

## Day 8: Structures and Unions (11-8-2025)

1. Define a structure for student record and print details.

IPO:

INPUT: Name, roll number, marks of a student

PROCESS: Store details in a structure

OUTPUT: Display student details

CODE;

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

```
struct Student
```

```
{
```

```
    char name[50];
```

```
    int roll;
```

```
    float marks;
```

```
}s;
```

```
void main()
```

```
{
```

```
    scanf("%s", s.name);
```

```
    scanf("%d", &s.roll);
```

```
    scanf("%f", &s.marks);
```

```
    printf("\nStudent Details:\nName: %s\nRoll: %d\nMarks: %.2f", s.name,  
s.roll, s.marks);
```

OUTPUT;

### Output

sam 123 95

Student Details:

Name: sam

Roll: 123

Marks: 95.00

2. Write a program to store and display employee details using structures.

IPO:

INPUT: Employee name, ID, salary

PROCESS: Store and retrieve values from structure

OUTPUT: Display employee details

CODE;

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

```
struct Employee
```

```
{
```

```
    char name[50];
```

```
    int id;
```

```
    float salary;
```

```
}e;
```

```
void main()
```

```
{
```

```
    scanf("%s%d%f", e.name, &e.id, &e.salary);
```

```
    printf("Name: %s\nID: %d\nSalary: %.2f", e.name, e.id, e.salary);
```

```
}
```

OUTPUT;

### Output

```
sam 12 100000  
Name: sam  
ID: 12  
Salary: 100000.00
```

3. Write a program to pass a structure to a function.

IPO:

INPUT: Student details

PROCESS: Pass structure variable to a function to print

OUTPUT: Display student details

CODE;

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

```
struct Student
```

```
{
```

```
    char name[50];
```

```
    int roll;
```

```
}s;
```

```
void display(struct Student s)
```

```
{
```

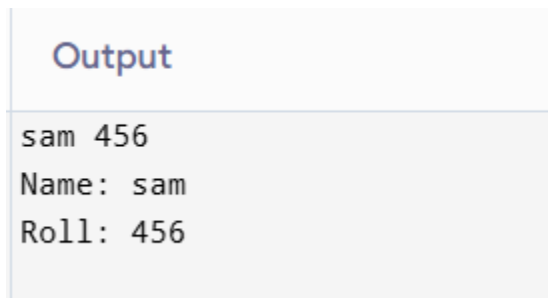
```
    printf("Name: %s\nRoll: %d", s.name, s.roll);
```

```
}
```

```
void main()
```

```
{  
    scanf("%s%d", s.name, &s.roll);  
    display(s);  
}
```

OUTPUT;

A screenshot of a program's output. It features a light gray rectangular box with a thin blue border. At the top left of the box, the word "Output" is written in a blue, sans-serif font. Below this header, the output text is displayed in a black, monospaced font. The output consists of three lines: "sam 456", "Name: sam", and "Roll: 456".

Output

sam 456  
Name: sam  
Roll: 456

4. Write a program to store multiple student records using array of structures.

IPO:

INPUT: Name, roll, marks for multiple students

PROCESS: Store in array of structures

OUTPUT: Display all student details

CODE;

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

```
struct Student
```

```
{  
    char name[50];  
    int roll;
```

```

    float marks;
}s[3];
void main()
{
    int i;
    for(i = 0; i < 3; i++)
        scanf("%s%d%f", s[i].name, &s[i].roll, &s[i].marks);
    for(i = 0; i < 3; i++)
        printf("%s %d %.2f\n", s[i].name, s[i].roll, s[i].marks);
}

```

OUTPUT;

Output
sam 12 95 john 13 96
tim 14 97
sam 12 95.00
john 13 96.00
tim 14 97.00

5. Write a program to demonstrate nested structures.

IPO:

INPUT: Employee details with address

PROCESS: Use structure inside another structure

OUTPUT: Display details with address

CODE;

```

#include <stdio.h>

struct Address
{
    char city[50];
    int pin;
};

struct Employee {
    char name[50];
    struct Address addr;
};

void main()
{
    struct Employee e;
    scanf("%s%s%d", e.name, e.addr.city, &e.addr.pin);
    printf("%s %s %d", e.name, e.addr.city, e.addr.pin);
}

```

OUTPUT;

Output
sam chennai 600122
sam chennai 600122

6. Write a program to calculate total and average marks using structures

IPO:

INPUT: Marks in 3 subjects

PROCESS: Store in structure, sum, and average

OUTPUT: Total and average marks

CODE;

