**What is the structure of a C program?**

BASIC STRUCTURE OF A C PROGRAM

Basic Structure of a C program contains following sections,

Documentation Section   
Link Section   
Definition Section   
Global Declaration Section   
main()   
{   
Declaration Section   
Executable part   
}   
Subprogram section   
Function 1   
Function 2   
.   
.   
function n

**- The Documentation**Section consists of a set of comment lines giving the name of the program and other details.   
**-The Link Section**provides instructions to the compiler to link functions from the system library. C program depends upon some header files for function definition that are used in the program. Each header file has extension ‘.h’. The header files are included at the beginning of the program in the C language. These files should be included using #include directive as given below   
**Example:**   
#include

(This will find header file in standard directory)   
or   
#include”stdio.h”( This will find header file in Current and Standard directory)   
**-The Definition Section** defines all symbolic constants.   
**-The Global Declaration Section:** There are some variables and those variables are declared in this section that is outside of all functions.   
**-main() function:** Every C program must have one main() function section. int main(void): This is the function definition for main().Parenthesis followed to main is to tell the user again that main() is a function. Int main(void) function return an integer.   
**void main(void):** This function takes no arguments and return nothing.   
**void main():** This function returns nothing and takes no arguments. The program contains statements that are enclosed within the braces. The opening braces “{“ and closing braces “}”.  
In these two braces main() function contains two parts,  
declaration and executable part. It is user defined function. The Opening braces sometimes called logical start and Closing braces known as logical end of the program.   
**-Declaration Part declares** all the variables used in the executable part. There should be at least one statement in the **executable part**which contains instructions to perform certain task. The declaration and executable part must appear between the opening and closing braces. All statements in the declaration part should end with the semicolon.   
**-The Subprogram Section** contains all the user defined functions that are called in the main function.

EXAMPLE PROGRAM:

1. /\*Documentation Section: program to find the area of circle\*/
2. #include<stdio.h> /\*link section\*/
3. #include<conio.h> /\*link section\*/
4. #define PI 3.14 /\*definition section\*/
5. **float** area; /\*global declaration section\*/
6. **void** main()
7. {
8. **float** r; /\*declaration part\*/
9. printf("Enter the radius of the circle\n"); /\*executable part starts here\*/
10. scanf("%f",&r);
11. area=PI\*r\*r;
12. printf("Area of the circle=%f",area);
13. getch();
14. }

C supports a constructed data type known as structures. A structure contains a number of data types grouped together. These data types may or may not be of the same type.

Structure help to organize complex data in a more meaningful way. It is a powerful concept that we may often to use in our program design.

**Declaring a structure**

struct book

{

char name ;

float price ;

int pages ;

} ;

This statement defines a new data type called struct book. Each variable of this data type will consist of a character variable called name, a float variable called price and an integer variable called pages. The general form of a structure declaration statement is given below:

struct <structure name>

{

structure element 1 ;

structure element 2 ;

structure element 3 ;

...... ......

} ;

Once the new structure data type has been defined one or more variables can be declared to be of that type. For example the variables b1, b2, b3 can be declared to be of the type struct book, as,

struct book b1, b2, b3 ;

This statement sets aside space in memory. It makes available space to hold all the elements in the structure—in this case, 7 bytes—one for name, four for price and two for pages. These bytes are always in adjacent memory locations.

If we so desire, we can combine the declaration of the structure type and the structure variables in one statement.

For example,

struct book

{

char name ;

float price ;

int pages ;

} ;

struct book b1, b2, b3 ;

is same as...

struct book

{

char name ;

float price ;

int pages ;

} b1, b2, b3 ;

or even... struct

{

char name ;

float price ;

int pages ;

} b1, b2, b3 ;

Like primary variables and arrays, structure variables can also be initialized where they are declared. The format used is quite similar to that used to initiate arrays.

struct book

{

char name[10] ;

float price ;

int pages ;

} ;

struct book b1 = { "Basic", 130.00, 550 } ;

struct book b2 = { "Physics", 150.80, 800 } ;

Note the following points while declaring a structure type:

(a) The closing brace in the structure type declaration must be followed by a semicolon.

(b) It is important to understand that a structure type declaration does not tell the compiler to reserve any space in memory. All a structure declaration does is, it defines the ‘form’ of the structure.

(c) Usually structure type declaration appears at the top of the source code file, before any variables or functions are defined. In very large programs they are usually put in a separate header file, and the file is included (using the preprocessor directive #include) in whichever program we want to use this structure type.

Accessing Structure Elements

Having declared the structure type and the structure variables, let us see how the elements of the structure can be accessed.

In arrays we can access individual elements of an array using a subscript. Structures use a different scheme. They use a dot (.) operator. So to refer to pages of the structure defined in our sample program we have to use,

b1.pages

Similarly, to refer to price we would use,

b1.price

Note that before the dot there must always be a structure variable and after the dot there must always be a structure element.

**Basic Structure of a C Program:**

* **Documentation section :** The documentation section consists of a set of comment lines giving the name of the program, the author and other details, which the programmer would like to use later.
* **Link section :**The link section provides instructions to the compiler to link functions from the system library.
* **Definition section :**The definition section defines all symbolic constants.
* **Global declaration section :**There are some variables that are used in more than one function. Such variables are called global variables and are declared in the global declaration section that is outside of all the functions. This section also declares all the user-defined functions.
* **main () function section :** Every C program must have one main function section. This section contains two parts; declaration part and executable part**Declaration part :** The declaration part declares all the variables used in the executable part.**Executable part :** There is at least one statement in the executable part. These two parts must appear between the opening and closing braces. The program execution begins at the opening brace and ends at the closing brace. The closing brace of the main function is the logical end of the program. All statements in the declaration and executable part end with a semicolon.
* **Subprogram section :** The subprogram section contains all the user-defined functions that are called in the main () function. User-defined functions are generally placed immediately after the main () function, although they may appear in any order. Note:All section, except the main () function section may be absent when they are not required.

**Sample C Program:**

#include<stdio.h> <———————-Preprocessing Directive

void main()

{ <——————–Start of a Program

/\*………….Printing Starts………….\*/

Printf(“Learn at every moment”);

/\*………….Printing starts……..\*/

} <———————-End of a Program

* In C many library functions are grouped category-wise and stored in different files known as header files. Ex. stdio.h–>standard input output header file
* To use the functions defined in the header file that need to be included in the program
* This can be achieved by the preprocessing directive “#include”
* “#include” includes the content of header file(stdio.h) at the beginning of program.
* Explanation of void main in detail is present later.

