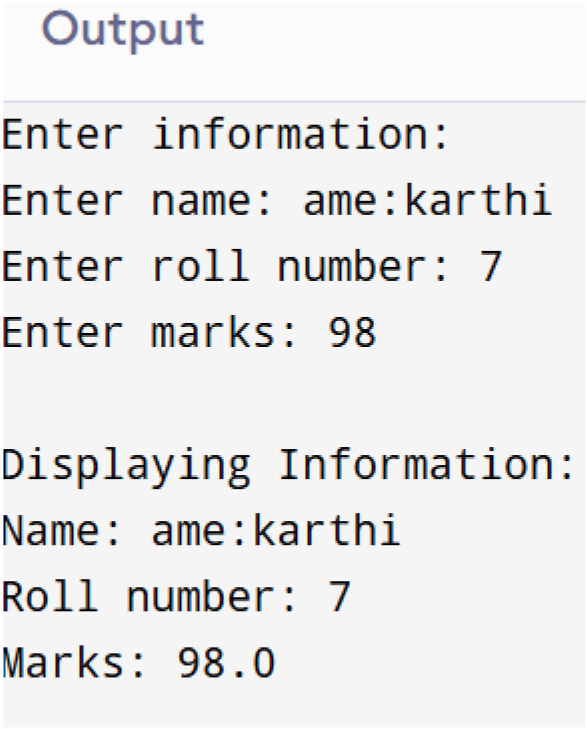
1. Define a structure for student record and print details.

IPO
Input:Enter a value as input.
Process: To find a structure for student record and print details.
Output:output the variable

Program

```
#include <stdio.h>
struct student
{
    char name[50];
    int roll;
    float marks;
} s;

int main() {
    printf("Enter information:\n");
    printf("Enter name: ");
    fgets(s.name, sizeof(s.name), stdin);

    printf("Enter roll number: ");
    scanf("%d", &s.roll);
    printf("Enter marks: ");
    scanf("%f", &s.marks);

    printf("\nDisplaying Information:\n");
    printf("Name: %s", s.name);
    printf("Roll number: %d\n", s.roll);
    printf("Marks: %.1f\n", s.marks);

    return 0;
}
```

Output



Output

```
Enter information:
Enter name: ame:karthi
Enter roll number: 7
Enter marks: 98

Displaying Information:
Name: ame:karthi
Roll number: 7
Marks: 98.0
```

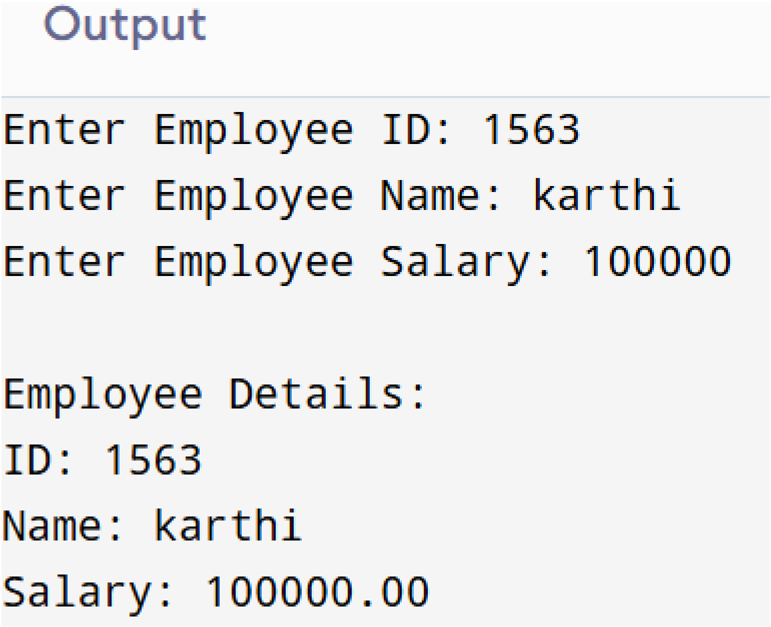
2. Write a program to store and display employee details using structures.

