C PROGRAMMING FOR ENGINEERS

Defined Structures
Array of Strutures

Basics of a Structure

- A structure is a collection of data values, called data *members*.
- A structure is a single unit.
- The data *members* of a structure can be of different data types.
- Structures are often called *aggregate* data types.

Defining a Structure

- General form:

Must start with the keyword struct.

Name can be any valid name, but it should be something meaningful.

```
struct name
{
    variable declaration;
    variable declaration;
    .
    .
};
```

← Notice the ; as part of the SYNTAX.

The keyword struct indicates that you are defining a structure. Generally structure definitions are placed immediately after preprocessor directives.

Example of a Structure

```
struct weather
{
    int year;
    double snowfall;
    char month,units;
};
```

This is called the structure tag.

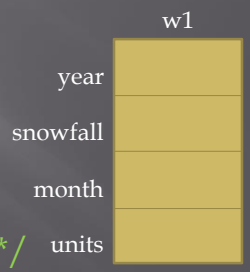
- The above statements define a structure.
- The name of the structure is weather, and it is the structure tag.
- The variables inside the braces are the members of the structure. They can be any valid C data type.
- The tag may be used to declare a data type just like declaring int, double, or char in the main function.

Initializing a Structure

```
struct weather
{
    int year;
    double snowfall;
    char month, units;
};

int main(void)
{
    /* Declare and initialize variables */
    struct weather w1={2013,42.5,"March","inches"};
```

Notice that the structure is defined outside of the main function.



Structure Member Operator and Assignment Operator

Example:

```
struct airplane
{
    char make, color;
    double price;
    int year;
};

int main(void)
{
    /* Declare variables */
    struct airplane plane1, plane2;

    /* Intialize plane1 */
    plane1.year = 1970;
    plane1.price = 35000;
    plane1.color = "blue/white";
    plane1.make = "Piper";

    /* Assign plane1 to plane2 */
    plane2 = plane1;
```

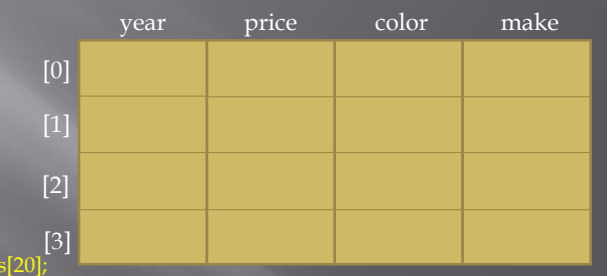
The period is called the structure member operator.

Array of Structures

- An array of structures may be declared in the same way as other C data types.
- Example:

```
struct airplane
{
    char make, color;
    double price;
    int year;
};

int main(void)
{
  struct airplane planes[20];
```



- The above array planes would be able to store 20 airplanes with different attributes for each airplane.
- The first array position would be planes[0] just like numerical arrays.
- Example:

```
planes[0].year = 1970;
planes[0].price = 35000;
planes[0].color = "blue/white";
planes[0].make = "Piper";
```