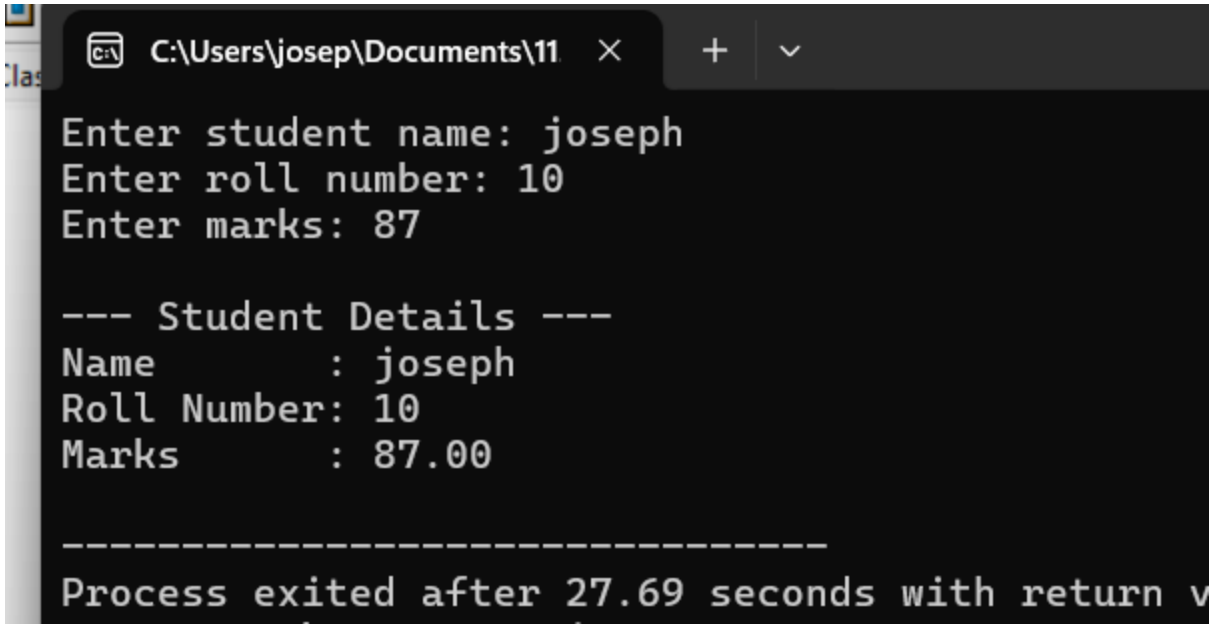
✅ **Day : Structures and Unions (11-8-2025)**

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

IPO: Input – student name, roll number, marks;
Process – store details in structure;
Output – display student details

.

```
#include <stdio.h>
struct Student
{
   char name[50];
   int rollNumber;
   float marks;
}
int main()
{
   struct Student s;
   printf("Enter student name: ");
   scanf("%[^\n]", s.name);
   printf("Enter roll number: ");
   scanf("%d", &s.rollNumber);
   printf("Enter marks: ");
   scanf("%f", &s.marks);
   printf("\n--- Student Details ---\n");
   printf("Name      : %s\n", s.name);
   printf("Roll Number: %d\n", s.rollNumber);
   printf("Marks     : %.2f\n", s.marks);
   return 0;
}
```