IPO
Input:Enter a value as input.
Process: To store and display employee details using structures.
Output:output the variable

Program:

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

int main() {
    struct Employee emp;

    printf("Enter Employee ID: ");
    scanf("%d", &emp.id);

    printf("Enter Employee Name: ");
    scanf("%s", emp.name);
    printf("Enter Employee Salary: ");
    scanf("%f", &emp.salary);

    printf("\nEmployee Details:\n");
    printf("ID: %d\n", emp.id);
    printf("Name: %s\n", emp.name);
    printf("Salary: %.2f\n", emp.salary);

    return 0;
}
```

Output



Output

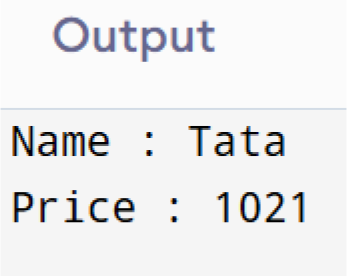
```
Enter Employee ID: 1563
Enter Employee Name: karthi
Enter Employee Salary: 100000

Employee Details:
ID: 1563
Name: karthi
Salary: 100000.00
```

3. Write a program to pass a structure to a function.
IPO
Input:Enter a value as input.
Process: To to pass a structure to a function
Output:output the variable

Program:

```
#include <stdio.h>
struct car
{
    char name[30];
    int price;
};
void print_car_info(struct car c)
{
    printf("Name : %s", c.name);
    printf("\nPrice : %d\n", c.price);
}

int main() {
    struct car c = { "Tata", 1021 };
    print_car_info(c);
    return 0;
}
```

Output:



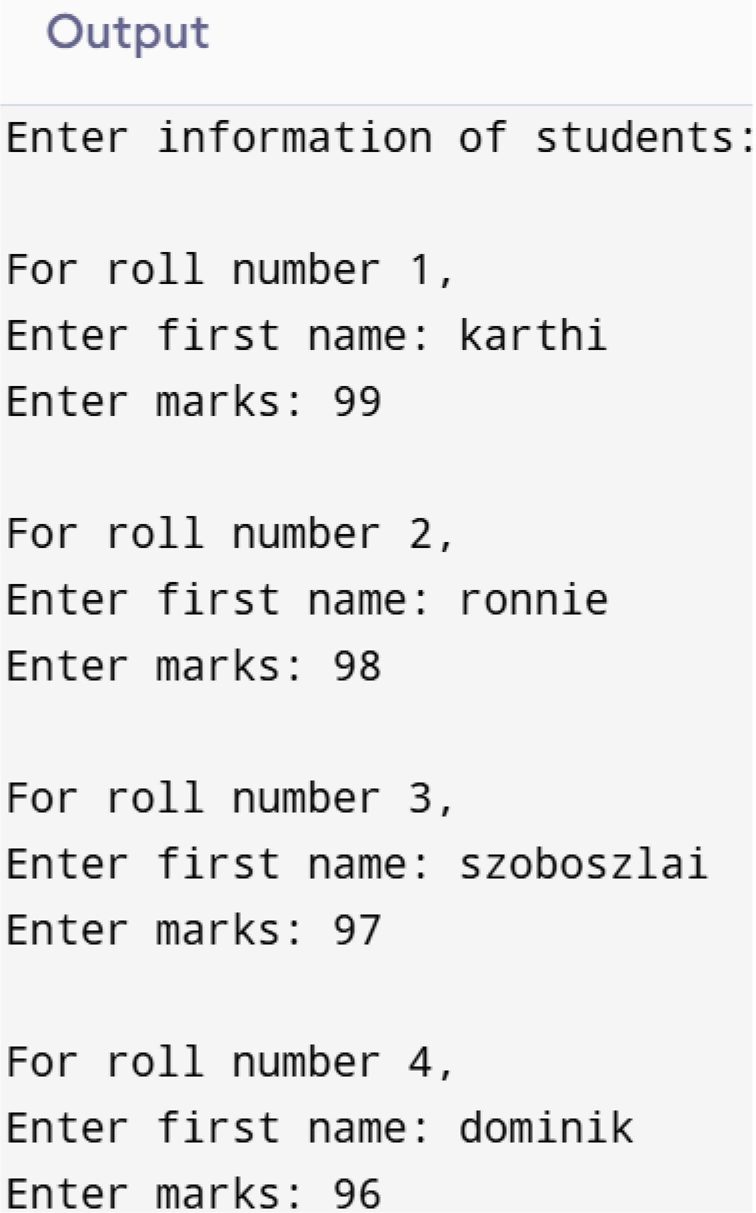
```
Name : Tata
Price : 1021
```

4. Write a program to store multiple student records using array of structures.
IPO
Input:Enter a value as input.
Process: To store multiple student records using array of structures.
Output:output the variable

Program:

