March 17, 2014 at 12:51

1. Introduction. This implements programs from Chapter 1 in “Data Structures—An Advanced Approach using C”

2. Structures. **Structures** are a compound data type that "*contains an arbitrary group of related data*". A structure can can contain fields of any kind of data, including other structures. A structure type name (or *tag*) is optional, used when defining a type.

```
#define WORDLENGTH 100
#define WORDCOUNT 100
  typedef struct wordcount {
    char word[WORDLENGTH];
    int frequency;
  };
  wordcount wordfrequency[WORDCOUNT];
```

3. It is also possible to define a type or structure and create a variable all at once:

```
  typedef struct wordcount2 {
    char word[WORDLENGTH];
    int frequency;
  } wordfrequency2[WORDCOUNT];
```

4. An example of when the tag might be omitted is shown below:

```
  typedef struct employee_data {
    struct {
      char street[16];
      char city[8];
      char state[2];
      int zip_code;
    } address;
    struct {
      int salary;
      int years_employed;
    } misc;
  };
```

5. Operations on Structures. The most common operation on stuctures is member access.

```
employee_data edwin;
printf("%d\n", edwin.address.zip_code);
```

6. Index.

address: 4, 5.
city: 4.
edwin: 5.
employee_data: 4, 5.
frequency: 2, 3.
misc: 4.
printf: 5.
salary: 4.
state: 4.
street: 4.
word: 2, 3.
wordcount: 2.
`WORDCOUNT`: 2, 3.
wordcount2: 3.
wordfrequency: 2.
wordfrequency2: 3.
`WORDLENGTH`: 2, 3.
years_employed: 4.
zip_code: 4, 5.

COMPSCI

	Section	Page
Introduction	1	1
Structures	2	2
Operations on Structures	5	3
Index	6	4