**A structure** is another user defined data type available in C that allows to combine data items of different kinds.

Structures are used to represent a record. Suppose you want to keep track of your books in a library. You might want to track the following attributes about each book −

* Title
* Author
* Subject
* Book ID

**Defining a Structure**

To define a structure, you must use the **struct** statement. The struct statement defines a new data type, with more than one member. The format of the struct statement is as follows −

1. **struct** [structure tag] {
3. member definition;
4. member definition;
5. ...
6. member definition;
7. } [one **or** more structure variables];

The **structure tag** is optional and each member definition is a normal variable definition, such as int i; or float f; or any other valid variable definition. At the end of the structure's definition, before the final semicolon, you can specify one or more structure variables but it is optional. Here is the way you would declare the Book structure −

1. **struct** Books {
2. **char** title[50];
3. **char** author[50];
4. **char** subject[100];
5. **int** book\_id;
6. } book;

**Accessing Structure Members**

To access any member of a structure, we use the **member access operator (.)**. The member access operator is coded as a period between the structure variable name and the structure member that we wish to access. You would use the keyword **struct** to define variables of structure type. The following example shows how to use a structure in a program −

1. #include <stdio.h>
2. #include <string.h>
4. **struct** Books {
5. **char** title[50];
6. **char** author[50];
7. **char** subject[100];
8. **int** book\_id;
9. };
11. **int** main( ) {
13. **struct** Books Book1; /\* Declare Book1 of type Book \*/
14. **struct** Books Book2; /\* Declare Book2 of type Book \*/
16. /\* book 1 specification \*/
17. strcpy( Book1.title, "C Programming");
18. strcpy( Book1.author, "Nuha Ali");
19. strcpy( Book1.subject, "C Programming Tutorial");
20. Book1.book\_id = 6495407;
22. /\* book 2 specification \*/
23. strcpy( Book2.title, "Telecom Billing");
24. strcpy( Book2.author, "Zara Ali");
25. strcpy( Book2.subject, "Telecom Billing Tutorial");
26. Book2.book\_id = 6495700;
28. /\* print Book1 info \*/
29. printf( "Book 1 title : %s\n", Book1.title);
30. printf( "Book 1 author : %s\n", Book1.author);
31. printf( "Book 1 subject : %s\n", Book1.subject);
32. printf( "Book 1 book\_id : %d\n", Book1.book\_id);
34. /\* print Book2 info \*/
35. printf( "Book 2 title : %s\n", Book2.title);
36. printf( "Book 2 author : %s\n", Book2.author);
37. printf( "Book 2 subject : %s\n", Book2.subject);
38. printf( "Book 2 book\_id : %d\n", Book2.book\_id);
40. **return** 0;
41. }

When the above code is compiled and executed, it produces the following result −

1. Book 1 title : C Programming
2. Book 1 author : Nuha Ali
3. Book 1 subject : C Programming Tutorial
4. Book 1 book\_id : 6495407
5. Book 2 title : Telecom Billing
6. Book 2 author : Zara Ali
7. Book 2 subject : Telecom Billing Tutorial
8. Book 2 book\_id : 6495700

**Structures as Function Arguments**

You can pass a structure as a function argument in the same way as you pass any other variable or pointer.

1. #include <stdio.h>
2. #include <string.h>
4. **struct** Books {
5. **char** title[50];
6. **char** author[50];
7. **char** subject[100];
8. **int** book\_id;
9. };
11. /\* function declaration \*/
12. **void** printBook( **struct** Books book );
14. **int** main( ) {
16. **struct** Books Book1; /\* Declare Book1 of type Book \*/
17. **struct** Books Book2; /\* Declare Book2 of type Book \*/
19. /\* book 1 specification \*/
20. strcpy( Book1.title, "C Programming");
21. strcpy( Book1.author, "Nuha Ali");
22. strcpy( Book1.subject, "C Programming Tutorial");
23. Book1.book\_id = 6495407;
25. /\* book 2 specification \*/
26. strcpy( Book2.title, "Telecom Billing");
27. strcpy( Book2.author, "Zara Ali");
28. strcpy( Book2.subject, "Telecom Billing Tutorial");
29. Book2.book\_id = 6495700;
31. /\* print Book1 info \*/
32. printBook( Book1 );
34. /\* Print Book2 info \*/
35. printBook( Book2 );
37. **return** 0;
38. }
40. **void** printBook( **struct** Books book ) {
42. printf( "Book title : %s\n", book.title);
43. printf( "Book author : %s\n", book.author);
44. printf( "Book subject : %s\n", book.subject);
45. printf( "Book book\_id : %d\n", book.book\_id);
46. }

When the above code is compiled and executed, it produces the following result −

1. Book title : C Programming
2. Book author : Nuha Ali
3. Book subject : C Programming Tutorial
4. Book book\_id : 6495407
5. Book title : Telecom Billing
6. Book author : Zara Ali
7. Book subject : Telecom Billing Tutorial
8. Book book\_id : 6495700

**Pointers to Structures**

You can define pointers to structures in the same way as you define pointer to any other variable −

1. **struct** Books \*struct\_pointer;

Now, you can store the address of a structure variable in the above defined pointer variable. To find the address of a structure variable, place the '&'; operator before the structure's name as follows −

1. struct\_pointer = &Book1;

To access the members of a structure using a pointer to that structure, you must use the → operator as follows −

1. struct\_pointer->title;

Let us re-write the above example using structure pointer.

1. #include <stdio.h>
2. #include <string.h>
4. **struct** Books {
5. **char** title[50];
6. **char** author[50];
7. **char** subject[100];
8. **int** book\_id;
9. };
11. /\* function declaration \*/
12. **void** printBook( **struct** Books \*book );
13. **int** main( ) {
15. **struct** Books Book1; /\* Declare Book1 of type Book \*/
16. **struct** Books Book2; /\* Declare Book2 of type Book \*/
18. /\* book 1 specification \*/
19. strcpy( Book1.title, "C Programming");
20. strcpy( Book1.author, "Nuha Ali");
21. strcpy( Book1.subject, "C Programming Tutorial");
22. Book1.book\_id = 6495407;
24. /\* book 2 specification \*/
25. strcpy( Book2.title, "Telecom Billing");
26. strcpy( Book2.author, "Zara Ali");
27. strcpy( Book2.subject, "Telecom Billing Tutorial");
28. Book2.book\_id = 6495700;
30. /\* print Book1 info by passing address of Book1 \*/
31. printBook( &Book1 );
33. /\* print Book2 info by passing address of Book2 \*/
34. printBook( &Book2 );
36. **return** 0;
37. }
39. **void** printBook( **struct** Books \*book ) {
41. printf( "Book title : %s\n", book->title);
42. printf( "Book author : %s\n", book->author);
43. printf( "Book subject : %s\n", book->subject);
44. printf( "Book book\_id : %d\n", book->book\_id);
45. }

When the above code is compiled and executed, it produces the following result −

1. Book title : C Programming
2. Book author : Nuha Ali
3. Book subject : C Programming Tutorial
4. Book book\_id : 6495407
5. Book title : Telecom Billing
6. Book author : Zara Ali
7. Book subject : Telecom Billing Tutorial
8. Book book\_id : 6495700

**Bit Fields**

Bit Fields allow the packing of data in a structure. This is especially useful when memory or data storage is at a premium. Typical examples include −

* Packing several objects into a machine word. e.g. 1 bit flags can be compacted.
* Reading external file formats -- non-standard file formats could be read in, e.g., 9-bit integers.