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

```
struct Marks
```

```
{
```

```
    int m1, m2, m3;
```

```
};
```

```
void main()
```

```
{
```

```
    struct Marks m;
```

```
    int total;
```

```
    float avg;
```

```
    scanf("%d%d%d", &m.m1, &m.m2, &m.m3);
```

```
    total = m.m1 + m.m2 + m.m3;
```

```
    avg = total / 3.0;
```

```
    printf("Total = %d\nAverage = %.2f", total, avg);
```

```
}
```

OUTPUT;

Output
87 89 96 Total = 272 Average = 90.67

7. Write a program to find the highest marks among students.

IPO:

INPUT: Name and marks of students

PROCESS: Compare marks to find highest

OUTPUT: Student with highest marks

CODE;

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

```
struct Student {
```

```
    char name[50];
```

```
    float marks;
```

```
}s[3];
```

```
void main()
```

```
{
```

```
    int i, maxIndex = 0;
```

```
    for(i = 0; i < 3; i++)
```

```
        scanf("%s%f", s[i].name, &s[i].marks);
```

```
    for(i = 1; i < 3; i++)
```

```
        if(s[i].marks > s[maxIndex].marks)
```

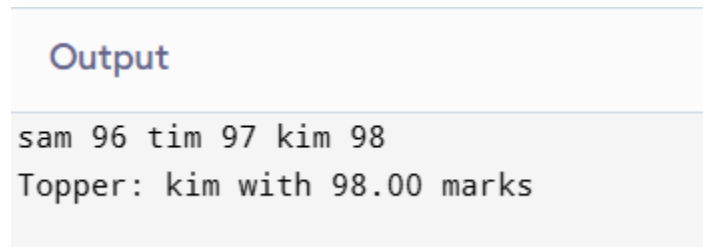


```
        maxIndex = i;

    printf("Topper: %s with %.2f marks", s[maxIndex].name,
s[maxIndex].marks);

}
```

OUTPUT;

A screenshot of a program's output. It shows two lines of text: "sam 96 tim 97 kim 98" and "Topper: kim with 98.00 marks". The text is displayed in a monospaced font on a light gray background.

```
Output
sam 96 tim 97 kim 98
Topper: kim with 98.00 marks
```

9. Write a program using union to store data of different types.

IPO:

INPUT: Different types of data

PROCESS: Store in union one at a time

OUTPUT: Display stored value

CODE;

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

```
union Data {
```

```
    int i;
```

```
    float f;
```

```
    char c;
```

```
}d;
```

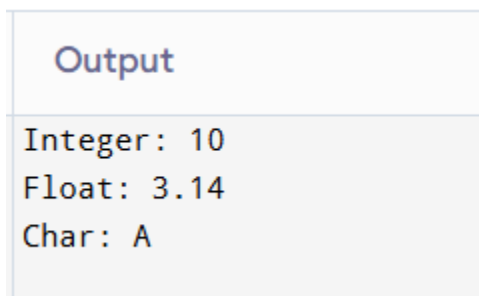
```
void main()
```

```
{
```

```
    d.i = 10;
```

```
printf("Integer: %d\n", d.i);  
d.f = 3.14;  
printf("Float: %.2f\n", d.f);  
d.c = 'A';  
printf("Char: %c\n", d.c);  
}
```

OUTPUT;

A screenshot of a program's output. It features a light gray rectangular box with a thin border. At the top left of the box, the word "Output" is written in a blue, sans-serif font. Below this header, the output of the program is displayed in a monospaced font: "Integer: 10", "Float: 3.14", and "Char: A", each on a new line.

Output

Integer: 10  
Float: 3.14  
Char: A

10.Compare and contrast structure vs union with a sample program.

IPO:

INPUT: Values for structure and union

PROCESS: Show how structure stores all fields, union shares memory

OUTPUT: Demonstrate memory usage difference

CODE;

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

```
struct S {
```

```
    int a;
```

```
    int b;
```

```
};
```

```
union U {
```

```
int a;
int b;
};
void main()
{
    struct S s1;
    union U u1;
    printf("Enter two integers for structure: ");
    scanf("%d%d", &s1.a, &s1.b);

    printf("Enter two integers for union: ");
    scanf("%d%d", &u1.a, &u1.b);

    printf("\nStructure values: %d %d", s1.a, s1.b);
    printf("\nUnion values: %d %d", u1.a, u1.b);

    printf("\nSize of structure: %d", (int)sizeof(s1));
    printf("\nSize of union: %d", (int)sizeof(u1));
}
```

OUTPUT;

## Output

Enter two integers for structure: 15 12

Enter two integers for union: 20 25

Structure values: 15 12

Union values: 25 25

Size of structure: 8

Size of union: 4