**STRUCTURE**

Structure is the collection of variables of different types under a single name for better handling.

Syntax of structure

struct structure\_name

{

data\_type member1;

data\_type member2;

.

.

data\_type memeber;

};

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

struct person

{

char name[50];

int cit\_no;

float salary;

};

This declaration above creates the derived data type struct person.

Structure variable declaration

When a structure is defined, it creates a user-defined type but, no storage is allocated. For the above structure of person, variable can be declared as:

struct person

{

char name[50];

int cit\_no;

float salary;

};

Inside main function:

struct person p1, p2, p[20];

Another way of creating structure variable is:

struct person

{

char name[50];

int cit\_no;

float salary;

}p1 ,p2 ,p[20];

In both cases, 2 variables p1, p2 and array p having 20 elements of type struct person are created.

Accessing members of a structure

There are two types of operators used for accessing members of a structure.

Member operator(.)

Structure pointer operator(->)

Any member of a structure can be accessed as: structure\_variable\_name.member\_name

Suppose, we want to access salary for variable p2. Then, it can be accessed as:

p2.salary

#include <iostream.h>

struct student{

char name[50];

int roll;

float marks;

};

int main()

{

struct student s;

cout<<"Enter information of students:\n";

cout<<"\nEnter roll number %d\n";

cin>>s.roll;

cout<<"Enter name: ";

cin>>s.name;

cout<<"Enter marks: ";

cin>>s.marks;

cout<<"Displaying information of students:\n\n";

cout<<"\nInformation for roll number:\n"<<s.roll;

cout<<"Name:"<<s.name;

cout<<"Marks:"<<s.marks;

return 0;

}

Arrays within Structures

include <iostream.h>

#include <string.h>

struct Books

{

char title[50];

char author[50];

char subject[100];

int book\_id[10];

};

int main( )

{

struct Books Book1; /\* Declare Book1 of type Book \*/

struct Books Book2; /\* Declare Book2 of type Book \*/

/\* book 1 specification \*/

strcpy( Book1.title, "C Programming");

strcpy( Book1.author, "Nuha Ali");

strcpy( Book1.subject, "C Programming Tutorial");

Book1.book\_id[0] = 6495407;

/\* book 2 specification \*/

Cout<<“Enter the BOOK2 Details \n”;

Cin>> Book2.title;

Cin>> Book2.author;

Cin>> Book2.subject;

Cin>>Book2.book\_id[1];

/\* strcpy( Book2.title, "Telecom Billing");

strcpy( Book2.author, "Zara Ali");

strcpy( Book2.subject, "Telecom Billing Tutorial");

Book2.book\_id = 6495700;

\*/

/\* print Book1 info \*/

Cout<< "Book 1 title : \n"<< Book1.title;

Cout<< "Book 1 author : \n"<< Book1.author;

Cout<< "Book 1 subject : \n"<< Book1.subject;

Cout<< "Book 1 book\_id :\n"<<Book1.book\_id[0];

/\* print Book2 info \*/

Cout<< "Book 2 title : \n"<< Book2.title;

Cout<< "Book 2 author : \n"<< Book2.author;

Cout<< "Book 2 subject : \n"<< Book2.subject;

Cout<< "Book 2 book\_id :\n"<<Book2.book\_id[1];

return 0;

}

Arrays of Structures

## Source Code to Store Information of 10 students Using Structure

#include <iostream.h>

struct student{

char name[50];

int roll;

float marks;

};

int main(){

struct student s[10];

int i;

cout<<"Enter information of students:\n";

for(i=0;i<10;++i)

{

s[i].roll=i+1;

cout<<"\nFor roll number\n"<<s[i].roll;

cout<<"Enter name: ";

cin>>s[i].name;

cout<<"Enter marks: ";

cin>>s[i].marks;

cout<<"\n";

}

Cout<<"Displaying information of students:\n\n";

for(i=0;i<10;++i)

{

Cout<<"\nInformation for roll number:\n"<<i+1;

Cout<<"Name: ";

puts(s[i].name);

cout<<"Marks:”<<s[i].marks;

}

return 0;

}

Source code to add two distance using structure

#include <stdio.h>

struct Distance{

int feet;

float inch;

}d1,d2,sum;

int main(){

printf("Enter information for 1st distance\n");

printf("Enter feet: ");

scanf("%d",&d1.feet);

printf("Enter inch: ");

scanf("%f",&d1.inch);

printf("\nEnter information for 2nd distance\n");

printf("Enter feet: ");

scanf("%d",&d2.feet);

printf("Enter inch: ");

scanf("%f",&d2.inch);

sum.feet=d1.feet+d2.feet;

sum.inch=d1.inch+d2.inch;

/\* If inch is greater than 12, changing it to feet. \*/

if (sum.inch>12.0)

{

sum.inch=sum.inch-12.0;

++sum.feet;