```
#include <stdio.h>
struct student
{
    char firstName[50];
    int roll;
    float marks;
} s[5];
int main()
{
    int i;
    printf("Enter information of students:\n");
    for (i = 0; i < 5; ++i)
    {
        s[i].roll = i + 1;
        printf("\nFor roll number %d,\n", s[i].roll);
        printf("Enter first name: ");
        scanf("%s", s[i].firstName);
        printf("Enter marks: ");
        scanf("%f", &s[i].marks);
    }
    printf("\nDisplaying Information:\n");
    for (i = 0; i < 5; ++i) {
        printf("\nRoll number: %d\n", s[i].roll);
        printf("First name: ");
        puts(s[i].firstName);
        printf("Marks: %.1f\n", s[i].marks);
    }
    return 0;
}
```

Output



Output

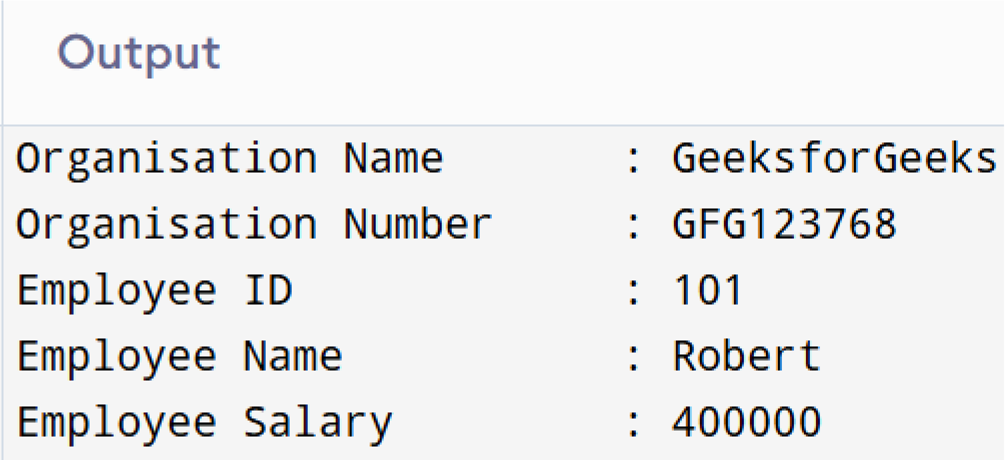
```
Enter information of students:

For roll number 1,
Enter first name: karthi
Enter marks: 99

For roll number 2,
Enter first name: ronnie
Enter marks: 98

For roll number 3,
Enter first name: szoboszlai
Enter marks: 97

For roll number 4,
Enter first name: dominik
Enter marks: 96
```

5. Write a program to demonstrate nested structures.
IPO
Input:Enter a value as input.
Process: To demonstrate nested structures.
Output:output the variable

Program:

```
#include <stdio.h>
#include <string.h>
struct Employee
{
   int employee_id;
   char name[20];
   int salary;
};
struct Organisation
{
   char organisation_name[20];
   char org_number[20];
   struct Employee emp;
};

int main() {
   struct Organisation org;
   org.emp.employee_id = 101;
   strcpy(org.emp.name, "Robert");
   org.emp.salary = 400000;
   strcpy(org.organisation_name, "GeeksforGeeks");
   strcpy(org.org_number, "GFG123768");
   printf("Organisation Name      : %s\n", org.organisation_name);
   printf("Organisation Number    : %s\n", org.org_number);
   printf("Employee ID            : %d\n", org.emp.employee_id);
   printf("Employee Name          : %s\n", org.emp.name);
   printf("Employee Salary        : %d\n", org.emp.salary);