C allows us to do this in a structure definition by putting :bit length after the variable. For example −

1. **struct** packed\_struct {
2. **unsigned** **int** f1:1;
3. **unsigned** **int** f2:1;
4. **unsigned** **int** f3:1;
5. **unsigned** **int** f4:1;
6. **unsigned** **int** type:4;
7. **unsigned** **int** my\_int:9;
8. } pack;

Here, the packed\_struct contains 6 members: Four 1 bit flags f1..f3, a 4-bit type and a 9-bit my\_int.

C automatically packs the above bit fields as compactly as possible, provided that the maximum length of the field is less than or equal to the integer word length of the computer. If this is not the case, then some compilers may allow memory overlap for the fields while others would store the next field in the next word.

2.6k views · [View 13 Upvoters](https://www.quora.com/What-is-the-structure-of-a-C-program)

[](https://www.quora.com/profile/Maya-Thakkar-1)

[Maya Thakkar](https://www.quora.com/profile/Maya-Thakkar-1), Product Development Manager at LearnVern (2017-present)

[Answered Feb 17, 2018](https://www.quora.com/What-is-the-structure-of-a-C-program/answer/Maya-Thakkar-1) · Author has **499** answers and **220k** answer views

Originally Answered: [What are structures using C?](https://www.quora.com/What-are-structures-using-C?no_redirect=1)

Structure using C:

* Before we start structure of C program first know the software to create the program.
* We will use turbo C software and that is available free of cost.
* The extension of C program is .c
* Now lets create very first program and also understand each of line.

If you want to learn more information about structure of C, so I can share video tutorial link with example.

[**Structure Using C**](https://goo.gl/X1a4GF)

1.7k views · [View 1 Upvoter](https://www.quora.com/What-is-the-structure-of-a-C-program)

[](https://www.quora.com/profile/Ian-Heggie)

[Ian Heggie](https://www.quora.com/profile/Ian-Heggie), BSc Computer Science, University of Melbourne (1981)

[Answered Sep 25, 2017](https://www.quora.com/What-is-the-structure-of-a-C-program/answer/Ian-Heggie) · Author has **1.4k** answers and **706.9k** answer views

There are a lot of good answers that tell of what the (big picture) **conventional** overall structure of a C program is, but it must be recognised that these are conventions to make it easier for people to read the code, rather than being a requirement of the language itself. An exhaustive structure guide will come in around 100 pages or so.

The only thing that is really required by the language is that you define/include things before you use them - source code is parsed top to bottom without backward references. (see [The International Obfuscated C Code Contest](http://www.ioccc.org/) for projects that “break” all the conventions but are still valid C programs).

In practise the best structure to use when adding code to an existing project**is the one already in use!** Likewise the best structure to use when creating a new project is the structure that the rest of the present and likely future programmers will expect to see. If you are totally new to the field, then starting with the standards recommended with the gnu compiler is a reasonable place to start. ( [GNU Coding Standards: Writing C](https://www.gnu.org/prep/standards/html_node/Writing-C.html" \t "_blank) ). My own personal preference is slightly different, but if I am adding to an existing code base, I adjust my coding style to match to aid comprehension.

Actually C Program structure divided in to several sections.

They are,

* **Documentations (Documentation Section)**  
  **->** The Documentation Section consists of a set of comment lines giving the name of the program and other details. The Link Section provides instructions to the compiler to link functions from the system library.
* **Preprocessor Statements (Link Section)**  
  **->** C Preprocessor directives: Before a C program is compiled in a compiler, source code is processed by aprogram called preprocessor. This process is calledpreprocessing. Commands used in preprocessor are called preprocessor directives and they begin with “#” symbol.
* **Global Declarations (Definition Section)**  
  **->**Global variables hold their values throughout the lifetime of your program and they can be accessed inside any of the functions defined for the program. A global variable can be accessed by any function. That is, a global variable is available for use throughout your entire program after its declaration.
* **The main() function**  
  **->**In C, the "main" function is treated the same as everyfunction, it has a return type (and in some cases accepts inputs via parameters). The only difference is that the main function is "called" by the operating system when the user runs the program.
  + **Local Declarations**  
    **->**A scope in any programming is a region of the program where a defined variable can have its existence and beyond that variable it cannot be accessed. There are three places where variables can be declared in C programming language − Inside a function or a block which is called local variables.
  + **Program Statements & Expressions**  
    **->**An expression statement consists of an optional expression followed by a semicolon (;). If the expression is present, the statement may have a value. If no expression is present, the statement is often called the null statement.

The printf function calls are expressions, so statements such as printf ("Hello World!\n"); are expression statements.

* User Defined Functions  
  **->**A function is a block of code that performs a specific task. Callows you to define functions according to your need. These functions are known as user-defined functions. ... You can create two functions to solve this problem: createCircle() function.

Example:

Description: Writes the words "Hello, World!" on the screen \*/

#include<stdio.h>

int main()

{

printf("Hello, World!\n");

return 0;

}

Structure of C program:

* A struct in C programming language is a composite data type declaration that defines a physically grouped list of variables to be placed under one name in a block of memory, allowing the different variables to be accessed via a single pointer, or the struct declared name which returns the same address. The struct can contain many other complex and simple data types in an association, so is a natural organizing type for records like the mixed data types in lists of directory entries reading a hard drive (file length, name, extension, physical or other mixed record type

A structure is a collection of related elements, possibly of different types, having a single name.

C has two ways to define a structure:

tagged structure and type-defined structures.

TAGGED STRUCTURE:

The structure definition associated with the structure name is referred as tagged structure.

It does not create an instance of a structure and does not allocate any memory.

The general form or syntax of tagged structure definition is as follows,

struct tag\_name

{

type1 member1;

type2 member2;

……………

  };

where struct is the keyword which tells the compiler that a structure is being defined, tag\_name is the name of the structure and member1, member2 … are called members of the structure.

The members are declared within curly braces.

The closing brace must end with the semicolon.

TYPEDEFINED STRUCTURE:

The structure definition associated with keyword typedef is called type-defined structure. This is the most powerful way of defining the structure.

The syntax of typedefined structure is:

typedef struct

{

type1 member1;

type2 member2;

……

} TYPE\_ID;

Where, typedef is keyword added to the beginning of the definition, struct is the keyword which tells the compiler that a structure is being defined and member1, member2…are called fields of the structure.

The closing brace must end with type definition name which in turn ends with semicolon.

