Practical File
of
Fundamentals of C Programming
23CS003

Submitted

in partial fulfillment for the award of the degree

of

BACHELEOR OF ENGINEERING

in

COMPUTER SCIENCE & ENGINEERING

CHITKARA UNIVERSITY

CHANDIGARH-PATIALA NATIONAL HIGHWAY
RAJPURA (PATIALA) PUNJAB-140401 (INDIA)

May, 2024

Submitted To:	**Submitted By:**
Dr. Preeti Sharma	**Komal**
Assistant Professor	**2310992042**

Chitkara University, Punjab **Sem-II (2023-2027)**

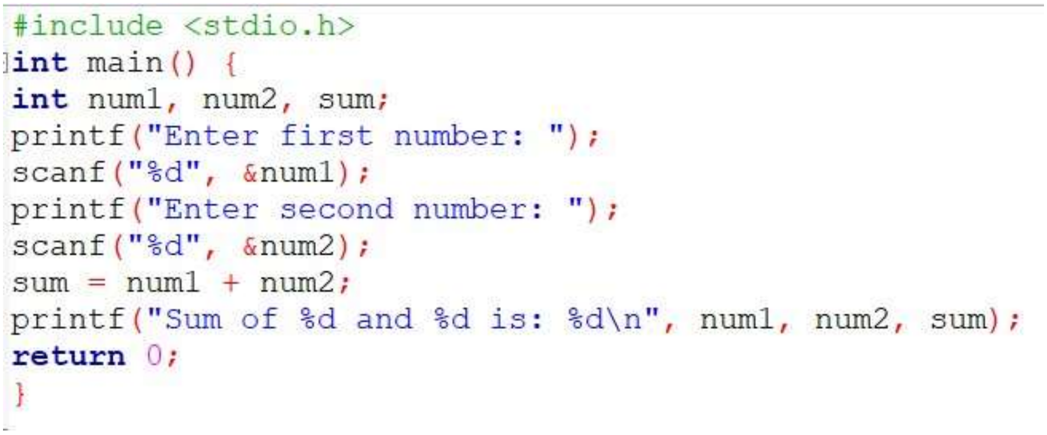
INDEX

11	Write the program to count the digits in a number and then print the reverse of the number also.	
12	Write a program to generate the Fibonacci series.	
13	Write a program to print the following patterns:	
	a) * b) *	
14	Write the program to print the following pattern: 1 2 3 4 5 6 2 4 6 8 10 12 3 6 9 12 15 18 4 8 12 16 20 24 5 10 15 20 25 30 6 12 18 24 30 36	
15	Write a program to check that the given number is prime, Armstrong or perfect using the concept of functions.	
16	Write a program to calculate the area and circumference of a circle using functions.	
17	Write a program to swap two variables using the concept of call by value and call by reference.	
18	Write a program to perform the following operations on 1D-Array: • Insert • Update • Delete • Display • Search	
19	Write a program to calculate the sum of array elements by passing it to a function.	

20	Write a program to show the use of passing pointer as arguments to the functions.	
21	Write a program matrix multiplication using the concept of 2D array.	
22	Write a program to transpose a given matrix.	
23	Write a program to find the factorial of a number by using the concept of recursion.	
24	Write a menu driven C program to show the use of in-built string functions like strlen, strcat, strcpy, strcmp, strrev etc.	
25	Write a program to display the sum of the digits of a number by using the concept of recursion.	
26	Write a C program to add two distances in inch & feet using the concept of structures.	
27	Write a C program to add two complex numbers using the concept of structures in C	
28	Write a program in C to store the information of five employees using both concepts i.e. array of structure and array within structure.	
29	Write a Program in C to store and retrieve the information about students of a university by using the concept of file handling.	
30	Write a Program in C to find and replace a specific string in a file and also display the total number of appearances of that string.	

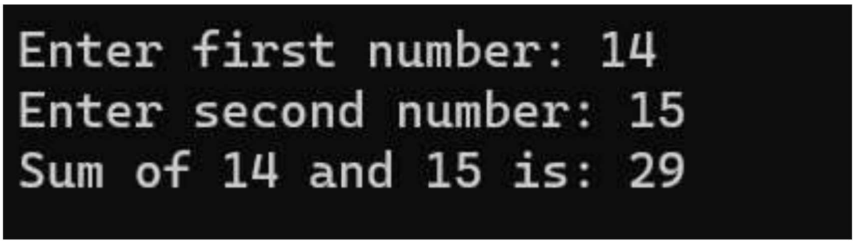
Program 1: Write a Program to show the use to input (Scanf)/output (Printf) statements and block structure of C-program by highlighting the features of "stdio.h".