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

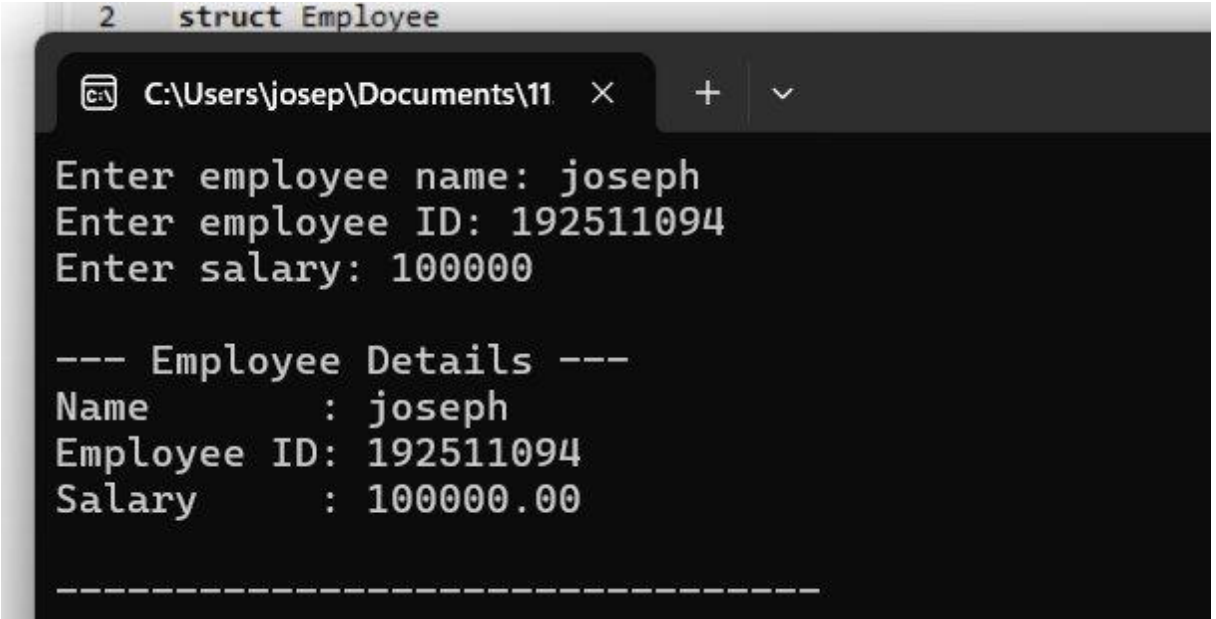
IPO:

Input – employee name, ID, salary;

Process – store details in structure;

Output – display employee details.

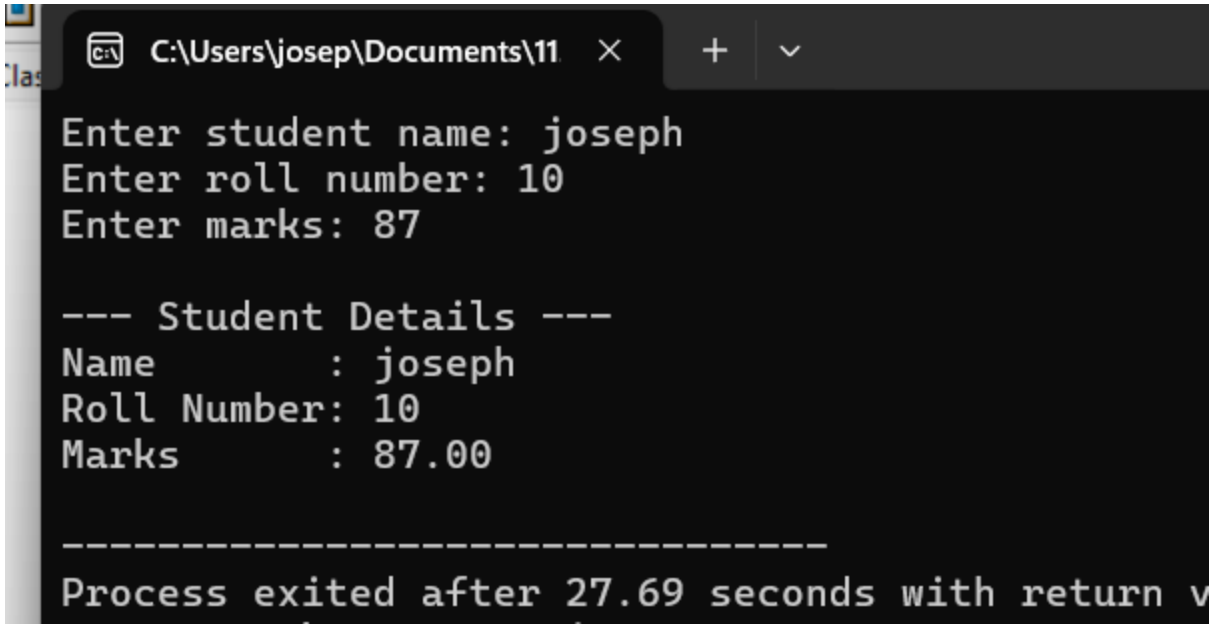
```c
#include <stdio.h>
struct Employee
{
   char name[50];
   int empID;
   float salary;
};

int main()
{
   struct Employee e;
   printf("Enter employee name: ");
   scanf("%[^\n]", e.name);
   printf("Enter employee ID: ");
   scanf("%d", &e.empID);
   printf("Enter salary: ");
   scanf("%f", &e.salary);
   printf("\n--- Employee Details ---\n");
   printf("Name      : %s\n", e.name);
   printf("Employee ID: %d\n", e.empID);
   printf("Salary    : %.2f\n", e.salary);
   return 0;
}
```