   return 0;
}
```

Output:



Output

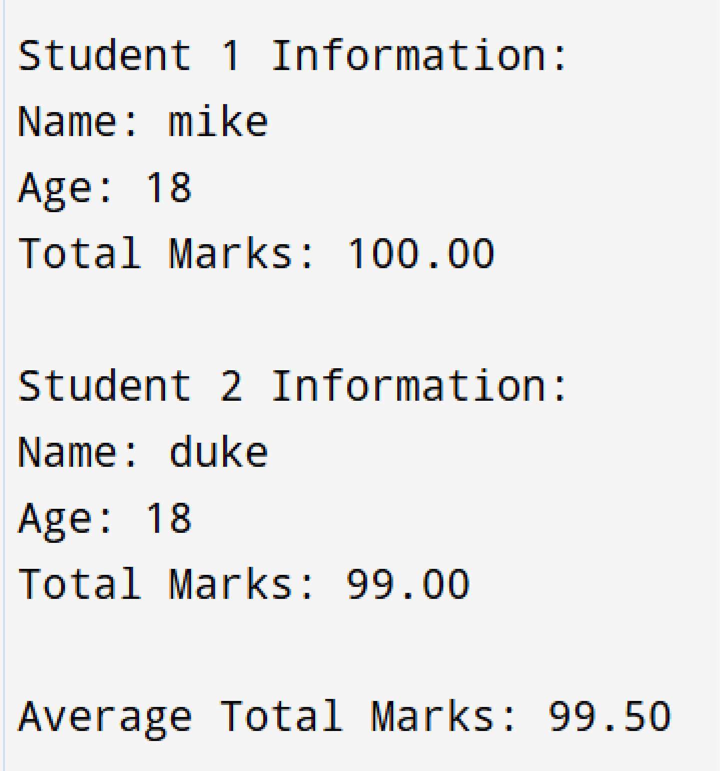
```
Organisation Name        : GeeksforGeeks
Organisation Number      : GFG123768
Employee ID              : 101
Employee Name            : Robert
Employee Salary          : 400000
```

6. Write a program to calculate total and average marks using structures.
IPO
Input:Enter a value as input.
Process: To calculate total and average marks using structures.
Output:output the variable

Program:

```
#include <stdio.h>
struct Student {
    char name[50];
    int age;
    float marks;
};
int main()
{
    struct Student student1, student2;
    float averageMarks;
    printf("Input details for Student 1:\n");
    printf("Name: ");
    scanf("%s", student1.name);
    printf("Age: ");
    scanf("%d", &student1.age);
    printf("Total Marks: ");
    scanf("%f", &student1.marks);
    printf("\nInput details for Student 2:\n");
    printf("Name: ");
    scanf("%s", student2.name);
    printf("Age: ");
    scanf("%d", &student2.age);
    printf("Total Marks: ");
    scanf("%f", &student2.marks);
    printf("\nStudent 1 Information:\n");
    printf("Name: %s\n", student1.name);
    printf("Age: %d\n", student1.age);
    printf("Total Marks: %.2f\n", student1.marks);
    printf("\nStudent 2 Information:\n");
    printf("Name: %s\n", student2.name);
    printf("Age: %d\n", student2.age);
    printf("Total Marks: %.2f\n", student2.marks);
    averageMarks = (student1.marks + student2.marks) / 2.0f;
    printf("\nAverage Total Marks: %.2f\n", averageMarks);

    return 0;
}
```

Output



```
Student 1 Information:
Name: mike
Age: 18
Total Marks: 100.00

Student 2 Information:
Name: duke
Age: 18
Total Marks: 99.00

Average Total Marks: 99.50
```

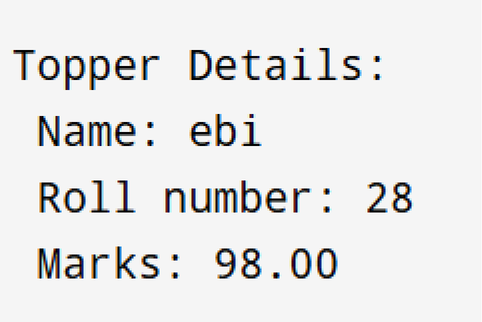
7. Write a program to find the highest marks among students.
IPO
Input:Enter a value as input.
Process: To find the highest marks among students
Output:output the variable

Program:

