Structure and Union

Structure

In C programming, a structure is a composite data type that allows you to group together variables of different data types under a single name. Structure is a collection of one or more variables, possibly of different data types (eg. Int, float, char) grouped together under a single name for convenient handling. This grouping makes it easier to organize and work with related pieces of data.

The syntax for declaring a structure in C is as follows:

```
struct structure_name {
   // member variables
   data_type1 member1;
   data_type2 member2;
   // ... more members if needed
};
```

Simple Example:

```
#include <stdio.h>
#include <conio.h>
#include <string.h>
// Define a structure named 'Person'
struct Person {
   char name[50];
   int age;
   float height;
};
main() {
   // Declare a variable of type 'Person'
   struct Person person1;
   // Access and modify members of the structure
   strcpy(person1.name, "John Doe");
   person1.age = 25;
   person1.height = 5.9;
   // Display information
   printf("Name: %s\n", person1.name);
   printf("Age: %d\n", person1.age);
   printf("Height: %.2f\n", person1.height);
   getch();
}
```

In this example, **struct Person** defines a structure with three members: a character array **name**, an integer **age**, and a float **height**. The **main** function then declares a variable **person1** of type **struct Person** and sets its members.

Structures are useful for organizing related data, and you can also have arrays of structures, nested structures, and use them in various ways to model complex data structures in your programs.

Union:

The union is a user-defined data types such every union member takes memory that contains a variety of objects. Such that the union members share space thereby resulting, conserves storage.

The syntax for declaring a union in C is as follows:

```
union union_name {
   // member variables
   data_type1 member1;
   data_type2 member2;
   // ... more members if needed
};
```

Simple Example:

```
#include <stdio.h>
#include <conio.h>
#include <string.h>
// Define a structure named 'Person'
union Person {
   char name[50];
   int age;
   float height;
};
main() {
   // Declare a variable of type 'Person'
   union Person person1;
   // Access and modify members of the structure
   strcpy(person1.name, "John Doe");
   person1.age = 25;
   person1.height = 5.9;
   // Display information
   printf("Name: %s\n", person1.name);
   printf("Age: %d\n", person1.age);
   printf("Height: %.2f\n", person1.height);
   getch();
}
```

Different between Union and Structure:

Union	Structure
1. The memory occupied by the union is of the highest data type of all.	1. Memory occurred by structure is the sum of individual data type.
2. Only one member is active at a time, so the active member can only be accessed.	2. All data (members) can be accessed simultaneously.

3. It cannot take part in the complex data structure.	3. It can take part in complex data structure.
4. Memory allocation is performed by sharing the memory with highest data type.	4. Memory allocation of every element is independently.
5. Example: *union sample{* *int a;* *char b[20];* *} s1;*	5. Example: *structure sample{* *int a;* *char b[20];* *} s1;*