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

IPO:
Input – student name, roll number, marks;
Process – pass structure to function and display;
Output – display student details

```
#include <stdio.h>
struct Student
{
   char name[50];
   int roll;
   float marks;
};
void display(struct Student s)
{
   printf("\n--- Student Details ---\n");
   printf("Name      : %s\n", s.name);
   printf("Roll Number: %d\n", s.roll);
   printf("Marks     : %.2f\n", s.marks);
}
int main()
{
   struct Student s1;
   printf("Enter name: ");
   scanf("%[^\n]", s1.name);
   printf("Enter roll number: ");
   scanf("%d", &s1.roll);
   printf("Enter marks: ");
   scanf("%f", &s1.marks);
   display(s1);
   return 0;
}
```

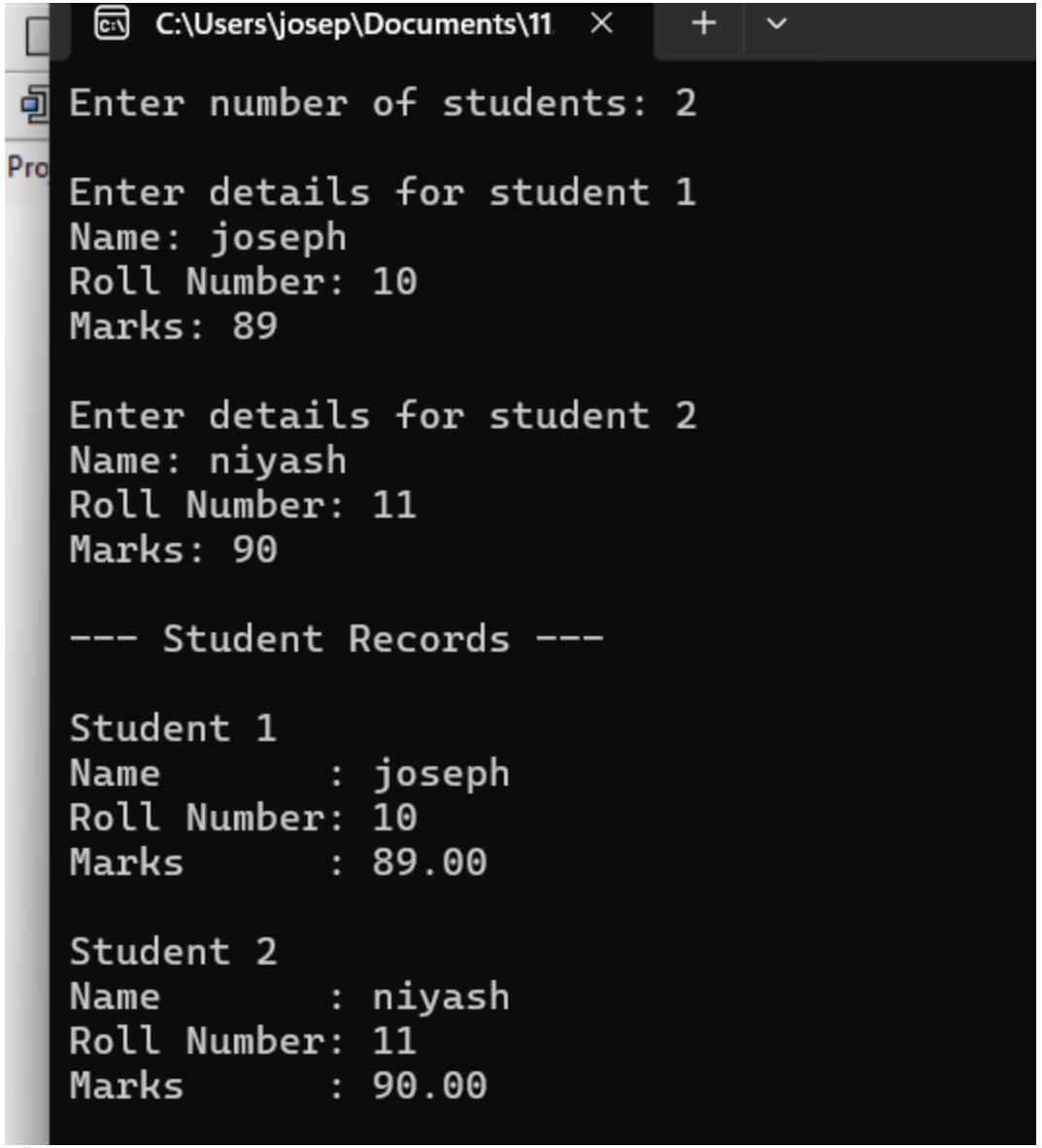


```
int main()
{
   struct Student s[100];
   int n, i;
   printf("Enter number of students: ");
   scanf("%d", &n);
   getchar();

   for (i = 0; i < n; i++) {
      printf("\nEnter details for student
%d\n", i + 1);

      printf("Name: ");
      scanf("%[^\n]", s[i].name);
      getchar();

      printf("Roll Number: ");
      scanf("%d", &s[i].roll);

      printf("Marks: ");
      scanf("%f", &s[i].marks);
      getchar();
   }
   printf("\n--- Student Records ---\n");
   for (i = 0; i < n; i++) {
      printf("\nStudent %d\n", i + 1);
      printf("Name      : %s\n",
s[i].name);
      printf("Roll Number: %d\n",
s[i].roll);
      printf("Marks     : %.2f\n",
s[i].marks);
   }
```

