***What is a structure?***  
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.

**Syntax of struct**

struct structureName

{

dataType member1;

dataType member2;

...

};

Here is an example:

struct Person

{

char name[50];

int citNo;

float salary;

};

Here, a derived type struct Person is defined. Now, you can create variables of this type.

For example1 .let us consider an employee database to store id,email,date\_of birth,designation and salary for the all employees of an organization.for this we define a structure with the name employee to hold all the above mentioned information as follows-

struct employee

{

int id;

char email[20];

char date\_of\_birth[10];

char designation;

float salary;

};

Similarly,we can define a structure vehicle to store make,model,year,price etc.

struct vehicle

{

char make[10];

char model[10];

int year[10];

float price;

};

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ***How to declare structure variables?*** A structure variable can either be declared with structure declaration or as a separate declaration like basic types.  **Create struct variables**  When a struct type is declared, no storage or memory is allocated. To allocate memory of a given structure type and work with it, we need to create variables.  Here's how we create structure variables:  struct Person  {  char name[50];  int citNo;  float salary;  };  int main()  {  struct Person person1, person2, p[20];  return 0;  }  Another way of creating a struct variable is:  struct Person  {  char name[50];  int citNo;  float salary;  } person1, person2, p[20];  In both cases, two variables person1, person2, and an array variable p having 20 elements of type struct Person are created.   |  | | --- | | **// A variable declaration with structure declaration**.  struct Point  {     int x, y;  } p1;  // The variable p1 is declared with 'Point'  **// A variable declaration like basic data types**  struct Point  {     int x, y;  };    int main()  {     struct Point p1;  // The variable p1 is declared like a normal variable  } |   ***How to initialize structure members?*** Structure members **cannot be** initialized with declaration. For example the following C program fails in compilation.   |  | | --- | | struct Point  {     int x = 0;  // COMPILER ERROR:  cannot initialize members here     int y = 0;  // COMPILER ERROR:  cannot initialize members here  }; |   The reason for above error is simple, when a datatype is declared, no memory is allocated for it. Memory is allocated only when variables are created.  Structure members **can be** initialized using curly braces ‘{}’. For example, following is a valid initialization.   |  | | --- | | struct Point  {     int x, y;  };    int main()  {     // A valid initialization. member x gets value 0 and y  // gets value 1.  The order of declaration is followed.     struct Point p1 = {0, 1};  } |   ***How to access structure elements?*** Structure members are accessed using dot (.) operator.   |  | | --- | | #include<stdio.h>  struct Point  {     int x, y;  };    int main()  {     struct Point p1 = {0, 1};    // Accessing members of point p1     p1.x = 20;     printf ("x = %d, y = %d", p1.x, p1.y);      return 0;  } |   **Output:**  x = 20, y = 1  ***What is an array of structures?*** Like other primitive data types, we can create an array of structures.   |  | | --- | | #include<stdio.h>  struct Point  {     int x, y;  };    int main()  {     // Create an array of structures  struct Point arr[10];   // Access array members     arr[0].x = 10;     arr[0].y = 20;      printf("%d %d", arr[0].x, arr[0].y);     return 0;  } |   **Output:**  10 20  **//declare a structure of a student with details like roll number,student name,and total marks using array with 50 elements.//write a program to read details of n students and print the list of students who have scored 75 marks and above**  #include<stdio.h>  struct student  {  int roll\_no;  char name[50];  int marks;    }; //the definition must end in semicolon  void main()  {  int i,n;  struct student s[100];  printf("enter how many students");  scanf("%d",&n);  for(i=0;i<n;i++)  {  printf("\nenter the roll no=");  scanf("%d",&s[i].roll\_no);  printf("\nenter the name");  scanf("%s",&s[i].name);  printf("\nenter the marks");  scanf("%d",&s[i].marks);  }  i=0;  while(i<n)  {  if(s[i].marks>=75)  {  printf("\n the name and roll number of students who scored 75 are %s and %d",s[i].name,s[i].roll\_no);  i++;  }  else  i++;  }  }  **//declare a structure employee having details name id phonenumber...print the details of employees.**  #include<stdio.h>  struct employee  { char name[15];  int id;  int phone[10];  };  void main()  {  struct employee emp[100]; //array of structure  int i;  for(i=0;i<3;i++)  {  printf("enter details{name,id,phone} of employee no %d \n",i);  scanf("%s %d %d",emp[i].name,&emp[i].id,emp[i].phone);  printf("\n\n");  }  for(i=0;i<3;i++)  {  printf("details of employee number %d is",i);  printf("%s %d %d",emp[i].name,emp[i].id,emp[i].phone);  printf("\n\n");  }  }  **//example to access members of a structure using period operator(dot operator)**  #include<stdio.h>  #include<string.h>  struct person  {  char name[15];  int id;  float height;  };  void main( )  {  struct person p1;  struct person p2;  printf("\n enter person's name,id & height \n");  scanf("%s %d %f",p1.name,&p1.id,&p1.height);  printf("name:%s \nid:%d\nheight:%f",p1.name,p1.id,p1.height);  strcpy(p2.name,"john");  p2.id=123;  p2.height=6.7;  printf("name:%s \nid:%d \nheight:%f",p2.name,p2.id,p2.height);  } |