void display_all(product array[], int num);`

call: `display_all(inventory, i);`

- `inventory`: array
- `i`: the number of elements in the array.

Next - Develop code using structs

1) Manage a List of Movies

2) Step 1 - Create a structure to manage 1 move

3) Then create an array of movies