structs in C

Structs allow a programmer to group different types of data into a single entity.

```c
struct personType
{
    char name[20];
    char rank;
    int serno;
};
```

The above statement is the declaration of a new type! It does NOT declare any storage!

The individual components of the struct are called members.

Style point: Use the word "type" in the struct name.

A struct Example

To declare a struct "variable":

```c
personType soldier;
```

To refer to members of the struct:

```c
cin >> soldier.name >> soldier.rank 
    >> soldier.serno;
```

soldier is called an "instance" of the personType struct.
structs with Array Members

Members can be any type, including arrays and other structs.

```c
struct studenttype
{
    int studid;
    int tests[3];
    int quizzes[3];
    int final;
    char coursegrade;
};
```

structs with Arrays

Some examples using arrays within structs

```c
studenttype student;

for (i = 0; i < 3; i++)
    student.quizzes[i] = 0;

testavg = (student.tests[0] +
            student.tests[1]
            student.tests[2])/3.0;

or

testavg = 0;
for (i = 0; i < 3; i++)
    testavg += student.tests[i];
    testavg = testavg/3.0;
```
Arrays of structs

```
studentype allstudents[50];

for (i = 0; i < 50; i++)
    for (j = 0; j < 8; j++)
        allstudents[i].quizzees[j] = 0;

for (i = 0; i < 50; i++)
{
    cin >> allstudents[i].studid;
    for(j = 0; j < 8; j++)
        cin >> allstudents[i].tests[j];
    for(j = 0; j < 8; j++)
        cin >> allstudents[i].quizzees[j];
    cin >> allstudents[i].final;
}
```

structs and Functions

```
struct examtype
{
    int a, b, c;
    float x, y;
};

int func1(examtype a)
{
    int suminte;
    suminte = a.a + a.b + a.c;
    return suminte;
} // END func1

void func2(examtype a)
{
    s.x = a.x + a.b + a.c;
    s.y = s.x / s.;
} // END func2

int main
{
    examtype arg;
    int r;

    r = func1(arg);
    func2(arg);
} // END main
```

* structs are passed by value by default *