**Day 8**

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

#include <stdio.h>

struct Student {

char name[50];

int age;

float marks;

};

int main() {

struct Student s = {"John", 20, 85.5};

printf("Name: %s\nAge: %d\nMarks: %.2f\n", s.name, s.age, s.marks);

return 0;

}

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

#include <stdio.h>

struct Employee {

int id;

char name[50];

float salary;

};

int main() {

struct Employee e = {101, "Alice", 55000.75};

printf("ID: %d\nName: %s\nSalary: %.2f\n", e.id, e.name, e.salary);

return 0;

}

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

#include <stdio.h>

struct Student {

char name[50];

int age;

};

void display(struct Student s) {

printf("Name: %s\nAge: %d\n", s.name, s.age);

}

int main() {

struct Student s1 = {"Bob", 19};

display(s1);

return 0;

}

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

#include <stdio.h>

struct Student {

char name[50];

int age;

};

int main() {

struct Student students[2] = {

{"Tom", 18},

{"Jerry", 19}

};

for (int i = 0; i < 2; i++) {

printf("Name: %s, Age: %d\n", students[i].name, students[i].age);

}

return 0;

}

5. Write a program to demonstrate nested structures.

#include <stdio.h>

struct Date {

int day, month, year;

};

struct Student {

char name[50];

struct Date dob;

};

int main() {

struct Student s = {"Lily", {15, 8, 2004}};

printf("Name: %s\nDOB: %02d/%02d/%04d\n", s.name, s.dob.day, s.dob.month, s.dob.year);

return 0;

}

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

#include <stdio.h>

struct Student {

char name[50];

float marks1, marks2, marks3;

};

int main() {

struct Student s = {"Eve", 78.5, 82.0, 91.5};

float total = s.marks1 + s.marks2 + s.marks3;

float avg = total / 3;

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

return 0;

}

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

#include <stdio.h>

struct Student {

char name[50];

float marks;

};

int main() {

struct Student s[3] = {

{"Anna", 87.5},

{"Ben", 92.0},

{"Cara", 89.3}

};

float max = s[0].marks;

char top[50];

for (int i = 0; i < 3; i++) {

if (s[i].marks > max) {

max = s[i].marks;

strcpy(top, s[i].name);

}

}

printf("Topper: %s with %.2f marks\n", top, max);

return 0;

}

8. Write a program to sort student records by name using structure.

#include <stdio.h>

#include <string.h>

struct Student {

char name[50];

int age;

};

int main() {

struct Student s[3] = {

{"Zara", 20},

{"Anna", 19},

{"Mike", 21}

};

struct Student temp;

for (int i = 0; i < 2; i++) {

for (int j = i + 1; j < 3; j++) {

if (strcmp(s[i].name, s[j].name) > 0) {

temp = s[i];

s[i] = s[j];

s[j] = temp;

}

}

}

for (int i = 0; i < 3; i++) {

printf("Name: %s, Age: %d\n", s[i].name, s[i].age);

}

return 0;

}

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

#include <stdio.h>

union Data {

int i;

float f;

char str[20];

};

int main() {

union Data d;

d.i = 10;

printf("Integer: %d\n", d.i);

d.f = 220.5;

printf("Float: %.1f\n", d.f);

strcpy(d.str, "C Programming");

printf("String: %s\n", d.str);

return 0;

}

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

#include <stdio.h>

#include <string.h>

struct SData {

int i;

float f;

char str[20];

};

union UData {

int i;

float f;

char str[20];

};

int main() {

struct SData s = {10, 20.5, "Struct"};

union UData u;

u.i = 10;

u.f = 20.5;

strcpy(u.str, "Union");

printf("Structure:\n");

printf("i = %d, f = %.1f, str = %s\n", s.i, s.f, s.str);

printf("\nUnion:\n");

printf("i = %d\n", u.i);

printf("f = %.1f\n", u.f);

printf("str = %s\n", u.str); // Only 'str' will be correct

return 0;

}