2.3k views · [View 5 Upvoters](https://www.quora.com/What-is-the-structure-of-a-C-program)



Anonymous

[Answered Nov 27, 2014](https://www.quora.com/What-is-the-structure-of-a-C-program/answers/8291094)

Documentation Section Link Section Definition Section Global Declaration Section main(){Declaration Section Executable part}Subprogram section Function 1Function 2. .**function** n  
**The Documentation Section** consists of a set of comment lines giving the name of the program and other details.  
**The Link Section** provides instructions to the compiler to link functions from the system library.   
**The Definition Section** defines all symbolic constants.  
**The Global Declaration Section:** There are some variables and those variables are declared in this section that is outside of all functions.   
**main() function:** Every C program must have one main function section. This section contains two parts, declaration and executable part.  
**Declaration Part** declares all the variables used in the executable part.  
 There should be at least one statement in the **executable part** which contains instructions to perform certain task.  
 The declaration and executable part must appear between the opening and closing braces. All statements in the declaration part should end with the semicolon.  
The **Subprogram Section** contains all the user defined functions that are called in the main function.

5.8k views · [View 2 Upvoters](https://www.quora.com/What-is-the-structure-of-a-C-program)

[](https://www.quora.com/profile/Rohit-Hegde-10)

[Rohit Hegde](https://www.quora.com/profile/Rohit-Hegde-10), learner

[Answered Dec 10, 2016](https://www.quora.com/What-is-the-structure-of-a-C-program/answer/Rohit-Hegde-10)

Originally Answered: [What is a structure in C language?](https://www.quora.com/What-is-a-structure-in-C-language?no_redirect=1)

Suppose you have 100 chocolates and you want to hide it from your friends so that they wont be able to find it and eat it. So you take those chocolates and hide it in a box!

Now this **box**is your **Structure.** And the chocolates in that box are your **Variables.**

Now read this para if you want to know what variables are: Variables are memory spaces where we store any kind of data. Suppose I want to store an integer value to in my computer. (any number which does not have decimal places is called an integer) So I will 1st tell the computer to reserve a place in the memory to store this value. This is done by this statement: **int** x; This is called declaring a variable. Here **int** denotes integer and **x** is the name of the reserved memory space. Semi-colon is given to indicate end of a command. x = 10; this means store 10 in the memory space named x. So here **x** is called as the **variable**. Variable can have any name (except keywords) like x, y, abc, your\_name, olala, bachchan, anything.

Now we need to know that the variables declared inside the structure are called as **members**of that structure.

The basic syntax/example for a structure is:

1. **struct** structure\_name
2. {
3. **int** abc;
4. **char** ch;
5. **int** arr[10];
6. }v1, v2;

Now here **abc, ch, arr[]**are the members of the structure.

Now to use this structure we need to declare its variables. In the above example, **v1**and **v2**are the 2 variables of the structure. **Now comes the most important part:**What are the variables of a structure (remember the variables that you are thinking of are called members of structure, I am talking about variables **of**a structure and not variables **in**the structure). To understand these variables consider the variables of the structure as *xerox copies* of the structure which can be used. Now **v1** is a xerox copy of the entire structure code and **v2** is another xerox copy of the entire structure code. So now we have 2 copies of the structure. These variables can now be used to access the members **abc, ch, arr[].**

To access the members we need to use the code: v1.abc=40; This means that assign value 40 to the copy of member **abc** with variable **v1**. v2.abc=80; This means that assign value 80 to the copy of member **abc**with variable **v2**. Now there are 2 copies of **abc**with 2 different values i.e. 40 and 80.

I hope I helped you in understanding what structures are.

For any further doubt message or comment below.

Cheers! :)

683 views · [View 1 Upvoter](https://www.quora.com/What-is-the-structure-of-a-C-program) · [View Sharers](https://www.quora.com/What-is-the-structure-of-a-C-program)

[](https://www.quora.com/profile/Rishu-Verma-11)

[Rishu Verma](https://www.quora.com/profile/Rishu-Verma-11), B.Tech Computer Science, Jaypee University of Information Technology,Waknaghat(Solan) (2019)

[Answered Jan 9, 2017](https://www.quora.com/What-is-the-structure-of-a-C-program/answer/Rishu-Verma-11)

Originally Answered: [What is a structure in C language?](https://www.quora.com/What-is-a-structure-in-C-language?no_redirect=1)

Structure is a collection of variables of different types under a single name.

**For example:**You want to store some information about a person: his/her name, citizenship number and salary. You can easily create different variables name, citNo, salary to store these information separately.

However, in the future, you would want to store information about multiple persons. Now, you'd need to create different variables for each information per person (person 1, 2, 3 and so on . .) as :

name1, citNo1, salary1, name2, citNo2, salary2

You can easily visualize how big and messy the code would look. Also, since no relation between the variables (information) would exist, it's going to be a daunting task.

A better approach will be to have a collection of all related information under a single name Person, and use it for every person. Now, the code will look much cleaner, readable and efficient as well.

This collection of all related information under a single name Person is a structure.

**Keyword struct** is used for creating a structure.

**Syntax of structure**

1. **struct** structure\_name
2. {
3. data\_type member1;
4. data\_type member2;
5. .
6. .
7. data\_type memeber;
8. };

**Note**: Don't forget the semicolon }; in the ending line.

We can create the structure for a person as mentioned above as:

1. **struct** person
2. {
3. **char** name[50];
4. **int** citNo;
5. **float** salary;
6. };

This declaration above creates the derived data type **struct** person.

When a structure is defined, it creates a user-defined type but, no storage or memory is allocated.

For the above structure of a person, variable can be declared as:

1. **struct** person
2. {
3. **char** name[50];
4. **int** citNo;
5. **float** salary;
6. };
7. **int** main()
8. {
9. **struct** person person1, person2, person3[20];
10. **return** 0;
11. }

Another way of creating a structure variable is:

1. **struct** person
2. {
3. **char** name[50];
4. **int** citNo;
5. **float** salary;
6. } person1, person2, person3[20];

In both cases, two variables person1, person2 and an array person3 having 20 elements of type **struct person** .

Hope this will work , for any queries feel free to ask!

939 views · Answer requested by [Rachit Mathur](https://www.quora.com/profile/Rachit-Mathur-27)

[](https://www.quora.com/profile/Prajwal-Dcunha)

[Prajwal Dcunha](https://www.quora.com/profile/Prajwal-Dcunha), knows English

[Answered Apr 13, 2017](https://www.quora.com/What-is-the-structure-of-a-C-program/answer/Prajwal-Dcunha) · Author has **254** answers and **554.7k** answer views

Originally Answered: [What is a C language structure?](https://www.quora.com/What-is-a-C-language-structure?no_redirect=1)

To define a structure, you must use the **struct** statement. The struct statement defines a new data type, with more than one member. The format of the struct statement is as follows −

struct [structure tag] {  
  
 member definition;  
 member definition;  
 ...  
 member definition;  
} [one or more structure variables];

The **structure tag** is optional and each member definition is a normal variable definition, such as int i; or float f; or any other valid variable definition. At the end of the structure's definition, before the final semicolon, you can specify one or more structure variables but it is optional. Here is the way you would declare the Book structure :

struct Books {  
 char title[50];  
 char author[50];  
 char subject[100];  
 int book\_id;  
} book;

644 views · [View 3 Upvoters](https://www.quora.com/What-is-the-structure-of-a-C-program)

[](https://www.quora.com/profile/Rajnish-Kumar-1623)

[Rajnish Kumar](https://www.quora.com/profile/Rajnish-Kumar-1623), Content Writer

[Answered Jul 22, 2018](https://www.quora.com/What-is-the-structure-of-a-C-program/answer/Rajnish-Kumar-1623) · Author has **313** answers and **35.4k** answer views

Originally Answered: [What is a structure in C language?](https://www.quora.com/What-is-a-structure-in-C-language?no_redirect=1)

A structure is data type which collect different atoms of the information that comprise a given entity and structure is also known as the collection of heterogeneous element . Structure is mainly used in the offices , schools . A structure contains a number of data types grouped together . These data type may be same or different .

