

Aim:- Program to understand structure & Union operations.

Software required:- Turbo C/C++

Theory:-

Structure:- Structure in C language is a user defined datatype that allows you to hold different type of elements.

Difference Between arrays & structures -

An array is a collection of related data elements of same type. Structure can have elements of different types.

Syntax for defining structure -

// The struct keyword is used to define structure struct

Structure - name

{

data_type member 1;

data_type member 2;

:

data_type member N;

} ;

Union :- A union is a special data types in the ~~same~~ available in C that allows to store different data types in the same memory location. You can define a union with many members, but only one member can contain a value at any given time. Unions provide an efficient way of using the same memory location for multiple - purpose. To define a union, you must use the union statement in the same way as you did while defining a structure. The union statement defines a new data type with more than one member for your program. The format of the union statement is as follows -

```
union [union tag] { member  
    {  
        member definition;  
        member definition;  
        ;  
        member definition;  
    } [one or more union variables];
```

Output of the Code:-

Enter the total number of students:-

2

Enter the roll no:- 1

Enter the name: Karthi

Enter three marks: 98

97

99

Enter the roll no:- 2

Enter the name: Kannan

Enter three marks: 90

87

83

Student details:-

1. Karthi 98 97 99 294 98.000000

2. Kannan 90 87 83 266 88.666667

Practical No.

PAGE NO. 55
DATE:

Code 1

```
#include <stdio.h>
```

```
#include <conio.h>
```

```
struct student
```

```
{
```

```
int rollno, m1, m2, m3, total;
```

```
float average;
```

```
char name[30];
```

```
};
```

```
void main()
```

```
{
```

```
struct student s[10];
```

```
int n, i;
```

```
clrscr();
```

```
printf("\nEnter the total number of students:");
```

```
scanf("%d", &n);
```

```
for(i=0; i<n; i++)
```

```
{
```

```
printf("Enter the roll no:");
```

```
scanf("%d", &s[i].rollno);
```

```
printf("Enter three marks:");
```

```
scanf("%d %d %d", &s[i].m1, &s[i].m2, &s[i].m3);
```

```
s[i].total = s[i].m1 + s[i].m2 + s[i].m3;
```

```
s[i].average = (float) s[i].total / 3;
```

```
}
```

Instructor's Sign

```

printf("\n\nStudent details\n");
for(i=0; i<n; i++)
{
    printf("\n%.d%.s%.d%.d%.d%.d%.f\n", sci.college_id,
        sci.name, sci.m1, sci.m2, sci.m3, sci.total,
        sci.average);
}
getch();
}

```

Code 2

```

#include <stdio.h>
#include <conio.h>
#include <string.h>
struct student_collage_detail
{
    int college_id;
    char collage_name[50];
};
struct student_detail
{
    int id;
    char name[20];
    float percentage; // structure within structure
}

```

Output of the Code 2

Id is : 1

Name is : Raju

Percentage is : 90.500000

College Id is : 71145

College Name is : Global Institute of Technology.

Practical No.

PAGE NO.

57

DATE:

```
structure struct
student_collage-detail.clg data;
}
student
stu-data;
```

```
void main ()
```

```
{
```

```
struct student-detail
```

```
stu_data = {1, "Raju", 90.5, 71145, "Anna University"};
```

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

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

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

```
printf("College Id is: %d\n", stu_data.clg_data.collage  
-id);
```

```
printf("College name is: %s\n",
```

```
stu_data.clg_data.collage_name);
```

```
getch();
```

```
}
```

Output of Code 3

Union record 1 values example

Name: Raju

Subject: Maths

percentage: 86.500000;

Union record 2 values example

Name: Mani

Subject: Physics

percentage: 99.500000;

Practical No.

PAGE NO. 58
DATE:

Code 3

```
#include <stdio.h>
```

```
#include <conio.h>
```

```
#include <string.h>
```

```
union student
```

```
{
```

```
    char name[20];
```

```
    char subject[20];
```

```
    float percentage;
```

```
};
```

```
void main()
```

```
{
```

```
    union student record1;
```

```
    union record2;
```

```
    strcpy (record1.name, "Raju");
```

```
    strcpy (record1.subject, "Maths");
```

```
    record1.percentage = 86.50;
```

```
    printf("Union record 1 values example\n");
```

```
    printf("Name: %s\n", record1.name);
```

```
    printf("Subject: %s\n", record1.subject);
```

```
    printf("Percentage = %f\n", record1.percentage);
```

```
    printf("Union record 2 values example\n");
```

```
    printf("Name: %s\n", record2.name);
```

```
    printf("Subject: %s\n", record2.subject);
```

```
    printf("percentage = %f\n", record2.percentage);
```

```
    strcpy (record2.name, "Mani");
```

```
    strcpy (record2.subject, "Physics");
```

```
    strcpy record2.percentage = 99.50;
```

Instructor's Sign

```
getch();  
}
```

Viva Questions

Q1 What is the difference between a structure & an array?

Ans A structure is a data structure that can contains variables of different data types. An array is a data structure that can only contain variables of the same data type.

Q2 How is a structure different from a union?
Ans In structure, there is a specific memory location for every input data member. It can store multiple values of the various members.

In union, there is an allocation of only one shared memory for all the input data member. It stores one value at a time for all of its members.

Q3 What is the role of dot (.) and -> operators?

Ans The Dot (.) operator is used to normally access members of a structure or union. The Arrow (->) operators exists to access the member of the structure or the unions using pointers.

Q4 What are the properties of nesting one structure into another?

Ans Structure nesting refers to the practice of defining one structure inside another structure in C program. This allows you to create complex data structures where a structure can have members that are themselves structure. This concept is useful for organizing & representing hierarchical or composite data.

Q5 What do you understand by Enumerated data-type & Type-definition?

Ans An enumeration is a data-type that consists of a set of named values that represent integral constants, known as enumeration constants. We must list each of values in creating a name for each of them.

In C A type definition allows you to create an alias or a new name for an existing data type. Its primary purpose is to simplify complex type declaration for variable.