Solution:



```
#include <stdio.h>
int main() {
int num1, num2, sum;
printf("Enter first number: ");
scanf("%d", &num1);
printf("Enter second number: ");
scanf("%d", &num2);
sum = num1 + num2;
printf("Sum of %d and %d is: %d\n", num1, num2, sum);
return 0;
}
```

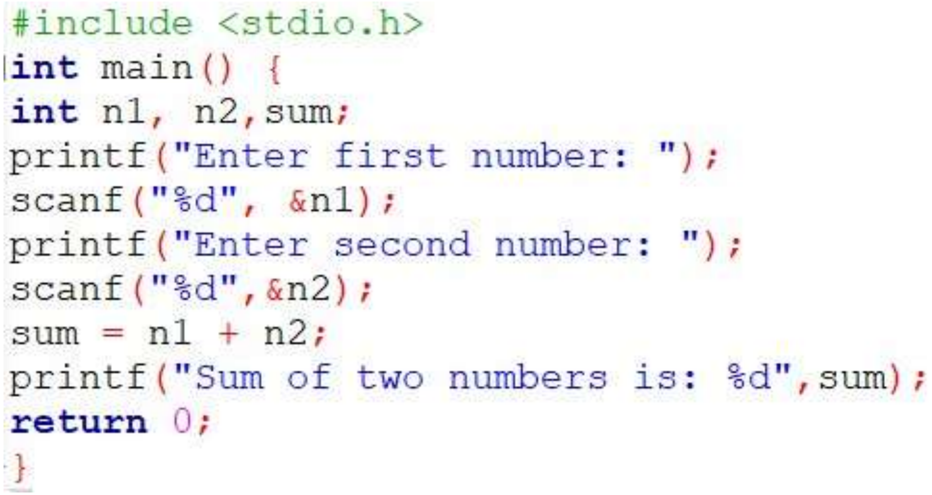
Output:



```
Enter first number: 14
Enter second number: 15
Sum of 14 and 15 is: 29
```

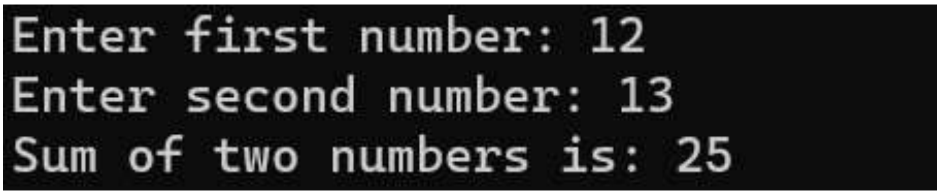
Program 2: Write a program to add two numbers and display the sum.

Solution:



```
#include <stdio.h>
int main() {
int n1, n2,sum;
printf("Enter first number: ");
scanf("%d", &n1);
printf("Enter second number: ");
scanf("%d",&n2);
sum = n1 + n2;
printf("Sum of two numbers is: %d",sum);
return 0;
}
```

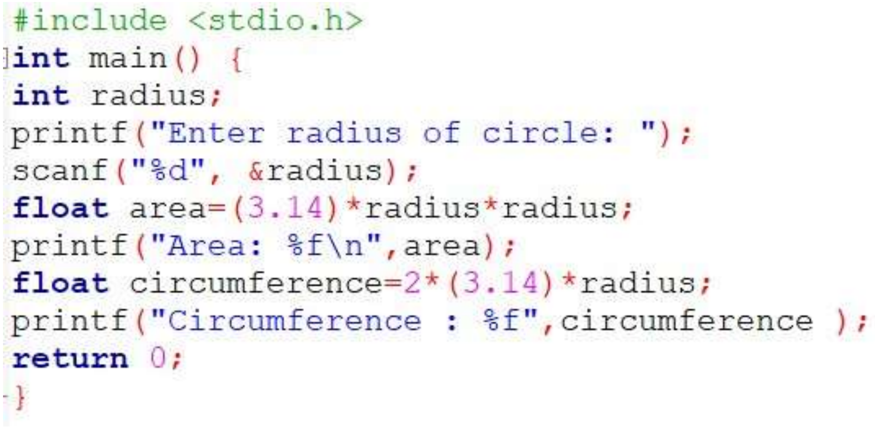
<u>Output</u>:



```
Enter first number: 12
Enter second number: 13
Sum of two numbers is: 25
```