EXAMPLE :  
 if you want to store data about the student then there information is like name , age , roll no , fees , class etc then there are uses of different data structure like string , int , float , char so for that we have use these different methods.

1 . use a structure variable.  
2 . you may use array for each different variables.

Read full article :; [Structures in C briefly explain with program of structure](https://www.programmingponds.com/2018/07/structures-in-c.html)

534 views

[](https://www.quora.com/profile/Ankush-Nagpal-9)

[Ankush Nagpal](https://www.quora.com/profile/Ankush-Nagpal-9), My mother tongue :P

[Updated Dec 17, 2016](https://www.quora.com/What-is-the-structure-of-a-C-program/answer/Ankush-Nagpal-9)

Originally Answered: [What is a structure in C language?](https://www.quora.com/What-is-a-structure-in-C-language?no_redirect=1)

Structure in C is a complex data type declaration where you can group certain data types under one single structure and call it using one pointer.Unlike arrays data types can be different.

example:

struct employee{

int salary;

char name[30];

};

Here employee is a data type.

int main(){

struct employee e1,e2; //variable declaration where e1 & e2 are variables of type employee

}

After declaring e1 and e2, 68 bytes are allocated in the memory. 34 for e1 and 34 for e2. (Assuming int is of size 4 bytes and name[30] takes 30 bytes.)

779 views · [View 1 Upvoter](https://www.quora.com/What-is-the-structure-of-a-C-program)

[](https://www.quora.com/profile/Birendra-Kumar-259)

[Birendra Kumar](https://www.quora.com/profile/Birendra-Kumar-259), mca Master of Computer Applications & Computer Programming, IIMT College of Engineering, Greater Noida (2018)

[Answered Oct 5, 2018](https://www.quora.com/What-is-the-structure-of-a-C-program/answer/Birendra-Kumar-259)

C program is a collection of ane or more functions. if a C program has only one function them it must be the main function becouse, C program stsrts with a main function and its execution end with main function.

=> main function is invoked by operating system.

=>If a C program does’t have a mian function then a program is compiled but is not executed.

=> Execution of a function in C program.

Syntex to a C program structure.

//comment

preprocesser directive

void main()

{

Local variable declaration;

statement;

}

188 views

[](https://www.quora.com/profile/Learn-Vern-3)

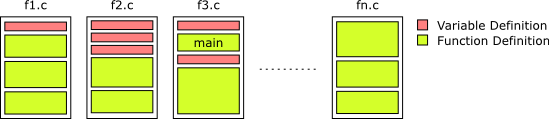
[Learn Vern](https://www.quora.com/profile/Learn-Vern-3)

[Answered Mar 7, 2018](https://www.quora.com/What-is-the-structure-of-a-C-program/answer/Learn-Vern-3) · Author has **253** answers and **98.8k** answer views

Nowadays, there exist multiple programming languages called “high level”, which are those that allow to write programs significantly different to the assembly code or machine language, which is the one used by the processor to execute all the programs. Java is an“object-oriented language” high level because programs are organized around a set of classes and objects. C, however, does not have any type of object, and the programs are simply organized as a set of functions, thus it is “a procedural language”.

Java is a language with a rich functionality, and although it shares a significant portion of this functionality with Java, the differences are greater than the similarities. The code structures in C are much *simpler* that those in Java. In mechanisms such as loops, conditionals, etc, both languages are similar, but in the remaining aspects such as data structure definition, C offers simpler mechanisms.

A C program is a set of functions, data type definitions and variable declarations contained in a set of files. A C program always start its execution by the function with name main. Any function can invoke any other function and the variables declared outside the function are either global or local to the current file (if they are declared with the **static** prefix). The following figure shows the structure of a C program contained in several files.



The C compiler is the program that translates a set of functions, definitions and declarations in multiple files into an executable file. The C compiler has a surprisingly simple behavior and performs much less work than expected when compared with others such as the Java compiler. To create an executable, the compiler processes the source files one by one independently. This means that the defined variables and functions *are not remembered when processing another file*. Furthermore, the compiler performs a single pass over the text, only those definitions up to the current compilation point are visible.

To Know More With Free Videos Visit: [**Structure of C Programming**](https://goo.gl/MFudBL)

3.8k views · [View 1 Upvoter](https://www.quora.com/What-is-the-structure-of-a-C-program)

[](https://www.quora.com/profile/Michael-Bauers)

[Michael Bauers](https://www.quora.com/profile/Michael-Bauers), worked at Viavi Solutions (2015-2017)

[Answered May 5, 2018](https://www.quora.com/What-is-the-structure-of-a-C-program/answer/Michael-Bauers) · Author has **3.2k** answers and **734.3k** answer views

Originally Answered: [What is the structure of a C program?](https://www.quora.com/What-is-the-structure-of-a-C-program-1?no_redirect=1)

If you want the grammar, this seems complete, but I didn’t check it - [ANSI C grammar (Yacc)](http://www.quut.com/c/ANSI-C-grammar-y.html)

If you just want the high level view, C files ( aka translation units) contain statements. Statements allowed outside of functions are generally declarations and definitions for types, variables and functions.

Inside of a function, you can define local variables, and use statements allowed in functions like if, for, while, switch, and expressions ( not an exhaustive list.)

Because C is preprocessed, you can use a handful of preprocessor macros, but note this is not technically part of the C language; they are part of the C preprocessor.

Just to make something clear, header files are not generally translation units. The preprocessor handles “bringing in” the header file into your source file, as if you had typed it all into the source file. They should be viewed as part of the translation unit they were included in. Whether the statements in the header will compile or not, is dependent on the all the other code in the translation unit they were brought into.