```
#include <stdio.h>
#include <string.h>
#define MAX_STUDENTS 5
```

```
struct Student
{
    char name[50];
    int roll;
    float marks;
};
int main()
{
    struct Student students[MAX_STUDENTS];
    int i, topIndex = 0;
    for (i = 0; i < MAX_STUDENTS; ++i) {
        printf("Enter details for student %d:\n", i + 1);
        printf(" Name: ");
        scanf("%s", students[i].name);
        printf(" Roll number: ");
        scanf("%d", &students[i].roll);
        printf(" Marks: ");
        scanf("%f", &students[i].marks);
    }
    for (i = 1; i < MAX_STUDENTS; ++i) {
        if (students[i].marks > students[topIndex].marks) {
            topIndex = i;
        }
    }
    printf("\nTopper Details:\n");
    printf(" Name: %s\n", students[topIndex].name);
    printf(" Roll number: %d\n", students[topIndex].roll);
    printf(" Marks: %.2f\n", students[topIndex].marks);

    return 0;
}
```

Output



```
Topper Details:
 Name: ebi
 Roll number: 28
 Marks: 98.00
```

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

IPO

Input:Enter a value as input.

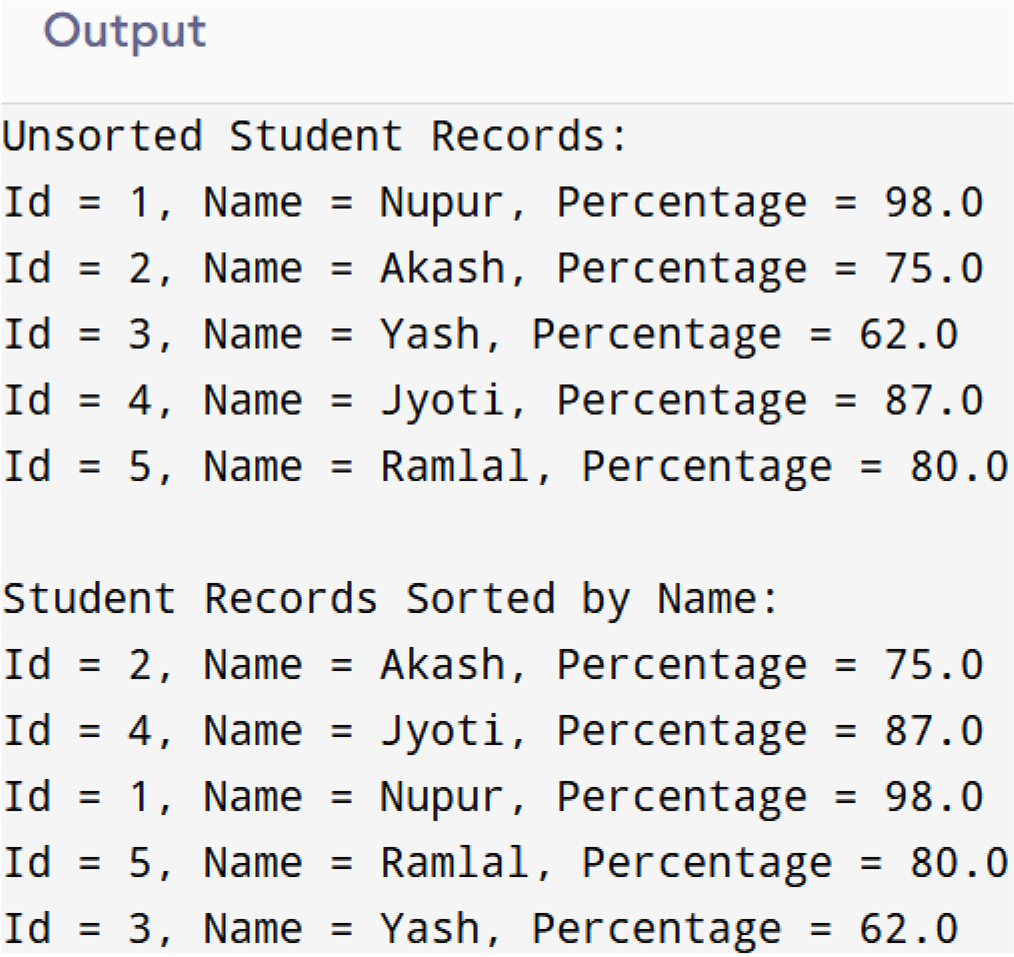
Process: To sort student records by name using structure.

Output:output the variable

Program:

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

struct Student {
   int student_id;
   char student_name[50];
   float student_percentage;
};

int compareByName(const void *a, const void *b) {
   const struct Student *s1 = (const struct Student *)a;
   const struct Student *s2 = (const struct Student *)b;
   return strcmp(s1->student_name, s2->student_name);
}

int main() {
   int n = 5;
   struct Student arr[5] = {
      {1, "Nupur", 98.0},
      {2, "Akash", 75.0},
      {3, "Yash", 62.0},
      {4, "Jyoti", 87.0},
      {5, "Ramlal", 80.0}
   };

   printf("Unsorted Student Records:\n");
   for (int i = 0; i < n; ++i) {
      printf("Id = %d, Name = %s, Percentage = %.1f\n",
           arr[i].student_id, arr[i].student_name, arr[i].student_percentage);
   }

   qsort(arr, n, sizeof(struct Student), compareByName);

   printf("\nStudent Records Sorted by Name:\n");
   for (int i = 0; i < n; ++i) {
      printf("Id = %d, Name = %s, Percentage = %.1f\n",
           arr[i].student_id, arr[i].student_name, arr[i].student_percentage);
   }

   return 0;
}
```