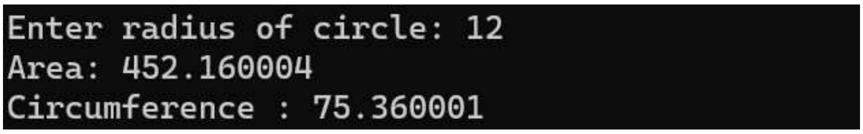
Program 3: Write a program to calculate the area and the circumference of a circle by using radius as the input provided by the user.

Solution:



```
#include <stdio.h>
int main() {
int radius;
printf("Enter radius of circle: ");
scanf("%d", &radius);
float area=(3.14)*radius*radius;
printf("Area: %f\n",area);
float circumference=2*(3.14)*radius;
printf("Circumference : %f",circumference );
return 0;
}
```

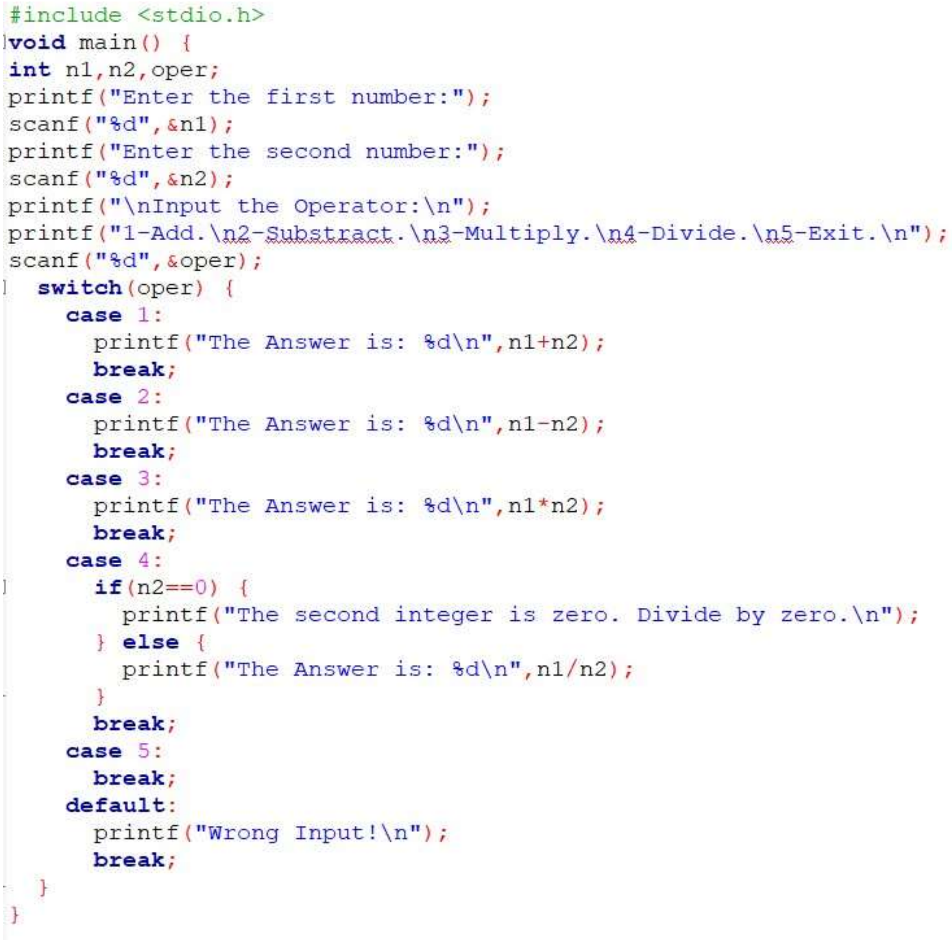
Output:



```
Enter radius of circle: 12
Area: 452.160004
Circumference : 75.360001
```

