Definition of Structure

A structure is a custom data type in the C language. Structures can hold multiple members of different data types under a single unit. The elements of a structure are stored in contiguous memory locations and can be retrieved and accessed at any time. Every data object in a structure is a member or field.

How to Define a Structure?

A structure is defined using the struct statement. The struct keyword defines a new data type with more than one member.

Syntax of Declaring a Structure

```
struct [structure name]

{

type member_1;

type member_2;

. . .

type member_n;

};
```

Example of Structure

```
struct student

{

    int rollno;

    char name[50];

    string phone;

 };
```

Definition of Union

A Union is a user-defined data type. It is like the structure, except that all its members start at the exact location in memory. The union combines objects of different data types in the exact memory location. A user can define a union with many members, but only one member can hold a value at any given time. The storage space allocated for the union variable is equal to the total space required by the most prominent data member of the union.

Union provides such variables that can be accessed in several ways and the exact memory location simultaneously. A union provides an efficient way of using a single memory location for various tasks.

How to Define a Union?

The union statement is used for defining a union. It defines a new data type that can store multiple member variables of different data types in the exact memory location. The syntax to define a union using the union keyword is similar to defining a structure.

Syntax of Declaring a Union

```
union [union name]

{

type member_1;

type member_2;

. . .

type member_n;

};
```

Example of Union

```
union Student {

   char name[32];

   int age;

   string email;

 };
```

Similarities between Structure and Union

The following are the similarities between structure and union:

- Both structure and union are the custom data types that store different types of data together as a single entity
- The structure and union members can be objects of any type, such as other structures, unions, or arrays.
- Both structures or unions can be passed by value to a function and returned to the value by functions. The argument will need to have the same type as the function parameter.
- To access members, we use the '.' operator.

Structure	**Union**
We use the struct statement to define a structure.	We use the union keyword to define a union.
Every member is assigned a unique memory location.	All the data members share a memory location.
Change in the value of one data member does not affect other data members in the structure.	Change in the value of one data member affects the value of other data members.
You can initialize multiple members at a time.	You can initialize only the first member at once.
A structure can store multiple values of the different members.	A union stores one value at a time for all of its members
A structure's total size is the sum of the size of every data member.	A union's total size is the size of the largest data member.
Users can access or retrieve any member at a time.	You can access or retrieve only one member at a time.