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

IPO:

Input – number of students, each student's name, roll, marks;
Process – store data in array of structures;
Output – display all student records



5. Write a program to demonstrate nested structures.

IPO: Input – student name, roll, date of birth;
Process – store details using nested structures;
Output – display student details with DOB

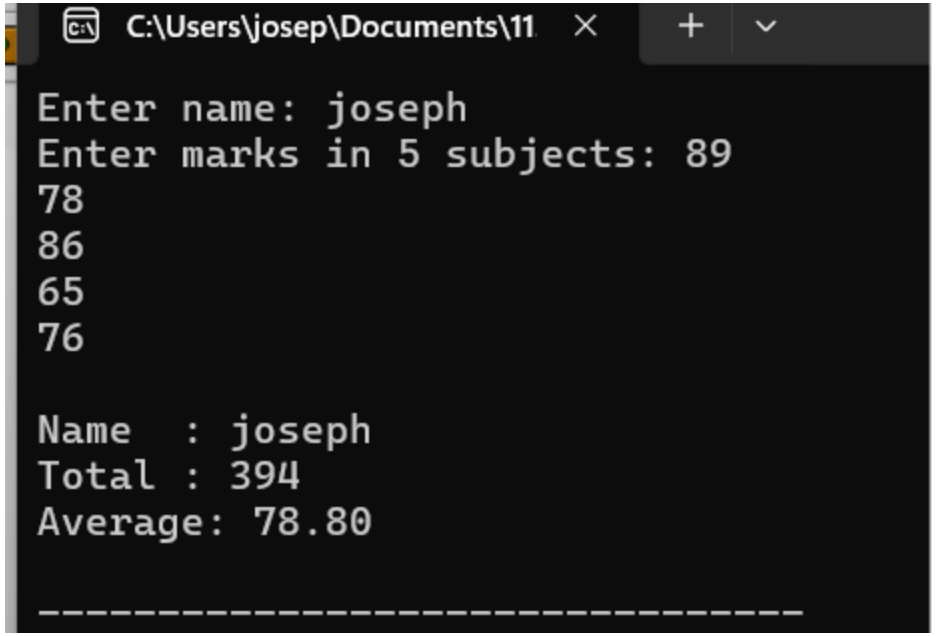
```
#include <stdio.h>
struct Date { int day, month, year; };
struct Student { char name[50]; int roll; struct Date dob; };
int main()
{
   struct Student s;
   printf("Enter name: ");
   scanf("%[^\n]", s.name);
   printf("Enter roll: ");
   scanf("%d", &s.roll);
   printf("Enter DOB (dd mm yyyy): ");
   scanf("%d %d %d", &s.dob.day, &s.dob.month, &s.dob.year);
   printf("\nName: %s\nRoll: %d\nDOB : %02d-%02d-%04d\n",
        s.name, s.roll, s.dob.day, s.dob.month, s.dob.year);
   return 0;
}
```

```
Enter name: joseph
Enter roll: 10
Enter DOB (dd mm yyyy): 11 06 2007

Name: joseph
Roll: 10
DOB : 11-06-2007
```

