Structure: Structure is a user-defined data type in C language which allows us to combine data of different types together.

Struct keyword is used to define a structure. struct defines a new data type which is a collection of primary and derived data types.

Sub: Chapter: 11 (Structures and Union)

Day | | | | | | | |

Time: Date: / /

What is structure / Define structure?

→ A structure is a keyword that create user defined data type in C. A structure create a data type that can be used to group item of possibly different types into a single type. We can deffine a structure as follows:

```
Struct book_bank → structure tag
{
    char title[20];
    char author[15];
    int pages;
    float prices;
};
```

Template

structure Element

Here, the keyword struct declares a structure to hold the details of four data fields, namely title, authors, pages, prices. These fields are called structure elements or members. Each member belong to a different type of data. book_bank is named of the structure and it is called the structure tag. This structure tag is used to declare structure variable of its type, later in the program.

Sub :

Day

Time : Date : / /

In defining a structure, we may note the following syntax:

1. The template is terminated with a semicolon.
2. Each structure element is declared independently for its name and type in a separate statement inside the template.
3. The tag name such as book_bank can be used to declare structure variable of its type, later in the program.

Q What are the difference between structure and union?

→ Difference between structure and union:

1. A structure is a user-defined data type available in C that allows to combine data items of different kinds. Whereas a union is a special data type available in C that allows storing different data types in the same memory location.
2. The keyword struct is used to define a structure where the keyword union is used to define a union.

Sub:

Day

Time: Date: / /

3. The size of structure is greater than or equal to the sum of sizes of its members. On the other hand, the size of union is equal to the size of the largest member.

4. Each member within a structure, is assigned unique storage area of location. Whereas all members of a union used the same location.

5. Individual member can be accessed at a time. But, only one member can be accessed at a time.

In struct data type, In union data type,

6. Several member of a structure can initialize at once but only the first member of a union can be ~~intialized~~ initialized.

Sub : Day Time : Date : / /

Q What is the difference between Arrays and structure?

→ Difference between arrays and structure:

1. Array refers to a collection of related data elements of same type. Whereas structure refers to a collection of different data elements of different data type.

2. Array uses "[]" (square bracket) for element access. Where structure uses "." (Dot operator) for element access.

3. Array size is fixed and it is basically the number of elements, structure size is not fixed as each element of structure can be of different type and size.

4. Array declaration is done simply using [] and not any keyword. Where structure declaration is done with the help of "struct" keyword.

5. An array behaves like a built in data type. But a structure is a user-defined data type.

6. Array elements are accessed by their index. Whereas structure elements are accessed by their names.

Sub:

Day

Time: Date: / /

Q. How to declare structure variable?

→ A structure create a data type that can be used to group item possibly different types into a single type. For declaring structure variable, first we need to create a template using keyword struct. Creating a structure data type, it includes following elements:

1. The keyword struct
2. The structure tag name
3. List of members of different data type
4. terminating with a semicolon.

For example,

```
struct book_bank
{
    char title[20];
    char author[15];
    int pages;
    float price;
};
```

structure tag/tagname

Here the book-bank is a data type. Now we declare the structure variable and it includes these following elements:

1. The keyword struct
2. The structure tag name
3. List of variables names seperated by comma
4. A terminating semicolon.

Sub:

Time: Date: / /

For example, the statement

```
struct book_bank book1, book2, book3;
```

Declares book1, book2, book3 as variables of type struct book_bank.

The complete declaration might look like this:

```
struct book_bank
{
    char title[20];
    char author[15];
    int pages;
    float price;
};
int main()
{
    struct book_bank book1, book2, book3;
    return 0;
}
```

Alternative:

Sub: _____ Time: Date: / /

Alternative:

```
struct book_bank
{
    char title[20];
    char author[15];
    int pages;
    float price;
} book1, book2, book3;
```

Q. What are the ~~rules~~ elements for ~~int~~ initialize of a structure variable?

→ The compile-time initialization of a structure variable must have the following elements:

Element

1. The keyword struct.
2. The struct tag name.
3. The name of the variable to be declared.
4. The assignment operator "="
5. A set of values for the members of the structure variable, separated by commas and enclosed in a braces.
6. A terminating semicolon.

Sub :

Time : Date : / /

What is union?

→ Union is an user defined data type in C programming language. It is a collection of variables of different ~~types~~ data types in the same memory location. We can define a union with many members, but at a given point of time, only one members can contain a value.

We use the union keyword to define unions. For example:

```
union item
{
    int m;
    float x;
    char c;
} code;
```

This declares a variable code of type union item. The union contains three members, each with a different data type. To access a union members, we can use the same syntax that we use for structure members.

```
code.m
code.x
code.c
```

Sub:

Day

Time: Date: / /

So, the statement is,

```
code.m = 379;
code.x = 7859.36;
printf("%d", code.m);
```