1.1k views · [View 1 Upvoter](https://www.quora.com/What-is-the-structure-of-a-C-program)

[](https://www.quora.com/profile/Siddhant-Pathak-24)

[Siddhant Pathak](https://www.quora.com/profile/Siddhant-Pathak-24), studied Computer Science & Mathematics at Delhi Public School, Bhilai

[Answered Nov 15, 2017](https://www.quora.com/What-is-the-structure-of-a-C-program/answer/Siddhant-Pathak-24)

Originally Answered: [What are structures using C?](https://www.quora.com/What-are-structures-using-C?no_redirect=1)

Structures, in this context refer to data structures. Data Structures are different ways in which data is stored. C has some data structures which are defined by default, like integer and character. So, a variable which needs to store data of these types can be declared using keywords for the same like int, or char. However, if one wished to use a variable to store some otger forms of data, one can define their own data structures using the struct command. An example of a use of the same would be if one wished to store dates, which would not be one, but three integers. One could use the struct keyword to construct a new blueprint for variables which store three different variables that are all integers each, and have their own specific properties and values.

234 views

[](https://www.quora.com/profile/Shankhanil-Ghosh)

[Shankhanil Ghosh](https://www.quora.com/profile/Shankhanil-Ghosh), BTech Computer Science and Engineering, University of Calcutta (2020)

[Answered Dec 9, 2016](https://www.quora.com/What-is-the-structure-of-a-C-program/answer/Shankhanil-Ghosh) · Author has **84** answers and **55.2k** answer views

Originally Answered: [What is a structure in C language?](https://www.quora.com/What-is-a-structure-in-C-language?no_redirect=1)

A structure can be defined as an user-defined data type. It is an identity which can store one or more data, only that the type of data to be stored is to be decided by the coder.

The general syntax of defining a structure is

1. **struct** struct\_name{
3. //primitive data type variables to be stored
5. } object\_name\_of\_the\_struct

Though C is not an object oriented language, still the concept of structures introduce the idea of object orientation. Structures basically brings one or many primitive data variables under one single unit, called the object.

Let’s take up some examples:

1. **struct** student{
2. **char** name[100];
3. **char** roll[10];
4. **char** grade;
5. } s1, s2;

Here, three variables names name[], roll[], grade are compacted in one single unit calld student, which has 2 objects names s1 and s2.

Hope that helps

656 views · [View 1 Upvoter](https://www.quora.com/What-is-the-structure-of-a-C-program)

[](https://www.quora.com/profile/Simran-489)

[Simran](https://www.quora.com/profile/Simran-489), B.Sc Computer Science (2017)

[Answered Mar 18, 2019](https://www.quora.com/What-is-the-structure-of-a-C-program/answer/Simran-489)

The structure of a C program is as follows:

**\*Documentation Section**

-It is the comment section

-It makes the program readable

**\*Preprocessor Directive Section**

-Preprocessor Directive is a message to the compiler

-It has further two sections: ^ Link Section ^Definition Section

**\*Global Declaration Section**

-It contains declaration of global variables

**\*Main Program Section**

-It consists of local variable declaration

-It also comprises executable and non executable code

**\*Sub Program Section**

-It contains definition of user defined functions

All of the above are thoroughly explained with examples in the below link!

Enjoy learning!

81 views · Answer requested by [Vikas Sunsna](https://www.quora.com/profile/Vikas-Sunsna)

[](https://www.quora.com/profile/Alex-Lord-9)

[Alex Lord](https://www.quora.com/profile/Alex-Lord-9), BSc Computing, University of the West of England (2020)

[Answered Dec 14, 2017](https://www.quora.com/What-is-the-structure-of-a-C-program/answer/Alex-Lord-9)

Originally Answered: [What is the basic structure of a C program?](https://www.quora.com/What-is-the-basic-structure-of-a-C-program-1?no_redirect=1)

You could have done a simple Google search to find this out yourself but I’ll answer it anyway.

1. #include <stdio.h>
3. **int** main()
4. {
5. /\* Your code goes here \*/
7. **return** 0;
8. }

This is the simplest, most bare-bones C program you can write.

Hopefully this code makes sense to you.

1.3k views · [View 2 Upvoters](https://www.quora.com/What-is-the-structure-of-a-C-program)

[](https://www.quora.com/profile/Shivangi-Srivastava-189)

[Shivangi Srivastava](https://www.quora.com/profile/Shivangi-Srivastava-189), B.Tech from Uttarakhand Technical University (2021)

[Answered Jul 29, 2018](https://www.quora.com/What-is-the-structure-of-a-C-program/answer/Shivangi-Srivastava-189)

Originally Answered: [What is the structure of a C program?](https://www.quora.com/What-is-the-structure-of-a-C-program-2?no_redirect=1)

Structure is a collection of variables of different types under a single name.

example-You want to store some information about a person: his/her name, citizenship number and salary. You can easily create different variables

name, citNo, salary

to store these information separately.

However, in the future, you would want to store information about multiple persons. Now, you'd need to create different variables for each information per person:

name1, citNo1, salary1, name2, citNo2, salary2

176 views · [View 1 Upvoter](https://www.quora.com/What-is-the-structure-of-a-C-program)

[](https://www.quora.com/profile/Chandni-Gowda)

[Chandni Gowda](https://www.quora.com/profile/Chandni-Gowda), studied at Gujarat Technological University

[Answered Jul 20, 2018](https://www.quora.com/What-is-the-structure-of-a-C-program/answer/Chandni-Gowda)

Originally Answered: [What is a structure in C language?](https://www.quora.com/What-is-a-structure-in-C-language?no_redirect=1)

→Whenever we need to group some data items of same data types we are using a data structure called an Array.

→But to group some number of data items of different data type in C,a data structure called **Struct** is useful.

→Structure is an user-defined data type.

→It’s kind of grouping of data member of different data type.

→Data items stored in Struct are known as **Members** of structure.

→Memory allocation of these members in structure is Contiguous.

→We have to define Struct before we can use it.

**CODE**

Struct

{

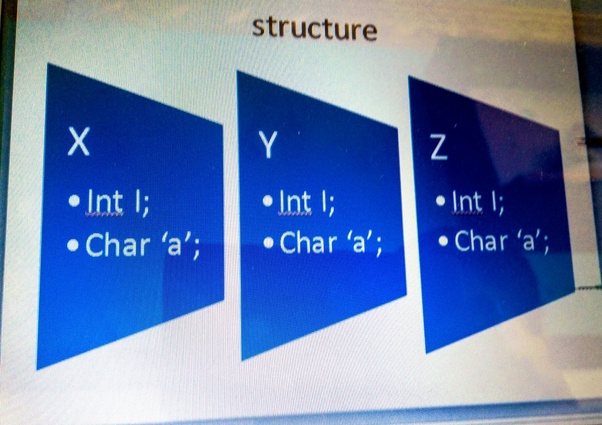
int i;

Char c='a';

} x,y,z;

int x,y,z;

→Here we defined three structures X Y and Z with same structure type



→ To initialize and access any one structure

→to access any member of structure we have to use .(dot).