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

IPO: Input – name, marks in 5 subjects;
Process – calculate total and average using structure;
Output – display name, total, average

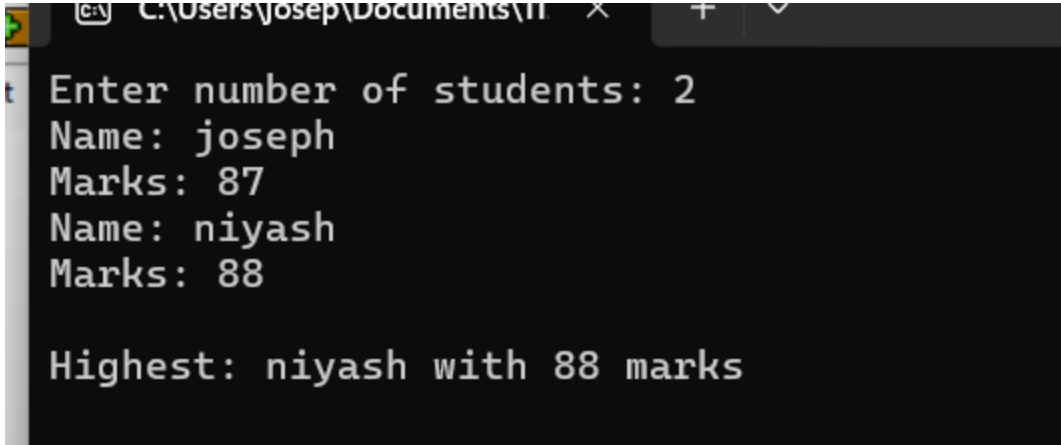
```
#include <stdio.h>
struct Student { char name[50]; int marks[5]; };
int main()
{
   struct Student s;
   int i, total = 0;
   float avg;
   printf("Enter name: ");
   scanf("%[^\n]", s.name);
   printf("Enter marks in 5 subjects: ");
   for (i = 0; i < 5; i++)
{
      scanf("%d", &s.marks[i]);
      total += s.marks[i];
   }
   avg = total / 5.0;
   printf("\nName  : %s\nTotal : %d\nAverage: %.2f\n", s.name, total, avg);
   return 0;
}
```



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

IPO: Input – number of students, each name & marks;
Process – compare marks to find highest;
Output – display student with highest marks