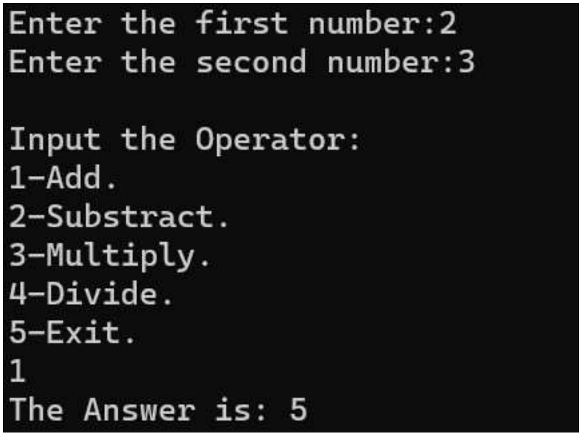
Program 4: Write a Program to perform addition, subtraction, division and multiplication of two numbers given as input by the user.

<u>Solution</u>:



```
#include <stdio.h>
void main() {
int n1,n2,oper;
printf("Enter the first number:");
scanf("%d",&n1);
printf("Enter the second number:");
scanf("%d",&n2);
printf("\nInput the Operator:\n");
printf("1-Add.\n2-Substract.\n3-Multiply.\n4-Divide.\n5-Exit.\n");
scanf("%d",&oper);
  switch(oper) {
    case 1:
      printf("The Answer is: %d\n",n1+n2);
      break;
    case 2:
      printf("The Answer is: %d\n",n1-n2);
      break;
    case 3:
      printf("The Answer is: %d\n",n1*n2);
      break;
    case 4:
      if(n2==0) {
        printf("The second integer is zero. Divide by zero.\n");
      } else {
        printf("The Answer is: %d\n",n1/n2);
      }
      break;
    case 5:
      break;
    default:
      printf("Wrong Input!\n");
      break;
  }
}
```

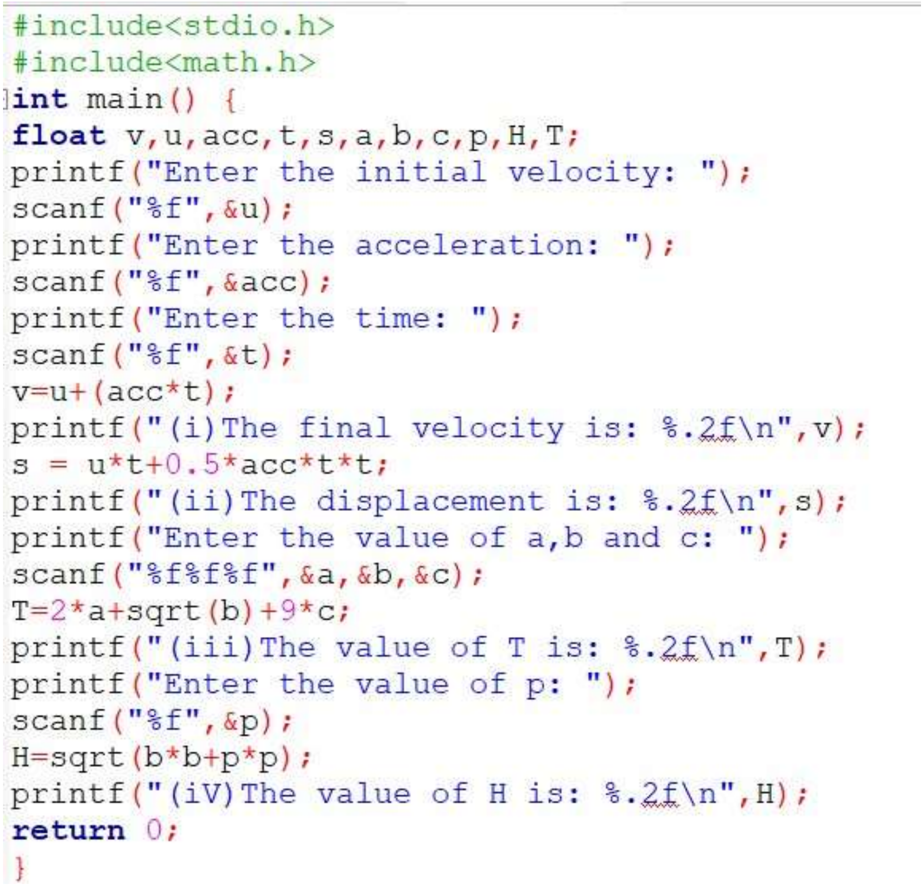
Output:



```
Enter the first number:2
Enter the second number:3

Input the Operator:
1-Add.
2-Substract.
3-Multiply.
4-Divide.
5-Exit.
1
The Answer is: 5
```