**CODE**

//syntax of Intialization

z.i=10;

//syntax to Access character

z.c;

483 views

[](https://www.quora.com/profile/Arogya-Thapa-Magar)

[Arogya Thapa Magar](https://www.quora.com/profile/Arogya-Thapa-Magar), BSc. CSIT Programming Language & Cricket, Asian College of Higher Studies (2021)

[Answered Feb 25, 2018](https://www.quora.com/What-is-the-structure-of-a-C-program/answer/Arogya-Thapa-Magar) · Author has **80** answers and **15.9k** answer views

Originally Answered: [What is a structure in C language?](https://www.quora.com/What-is-a-structure-in-C-language?no_redirect=1)

[What is Structure?](https://thecompletecodes.blogspot.com/2018/02/structure-arrays-are-used-to-store.html)

38 views

[](https://www.quora.com/profile/Surbhi-Joshi-28)

[Surbhi Joshi](https://www.quora.com/profile/Surbhi-Joshi-28), Trainer at LearnVern (2016-present)

[Answered Mar 3, 2018](https://www.quora.com/What-is-the-structure-of-a-C-program/answer/Surbhi-Joshi-28) · Author has **231** answers and **85.1k** answer views

Structure of C programming:

* Before we start the structure of C program first know the software to create the program.
* We will use Turbo C software and that is available free of cost.
* The extension of C program is .c

Create a program and understand, so please visit this site.

[**Structure of C program**](https://goo.gl/X1a4GF)

507 views

[](https://www.quora.com/profile/Stefan-Lindstr%C3%B6m)

[Stefan Lindström](https://www.quora.com/profile/Stefan-Lindstr%C3%B6m), Software Engineer at BnearIT AB (2015-present)

[Answered Feb 10, 2016](https://www.quora.com/What-is-the-structure-of-a-C-program/answer/Stefan-Lindstr%C3%B6m) · Author has **388** answers and **295k** answer views

Originally Answered: [What are the different types of structures in C language?](https://www.quora.com/What-are-the-different-types-of-structures-in-C-language?no_redirect=1)

I'm not sure this is what you mean, but anyways.

There'are a few ways you can lump data together.

Array: Lump together one or more items of a specified type in one contigious block of memory, e.g: int x[10]; struct foo[4]; and so on.

Struct: Lump together one, or more items, which can have different types into a contigious block of memory. Padding might be inserted into the struct to maintain alignment requirementsm ex:

struct foo

{

  char bar;

  int  herp;

  char derp;

  short the\_larch;

};

You can now more easily access items in your struct, e.g; struct foo aFoo; aFoo.herp = 42; On a platform with strict alignment requirements, the struct will contain padding to enforce alignment.

Union, as a struct, except the members of the union will share the same memory, and the union will never occupy more memory than the largest member, eg:

union bar

{

  struct foo x

  int y;

  float z;

  struct xyzzy w;

}

now sizeof(union bar) = max(sizeof(struct foo), sizeof(int), sizeof(z), sizeof(struct xyzzy)); So, union bar t; t.x.bar = 'p'; t.x.herp = 42; So, wheras a struct contains all its members, a union will only hold one of the members at a time.

Note however, it's unspecified behaviour to read a union member other than the last one stored into, e.g: uniton bar t; t.z = 18.273; int x = t.y; // unspecifified.

Of course, all these can be combined. You can have a strucy contating arrays, other structs and unions which in turn may containt arrays and other unions and structs and so on,

I hope this helps.

1k views · [View 1 Upvoter](https://www.quora.com/What-is-the-structure-of-a-C-program) · Answer requested by [Chris Fernandes](https://www.quora.com/profile/Chris-Fernandes-62)

[](https://www.quora.com/profile/Dhruv-Goel-48)

[Dhruv Goel](https://www.quora.com/profile/Dhruv-Goel-48), Associate Software Developer at OrionCoders (2017-present)

[Answered Jul 16, 2017](https://www.quora.com/What-is-the-structure-of-a-C-program/answer/Dhruv-Goel-48)

Originally Answered: [What is structure in C language?](https://www.quora.com/What-is-structure-in-C-language?no_redirect=1)

Structure is used to make abstract data types (ADT). ADT is a user defined data type that can have member functions and member variables according to the requirements of users. It is similar to a class. The only difference is that structure has everything in public.

For example,

struct trial{

int x;

char a;

};

Now the above ADT trial can have its objects as:

trial t1, t2;

207 views

[](https://www.quora.com/profile/Vinay-Chandel-15)

[Vinay Chandel](https://www.quora.com/profile/Vinay-Chandel-15), B.Tech CSE from Banasthali University (2020)

[Answered May 5, 2018](https://www.quora.com/What-is-the-structure-of-a-C-program/answer/Vinay-Chandel-15)

The following is the general structure of a C program.

DOCUMENTATION SECTION

INCLUSION SECTION

DEFINITION SECTION

GLOBAL VARIABLE DECLARATION

int main(void)

{

LOCAL VARIABLE DECLARATION

EXECUTABLE STATEMENTS

return 0;

}

Function 1( )

{

}

Function 2( )

{

}

111 views

[Ramadevi Chilukuri](https://www.quora.com/profile/Ramadevi-Chilukuri)

[Ramadevi Chilukuri](https://www.quora.com/profile/Ramadevi-Chilukuri), studied Project Management at Project Management Institute

[Answered Nov 17, 2017](https://www.quora.com/What-is-the-structure-of-a-C-program/answer/Ramadevi-Chilukuri) · Author has **253** answers and **80.7k** answer views

Originally Answered: [What are structures using C?](https://www.quora.com/What-are-structures-using-C?no_redirect=1)

A structure is holder of some properties with values.

You can consider a structure as a basis of class in c++ and java.

129 views

[](https://www.quora.com/profile/Kodipalli-Vaishnavi)

[Kodipalli Vaishnavi](https://www.quora.com/profile/Kodipalli-Vaishnavi), c,c++,embedded c

[Answered Jan 4, 2017](https://www.quora.com/What-is-the-structure-of-a-C-program/answer/Kodipalli-Vaishnavi)

Originally Answered: [What is a structure in C language?](https://www.quora.com/What-is-a-structure-in-C-language?no_redirect=1)

A structure is a user defined data type in C/C++. A structure creates a data type that can be used to group items of possibly different types into a single type.