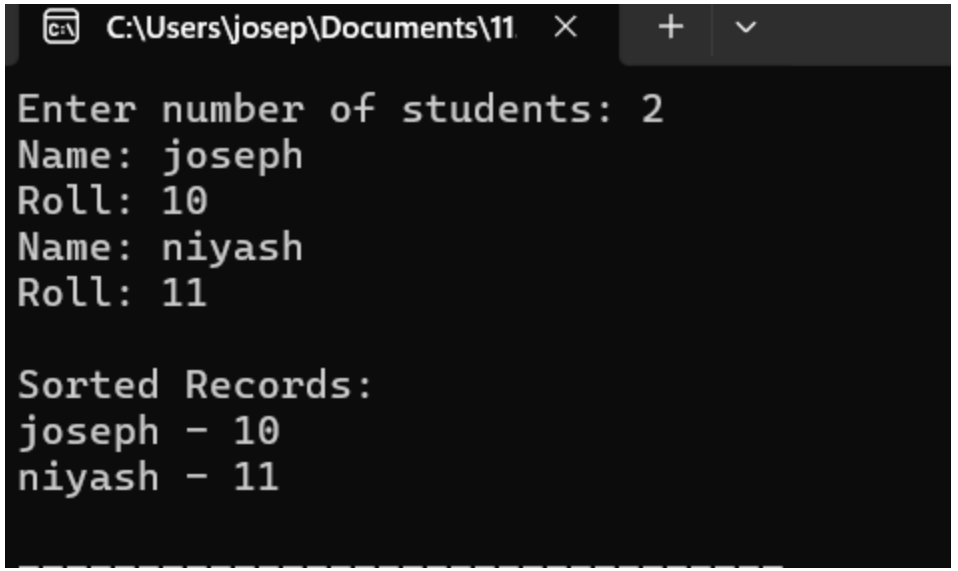
```
#include <stdio.h>
struct Student { char name[50]; int marks; };
int main()
{
   struct Student s[100]; int n,i,max=0;
   printf("Enter number of students: "); scanf("%d",&n);
   for(i=0;i<n;i++)
{
      printf("Name: "); scanf(" %[^\n]",s[i].name);
      printf("Marks: "); scanf("%d",&s[i].marks);
      if(s[i].marks > s[max].marks) max = i;
   }
   printf("\nHighest: %s with %d marks\n", s[max].name, s[max].marks);
}
```



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

IPO: Input – student names & rolls;
Process – sort by name;
Output – display sorted records

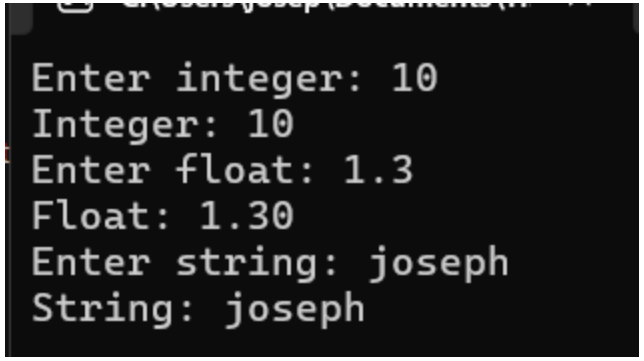
```
#include <stdio.h>
#include <string.h>
struct Student { char name[50]; int roll; };
int main()
{
   struct Student s[100], temp;
   int n, i, j;
   printf("Enter number of students: "); scanf("%d",&n);
   for(i=0;i<n;i++){
      printf("Name: "); scanf(" %[^\n]", s[i].name);
      printf("Roll: "); scanf("%d", &s[i].roll);
   }
   for(i=0;i<n-1;i++)
      for(j=i+1;j<n;j++)
         if(strcmp(s[i].name, s[j].name) > 0)
{
            temp = s[i]; s[i] = s[j]; s[j] = temp;
         }
   printf("\nSorted Records:\n");
   for(i=0;i<n;i++) printf("%s - %d\n", s[i].name, s[i].roll);
}
```



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

IPO: Input – integer, float, string;
Process – store using union (one at a time);
Output – display each value

```
#include <stdio.h>
union Data { int i; float f; char str[50]; };
int main()
{
   union Data d;
   printf("Enter integer: "); scanf("%d", &d.i);
   printf("Integer: %d\n", d.i);
   printf("Enter float: "); scanf("%f", &d.f);
   printf("Float: %.2f\n", d.f);
   printf("Enter string: "); scanf(" %[^\n]", d.str);
   printf("String: %s\n", d.str);
}
```



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

Input – assign values to structure and union;
Process – show memory usage and overwriting;
Output – compare structure vs union behavior

```
#include <stdio.h>
#include <string.h>
struct MyStruct { int i; float f; char str[20]; };
union MyUnion  { int i; float f; char str[20]; };
int main()
{
   struct MyStruct s;
   union MyUnion u;
   s.i = 10; s.f = 3.14; strcpy(s.str, "Hello");
   printf("Structure: %d, %.2f, %s\n", s.i, s.f, s.str);
   u.i = 10;
   printf("Union int: %d\n", u.i);
   u.f = 3.14;
   printf("Union float (overwrites int): %.2f\n", u.f);
   strcpy(u.str, "Hello");
   printf("Union string (overwrites float): %s\n", u.str);
}
```