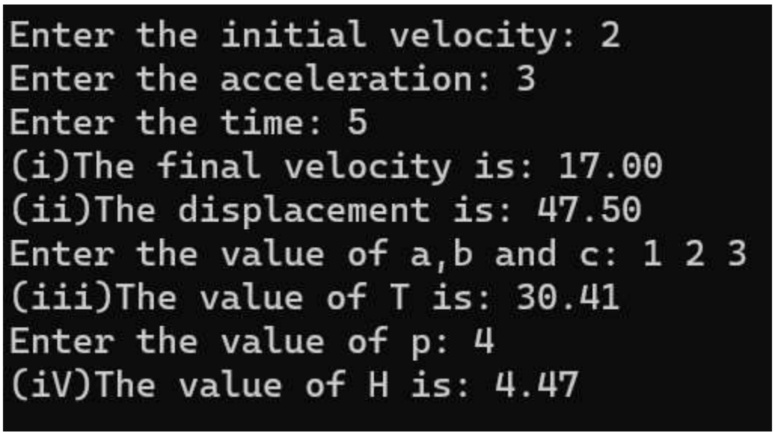
Program 5: Write a program to evaluate each of the following equations.
(i) $V = u + at$. (ii) $S = ut+1/2at^2$ (iii) $T=2*a+\surd b+9c$ (iv) $H=\surd b^2+p^2$

Solution:



```c
#include<stdio.h>
#include<math.h>
int main() {
float v,u,acc,t,s,a,b,c,p,H,T;
printf("Enter the initial velocity: ");
scanf("%f",&u);
printf("Enter the acceleration: ");
scanf("%f",&acc);
printf("Enter the time: ");
scanf("%f",&t);
v=u+(acc*t);
printf("(i)The final velocity is: %.2f\n",v);
s = u*t+0.5*acc*t*t;
printf("(ii)The displacement is: %.2f\n",s);
printf("Enter the value of a,b and c: ");
scanf("%f%f%f",&a,&b,&c);
T=2*a+sqrt(b)+9*c;
printf("(iii)The value of T is: %.2f\n",T);
printf("Enter the value of p: ");
scanf("%f",&p);
H=sqrt(b*b+p*p);
printf("(iV)The value of H is: %.2f\n",H);
return 0;
}
```

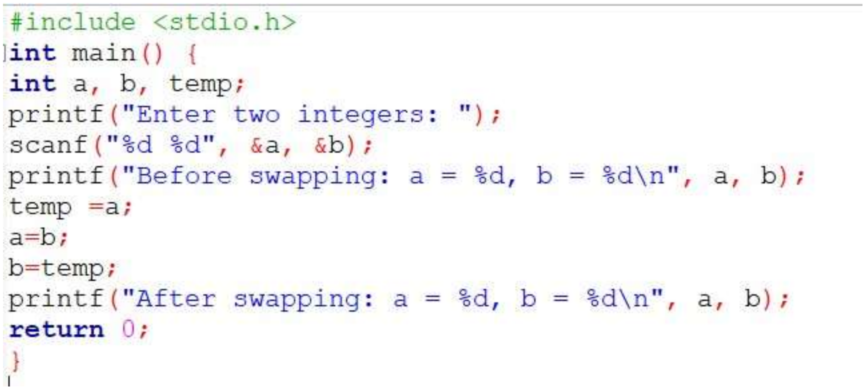
Output:



```
Enter the initial velocity: 2
Enter the acceleration: 3
Enter the time: 5
(i)The final velocity is: 17.00
(ii)The displacement is: 47.50
Enter the value of a,b and c: 1 2 3
(iii)The value of T is: 30.41
Enter the value of p: 4
(iV)The value of H is: 4.47
```