Output



Output

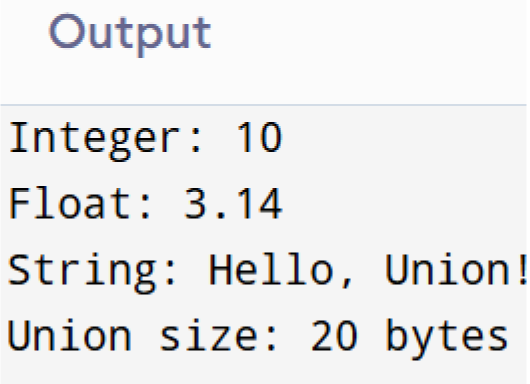
```
Unsorted Student Records:
Id = 1, Name = Nupur, Percentage = 98.0
Id = 2, Name = Akash, Percentage = 75.0
Id = 3, Name = Yash, Percentage = 62.0
Id = 4, Name = Jyoti, Percentage = 87.0
Id = 5, Name = Ramlal, Percentage = 80.0

Student Records Sorted by Name:
Id = 2, Name = Akash, Percentage = 75.0
Id = 4, Name = Jyoti, Percentage = 87.0
Id = 1, Name = Nupur, Percentage = 98.0
Id = 5, Name = Ramlal, Percentage = 80.0
Id = 3, Name = Yash, Percentage = 62.0
```

9. Write a program using union to store data of different types.
IPO
Input:Enter a value as input.
Process: To store data of different type
Output:output the variable

Program:

```
#include <stdio.h>
#include <string.h>
union Data
{
    int i;
    float f;
    char str[20];
};
int main()
{
    union Data data;
    data.i = 10;
    printf("Integer: %d\n", data.i);

    data.f = 3.14f;
    printf("Float: %.2f\n", data.f);

    strcpy(data.str, "Hello, Union!");
    printf("String: %s\n", data.str);

    printf("Union size: %lu bytes\n", sizeof(data));

    return 0;
}
```