}

printf("\nSum of distances=%d\'-%.1f\"",sum.feet,sum.inch);

return 0;

}

Output

Enter information for 1st distance

Enter feet: 12

Enter inch: 3.45

Enter information for 1st distance

Enter feet: 12

Enter inch: 9.2

Sum of distances=25'-0.6"

Structure within Structure

struct date

{

int date;

int month;

int year;

};

struct Employee

{

char ename[20];

int ssn;

float salary;

struct date doj;

}emp1;

Declare Embedded structures

struct Employee

{

char ename[20];

int ssn;

float salary;

struct date

{

int date;

int month;

int year;

}doj;

}emp1;

Accessing Nested Members :

Accessing Month Field : emp1.doj.month

Accessing day Field : emp1.doj.day

Accessing year Field : emp1.doj.year

Example program

#include <stdio.h>

struct Employee

{

char ename[20];

int ssn;

float salary;

struct date

{

int date;

int month;

int year;

}doj;

}emp = {"Pritesh",1000,1000.50,{22,6,1990}};

int main(int argc, char \*argv[])

{

printf("\nEmployee Name : %s",emp.ename);

printf("\nEmployee SSN : %d",emp.ssn);

printf("\nEmployee Salary : %f",emp.salary);

printf("\nEmployee DOJ : %d/%d/%d", \

emp.doj.date,emp.doj.month,emp.doj.year);

return 0;

}

Output :

Employee Name : Pritesh

Employee SSN : 1000

Employee Salary : 1000.500000

Employee DOJ : 22/6/1990

Structures and Functions

Passing structure to function in C:

It can be done in below 3 ways.

* Passing structure to a function by value
* Passing structure to a function by address(reference)
* No need to pass a structure – Declare structure variable as global

Example program – passing structure to function in C by value:

           In this program, the whole structure is passed to another function by value. It means the whole structure is passed to another function with all members and their values. So, this structure can be accessed from called function. This concept is very useful while writing very big programs in C.

#include <stdio.h>

#include <string.h>

struct student

{

int id;

char name[20];

float percentage;

};

void func(struct student record);

int main()

{

struct student record;

record.id=1;

strcpy(record.name, "Raju");

record.percentage = 86.5;

func(record);

return 0;

}

void func(struct student record)

{

printf(" Id is: %d \n", record.id);

printf(" Name is: %s \n", record.name);

printf(" Percentage is: %f \n", record.percentage);

}

Output:

|  |
| --- |
| Id is: 1  Name is: Raju  Percentage is: 86.500000 |

Example program – Passing structure to function in C by address:

           In this program, the whole structure is passed to another function by address. It means only the address of the structure is passed to another function. The whole structure is not passed to another function with all members and their values. So, this structure can be accessed from called function by its address.

#include <stdio.h>

#include <string.h>

struct student

{

int id;

char name[20];

float percentage;

};

void func(struct student \*record);

int main()

{

struct student record;

record.id=1;

strcpy(record.name, "Raju");

record.percentage = 86.5;

func(&record);

return 0;

}

void func(struct student \*record)

{

printf(" Id is: %d \n", record->id);

printf(" Name is: %s \n", record->name);

printf(" Percentage is: %f \n", record->percentage);

}

Output:

|  |
| --- |
| Id is: 1  Name is: Raju  Percentage is: 86.500000 |

Example program to declare a structure variable as global in C:

           Structure variables also can be declared as global variables as we declare other variables in C. So, When a structure variable is declared as global, then it is visible to all the functions in a program. In this scenario, we don’t need to pass the structure to any function separately.

#include <stdio.h>

#include <string.h>

struct student

{

int id;

char name[20];

float percentage;

};

struct student record; // Global declaration of structure

void structure\_demo();

int main()

{

record.id=1;

strcpy(record.name, "Raju");

record.percentage = 86.5;

structure\_demo();

return 0;

}

void structure\_demo()

{

printf(" Id is: %d \n", record.id);

printf(" Name is: %s \n", record.name);

printf(" Percentage is: %f \n", record.percentage);

}

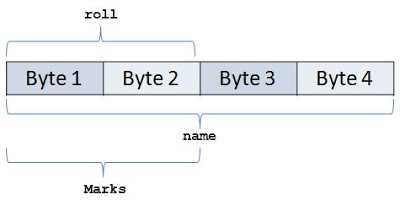
Output:

|  |
| --- |
| Id is: 1  Name is: Raju  Percentage is: 86.500000 |

Union

|  |  |
| --- | --- |
| union stud  {  int roll;  char name[4];  int marks;  }s1; | struct stud  {  int roll;  char name[4];  int marks;  }s1; |

All Union Members Occupy Same Memory Area

  
For the union maximum memory allocated will be equal to the data member with maximum size. In the example character array ‘name’ have maximum size thus maximum memory of the union will be 4 Bytes.

