User Defined Types

11-12-2003

Opening Discussion

- What did we talk about last class?
- Do you have questions about the assignment?
- What types do you know of in C right now?

Motivation

- Sometimes pieces of data just seem to go together. You might be seeing this is assignment #7. In the grade book you have names with grades associated with them. These are parallel arrays. If you wanted to move a student to a different "row" you would need to move all their grades as well. It would be nice to have a single type to store all the data for a student.

Structures in C

- Of course, C provides this ability for us through the use of structures.
- A structure is a conglomerate of data, but unlike an array the members can be of different types and we access them by name instead of by an integer index.

```
struct {
     char name[40];
     double grade[19];
} stu1,stu2;
```

Initializing structs

- We can initialize structs much like we did arrays. The value order needs to match that in the declaration. (I don't recommend this).

```
struct {
     double x,y,z;
     double vx,vy,vz;
} particle={1.0, 0.0, 1.0, 0.0, 1.0, -0.1};
struct {
     char name[20];
     int score;
} me={"Mark",100};
```

Selecting Members

- The way we access members of a structure is using the '.' notation.
- We put the variable name first, then have a period followed by the member name that we are interested in. This can be used just like any variable of the proper type.

```
scanf("%s",stu1.name);
scanf("%lf",&(stu1.grade[i]));
particle.x+=dt*particle.vx;
```

Structures as Types

- So far we have declared variables for these structures, but we don't have names to refer to them by to use them as real types. We couldn't declare vars in more than one place or pass or return them.
- One way around this is to use a structure tag. We put a name after the word struct and can later refer to the type with struct followed by that name.

typedef (An Alternate Approach)

- In C, the typedef keyword allows you to define an alternate name for any type.
  - typedef int bool
- The first argument can also be a struct. This allows us to not use the struct keyword every time we use that type.
- Which method you choose is completely up to you.

Assignments of structs

- When you do an assignment from one struct variable to another, C simply copies all the memory for that struct from one location to the other.
- The same is true when structs are passed by value or returned. This can cause significant overhead if the structure is very large. For that reason structs are often passed by reference even if we don't need to change them.

Code

- Now let’s write some code that shows you how to use the struct keyword in your code.

Minute Essay

- Create a struct type that stores an employee record with name and SSN as strings plus their hourly wage in a double.