Output



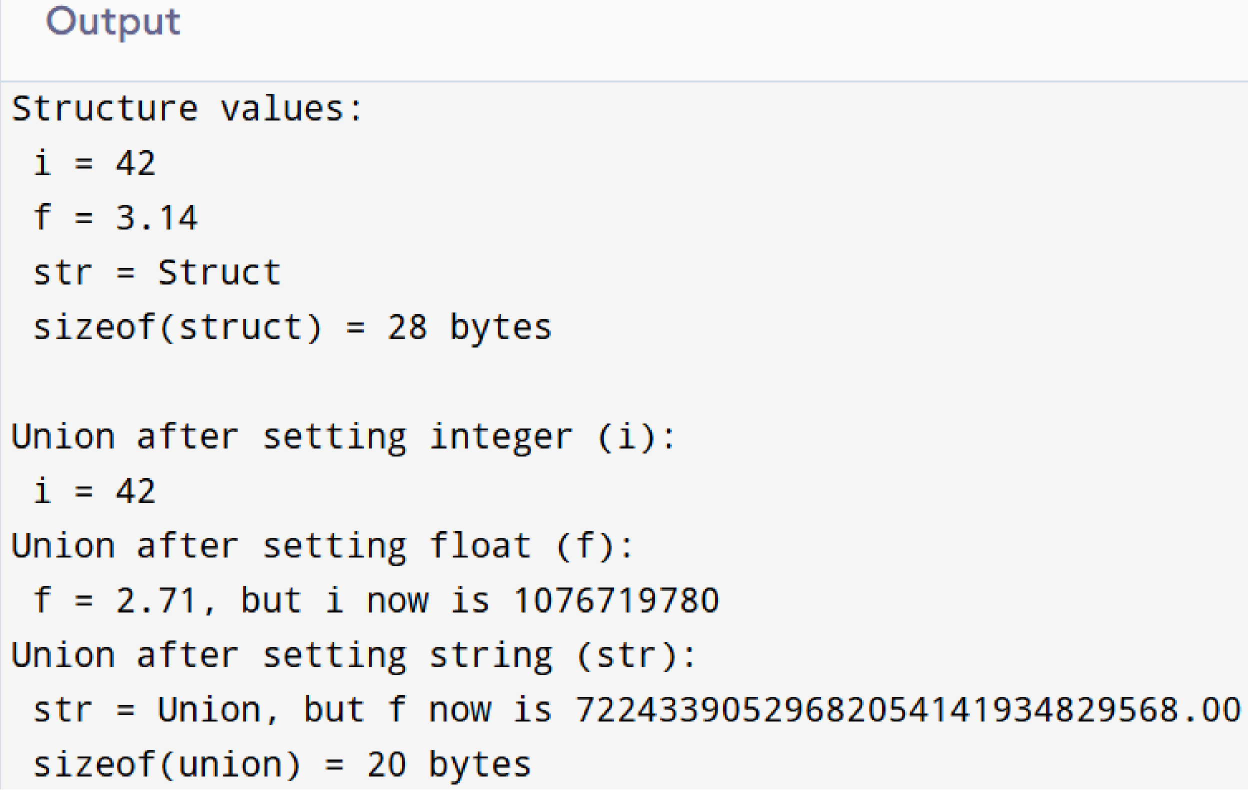
```
Integer: 10
Float: 3.14
String: Hello, Union!
Union size: 20 bytes
```

10. Compare and contrast structure vs union with a sample program.
IPO
Input:Enter a value as input.
Process: To Compare and contrast structure vs union
Output:output the variable

Program:

```
#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 = 42;
    s.f = 3.14f;
    strcpy(s.str, "Struct");

    printf("Structure values:\n");
    printf(" i = %d\n f = %.2f\n str = %s\n", s.i, s.f, s.str);
    printf(" sizeof(struct) = %zu bytes\n\n", sizeof(s));


    u.i = 42;
    printf("Union after setting integer (i):\n i = %d\n", u.i);

    u.f = 2.71f;
    printf("Union after setting float (f):\n f = %.2f, but i now is %d\n", u.f, u.i);

    strcpy(u.str, "Union");
    printf("Union after setting string (str):\n str = %s, but f now is %.2f\n", u.str, u.f);
    printf(" sizeof(union) = %zu bytes\n", sizeof(u));

    return 0;
}
```

Output



```
Structure values:
 i = 42
 f = 3.14
 str = Struct
 sizeof(struct) = 28 bytes

Union after setting integer (i):
 i = 42
Union after setting float (f):
 f = 2.71, but i now is 1076719780
Union after setting string (str):
 str = Union, but f now is 72243390529682054141934829568.00
 sizeof(union) = 20 bytes
```