Maximum Memory of Union = Maximum Memory of Union

Data Member

Only one Member will be active at a time.

Suppose we are accessing one of the data member of union then we cannot access other data member since we can access single data member of union because each data member shares same memory. By Using Union we can Save Lot of Valuable Space

Simple Program #1 :

#include <stdio.h>

union emp

{

int id;

char name[20];

}e1;

int main()

{

e1.id = 10;

printf("\nID : %d",e1.id);

strcpy(e1.name,"Pritesh");

printf("\nName : %s",e1.name);

return 0;

}

Output :

ID : 10

Name : Pritesh

Programs using Structure

Write a C program using structure to find student grades in a class. Make necessary assumption.

Ans.

Before writing program we make some assumption as following :

Maximum total marks is 500.

  Student\_percentage         Grades

     >=80                        A

     >=60                        B

     >=50                        C

     >=40                        D

     <40                         F

/\*C program to find students grades in a class through structure \*/

#include<stdio.h>

#include<conio.h>

struct stud

{  
  char nam[20];

  int obtain\_mark;

  int per;

  char grad[5];

};

struct stud s[5];

int i;

int main()

{

 for(i=1; i<=5; i++)

 {

  printf("Enter %d student name : ",i);

  scanf("%s",&s[i].nam);  
  printf("Enter %d student obtained marks = ",i);  
  scanf("%d",&s[i].obtain\_mark);  
  fflush(stdin);

 }

 for(i=1; i<=5; i++)  
   s[i].per=s[i].obtain\_mark/5;  
 for(i=1; i<=5; i++)  
 {

  if(s[i].per>=80)

    strcpy(s[i].grad,"A");

  else if(s[i].per>=60)

    strcpy(s[i].grad,"B");

  else if(s[i].per>=50)

    strcpy(s[i].grad,"C");

  else if(s[i].per>=40)

    strcpy(s[i].grad,"D");

  else

    strcpy(s[i].grad,"F");  
 }

 for(i=1; i<=5; i++)

  printf("\n%d student %s has obtained grade %s ",i,s[i].nam,s[i].grad);  
 getch();

 return 0;

}

[C Program Print Student Marklist](http://ecomputernotes.com/c-program/write-a-program-to-print-student-marklist)

#include<stdio.h>   
#include<conio.h>   
int k=0;   
struct stud   
{   
       int rn;   
       char name[30];   
       int m1,m2,m3,total;   
       float avg;   
       char grade,result;   
}s[30];   
       void main()   
{   
       int no,roll=101,i;   
       clrscr();   
       printf("Enter No of Students : ");   
       scanf("%d",&no);   
       for(i=0;i<no;i++)   
       {   
            clrscr();   
            s[k].rn=roll;   
            printf("\nEnter the Student Roll Number : %d ",s[k].rn);   
            printf("\nEnter the Student Name :");   
            fflush(stdin);   
            gets(s[k].name);   
            printf("\nEnter the Three Marks : ");   
            scanf("%d",&s[k].m1);   
            scanf("%d",&s[k].m2);   
            scanf("%d",&s[k].m3);   
            if(s[k].m1>=35 && s[k].m2>=35 && s[k].m3>=35)   
              {   
                 s[k].result='P';   
              }   
            else   
             {   
                 s[k].result = 'F';   
             }   
                 s[k].total = s[k].m1+s[k].m2+s[k].m3;   
                 printf("The Total is : %d",s[k].total);   
                 s[k].avg=s[k].total/3;   
                 if(s[k].avg>=60)   
                  {   
                      if(s[k].result == 'P')   
                         {   
                              s[k].grade = 'A';   
                         }   
                        else   
                        {   
                               s[k].grade = 'N';   
                        }   
                  }   
                 else if(s[k].avg>=50)   
                         {   
                              if(s[k].result == 'P')   
                                {   
                                       s[k].grade = 'B';   
                                 }   
                               else   
                                {   
                                       s[k].grade = 'N';   
                                }   
                         }   
                 else if(s[k].avg>=35)   
                        {   
                             if(s[k].result == 'P')   
                               {   
                                     s[k].grade = 'C';   
                               }   
                             else   
                              {   
                                     s[k].grade = 'N';   
                              }   
                        }   
                                      getch();   
                                      k++;   
                                      roll++;   
           }   
                       printf("\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");   
                       printf("\n                      STUDENT MARKLIST ");   
                       printf("\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");   
                       printf("\nROLL \tNAME   MARK1 MARK2 MARK3 TOTAL RESULT Average GRADE");   
                       for(i=0;i<no;i++)   
                           {   
                              printf("\n%d\t%s   %d    %d    %d    %d    %c    %0.1f     %c",s[i].rn,s[i].name,s[i].m1,s[i].m2,s[i].m3,s[i].total,s[i].result,s[i].avg,s[i].grade);   
                           }   
                              getch();   
}