***How to create a structure?***  
‘struct’ keyword is used to create a structure. Following is an example.

**struct** addrress

{

**char** name[50];

**char** street[100];

**char** city[50];

**char** state[20]

**int** pin;

};

297 views · [View 1 Upvoter](https://www.quora.com/What-is-the-structure-of-a-C-program)

[](https://www.quora.com/profile/Karan-Vachhani)

[Karan Vachhani](https://www.quora.com/profile/Karan-Vachhani), works at Drug and Pharmaceutical Companies (2016-present)

[Answered Jul 29, 2018](https://www.quora.com/What-is-the-structure-of-a-C-program/answer/Karan-Vachhani)

Originally Answered: [What is the structure of a C program?](https://www.quora.com/What-is-the-structure-of-a-C-program-2?no_redirect=1)

1. First all it comes with documents section. (Optional)
2. Link section (e.g. headear file #include<conio.h>)
3. Than if there is global declaration (e.g. inter or function which we're going to use in global program)
4. Than it start with main section { e.g. int main() }
5. In the main section you will have ..
   1. Local declaration
   2. Other program statements & expressions (e.g. printf , scanf , or other predefined fuctions etc )
6. At the end you will have user define function
7. And its ends with ending tag :)

880 views · [View 1 Upvoter](https://www.quora.com/What-is-the-structure-of-a-C-program)

[](https://www.quora.com/profile/Sujib-Kumar-Mondal)

[Sujib Kumar Mondal](https://www.quora.com/profile/Sujib-Kumar-Mondal), Private Tutor at Computer Programming Languages (2017-present)

[Answered Nov 18, 2017](https://www.quora.com/What-is-the-structure-of-a-C-program/answer/Sujib-Kumar-Mondal)

Originally Answered: [What are structures using C?](https://www.quora.com/What-are-structures-using-C?no_redirect=1)

Structure is a process, to make user defined Data-Types. It looks a little similar like creation of “Class” in OOP and creation of “Function”.

One can use this Data-Type to initialize variables. And later on, those variables will have the properties of that Data-Type.

150 views

[](https://www.quora.com/profile/Learn-Vern-7)

[Learn Vern](https://www.quora.com/profile/Learn-Vern-7)

[Answered Mar 24, 2018](https://www.quora.com/What-is-the-structure-of-a-C-program/answer/Learn-Vern-7) · Author has **224** answers and **162k** answer views

Originally Answered: [What is a structure in C language?](https://www.quora.com/What-is-a-structure-in-C-language?no_redirect=1)

A **struct** in the **C programming language** (and many derivatives) is a composite data type (or record) declaration that defines a physically grouped list of variables to be placed under one name in a block of memory, allowing the different variables to be accessed via a single pointer,

To Know more with free videos visit Learnvern

171 views

[Shalini Pandya](https://www.quora.com/profile/Shalini-Pandya-1)

[Shalini Pandya](https://www.quora.com/profile/Shalini-Pandya-1)

[Answered Apr 30, 2018](https://www.quora.com/What-is-the-structure-of-a-C-program/answer/Shalini-Pandya-1) · Author has **152** answers and **15.5k** answer views

tructure of C program:

* Before we start structure of C program first know the software to create the program.
* We will use Turbo c software and that is available free of cost.
* The extension of C program is .c

Learn full course of C [Programming](https://www.tops-int.com/c-cplus/)

219 views

[](https://www.quora.com/profile/Konduru-Vijayalakshmi-1)

[Konduru Vijayalakshmi](https://www.quora.com/profile/Konduru-Vijayalakshmi-1), Software Test Engineer at Wipro (2014-present)

[Answered Aug 1, 2017](https://www.quora.com/What-is-the-structure-of-a-C-program/answer/Konduru-Vijayalakshmi-1) · Author has **363** answers and **836.2k** answer views

[**Structure of C program**](http://www.gcreddy.com/2015/06/overview-of-c-language.html)

• Documentation Section //optional  
• Link section //optional  
• Defining section //optional  
• Global declaration section //optional  
• Main function section //Must   
{   
Declaration part   
Executable part.  
}  
• Sub program section //optional  
• Function 1  
• Function 2  
• Function n

3.6k views · [View 1 Upvoter](https://www.quora.com/What-is-the-structure-of-a-C-program)

[](https://www.quora.com/profile/Gopinath-Vikkram)

[Gopinath Vikkram](https://www.quora.com/profile/Gopinath-Vikkram)

[Answered Aug 1, 2018](https://www.quora.com/What-is-the-structure-of-a-C-program/answer/Gopinath-Vikkram)

Documentation section

Link section

Definition section

Global declaration section

Main() function section

{

Declaration section

Execution section

}

Subprogram

function 1

function 2

function n

27 views

[](https://www.quora.com/profile/Disha-Shah-272)

[Disha Shah](https://www.quora.com/profile/Disha-Shah-272), Trainer At LearnVern

[Answered Apr 4, 2018](https://www.quora.com/What-is-the-structure-of-a-C-program/answer/Disha-Shah-272) · Author has **202** answers and **44.9k** answer views

Originally Answered: [In C programming, what is a structure?](https://www.quora.com/In-C-programming-what-is-a-structure?no_redirect=1)

A structure is a user defined data type in C/C++. A structure creates a data type that can be used to group items of possibly different types into a single type.

If you want to learn structure of C programming with example,

visit at learnverncom/unit/structure-of-c-programming

395 views

[](https://www.quora.com/profile/Venkata-Sai-Chowdary-Katepalli)

[Venkata Sai Chowdary Katepalli](https://www.quora.com/profile/Venkata-Sai-Chowdary-Katepalli), B.Tech C & Web Development, Prasad V. Potluri Siddhartha Institute of Technology (2018)

[Answered Nov 15, 2017](https://www.quora.com/What-is-the-structure-of-a-C-program/answer/Venkata-Sai-Chowdary-Katepalli)

Originally Answered: [What are structures using C?](https://www.quora.com/What-are-structures-using-C?no_redirect=1)

Structures are used to prepare a **skeleton**. If a student data has to be saved then structure will be useful to prepare a student details skeleton.

Ex : struct **student** {

Int rollno;

Char \*name;

} **Std1**;

Structure will create a structure and use by creating a structure variable and use.

**Student std2**;

Here we are seeing two variables std1, std2

We can use those by using '->' **std1->rollno**

208 views

[](https://www.quora.com/profile/Learn-Vern-1)

[Learn Vern](https://www.quora.com/profile/Learn-Vern-1)

[Answered May 9, 2018](https://www.quora.com/What-is-the-structure-of-a-C-program/answers/82690907) · Author has **587** answers and **205.4k** answer views

Originally Answered: [What is a structure in C language?](https://www.quora.com/What-is-a-structure-in-C-language?no_redirect=1)

A **struct** in the **C programming language** (and many derivatives) is a composite data type (or record) declaration that defines a physically grouped list of variables to be placed under one name in a block of memory, allowing the different variables to be accessed via a single pointer.

To know more visit Learnvern.

175 views

[Shruti Pandya](https://www.quora.com/profile/Shruti-Pandya-15)

[Shruti Pandya](https://www.quora.com/profile/Shruti-Pandya-15), Student From IIT - Bombay

[Answered Apr 7, 2018](https://www.quora.com/What-is-the-structure-of-a-C-program/answer/Shruti-Pandya-15) · Author has **142** answers and **49.6k** answer views

Structure in C with programming examples for beginners and professionals covering concepts, Declaring structure variable, Accessing members of structure, control statements, c array, c pointers, c structures, c union, c strings and more.

**Structure of a C program**

Structure is a user defined datatype which is a collection of variables of different datatype under a single name. It is similar to array but an array holds data of similar type only.