CSC 209 Review 7 Solution

August 26, 2020

1 Exercises

1. First, I need to justify if the folllowing declarations legal on an individual basis:

   `struct {int x, y;} x;`

   `struct {int x, y;} y;`

   The struct `struct {int x, y;} x;` is legal. `struct {int x, y;} x;` is equivalent to

```
struct {
    int x;
    int y;
} x;
```

   and 'x' beside struct represents variable of that type. It is used to declare struct and access members of the struct (e.g. `x.x`,, `x.y`).

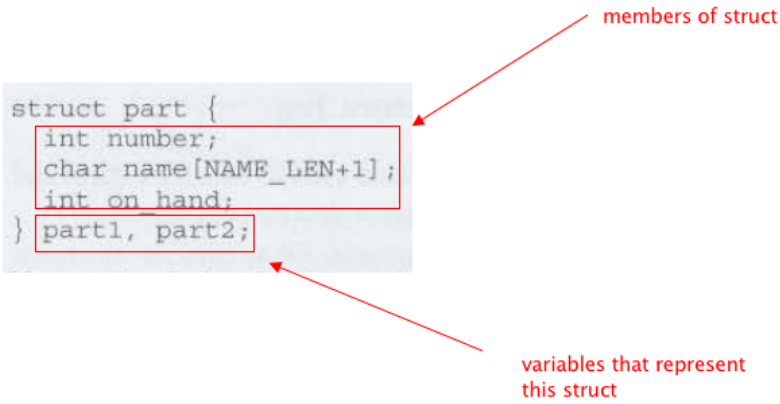
   The same is true for `struct {int x, y;} y;`.

   Second, I need to answer if both declarations of struct can appear in a program.

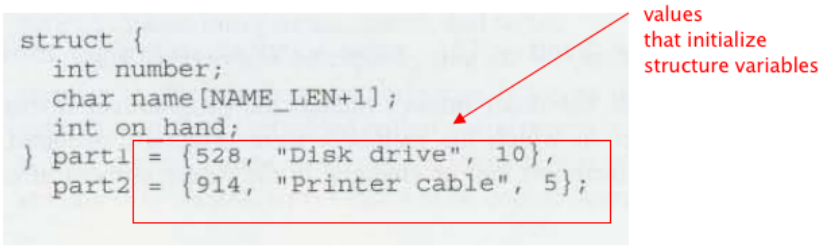
   The answer is yes. Each structure has a separate name space for it's members.

   Notes

   - **Declaring Structure Variables**
     - Struct can have many variables that represent the same struct



- **Initializing Structure Variables**
  - Struct can be initialized with preset values (like python class under __init__)



2. a) I need to declare structure variables named `c1`, `c2` and `c3`, each having members `real` and `imaginary` of type `double`.

   The solution to this problem is:

```
1    struct {
2        double real, imaginary;
3    } c1, c2, c3;
```

   b) I need to modify the declaration in part a) so that

   - `c1`'s members initially have the values 0.0 and 1.0
   - `c2`'s members initially have the values 1.0 and 0.0
   - `c3` is not initialized

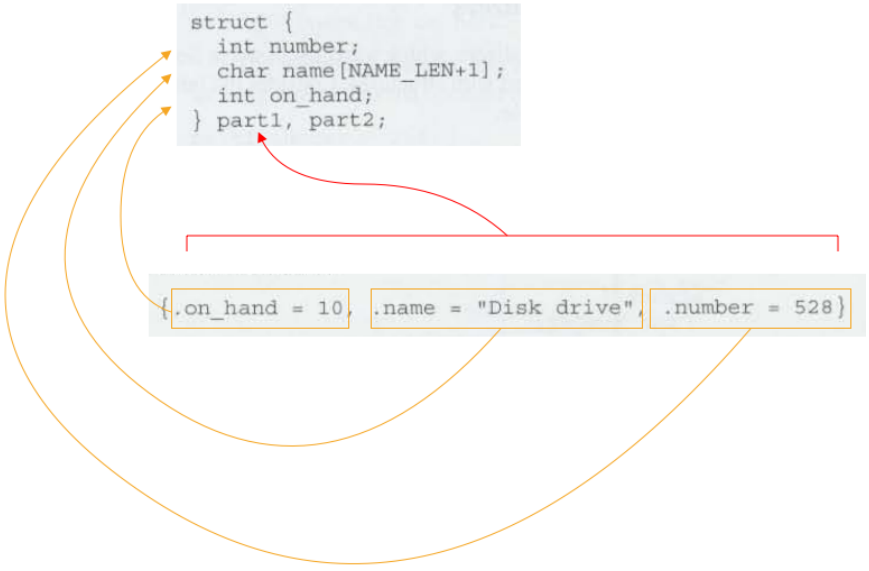
   The solution to this problem is:

```
struct {
    double real, imaginary;
} c1 = {0.0, 1.0},
  c2 = {1.0, 0.0},
  c3;
```

Notes

- **Designated Initializer**
  - Allows specific member variable to be initialized
  - Allows member variables to be initialized in any order

Example



c) I need to write statements that add the corresponding members of `c1` and `c2` and store the result in `c3`.

The solution to this problem is:

```
struct {
    double real, imaginary;
} c1 = {0.0, 1.0},
    c2 = {1.0, 0.0},
    c3;

...

c3 = c1 + c2;
```