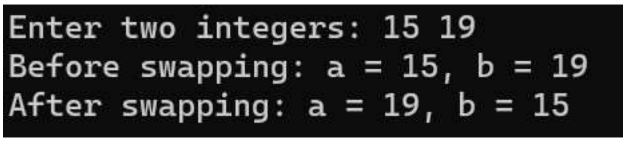
Program 6: Write a program to swap two variable:

a) By using temporary variable. Solution:



```
#include <stdio.h>
int main() {
int a, b, temp;
printf("Enter two integers: ");
scanf("%d %d", &a, &b);
printf("Before swapping: a = %d, b = %d\n", a, b);
temp =a;
a=b;
b=temp;
printf("After swapping: a = %d, b = %d\n", a, b);
return 0;
}
```

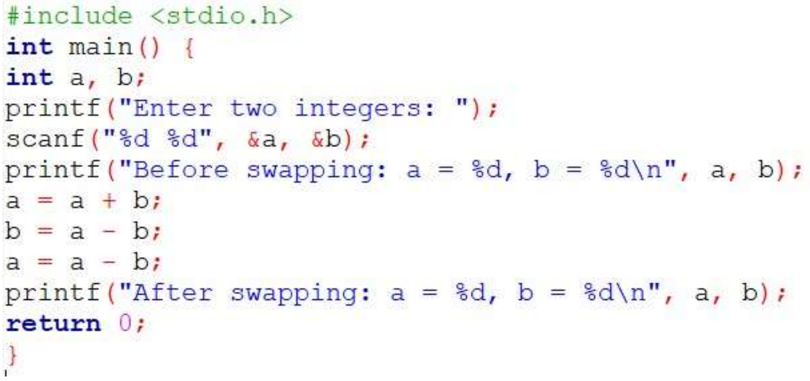
Output:



```
Enter two integers: 15 19
Before swapping: a = 15, b = 19
After swapping: a = 19, b = 15
```

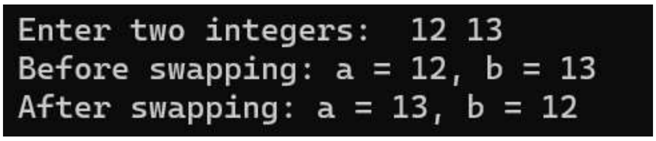
b) Without using temporary variable

Solution:



```
#include <stdio.h>
int main() {
int a, b;
printf("Enter two integers: ");
scanf("%d %d", &a, &b);
printf("Before swapping: a = %d, b = %d\n", a, b);
a = a + b;
b = a - b;
a = a - b;
printf("After swapping: a = %d, b = %d\n", a, b);
return 0;
}
```

Output:

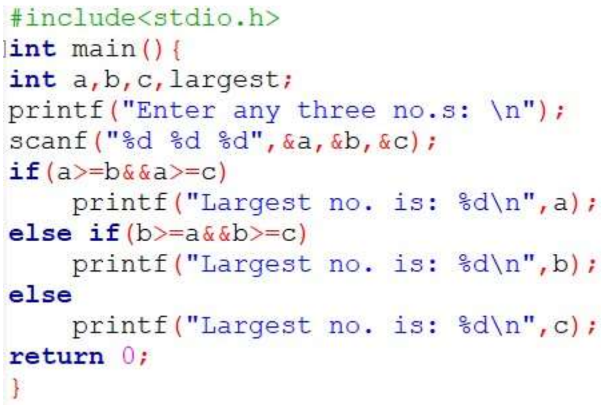


```
Enter two integers:  12 13
Before swapping: a = 12, b = 13
After swapping: a = 13, b = 12
```

Program 7: Write a Program to find the greatest among three numbers using:

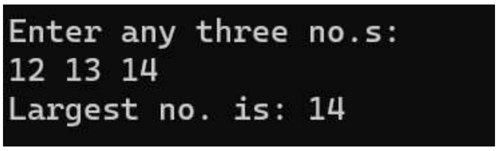
- **Conditional Operator**
- **If-Else statement**

Solution:



```
#include<stdio.h>
int main(){
int a,b,c,largest;
printf("Enter any three no.s: \n");
scanf("%d %d %d",&a,&b,&c);
if(a>=b&&a>=c)
    printf("Largest no. is: %d\n",a);
else if(b>=a&&b>=c)
    printf("Largest no. is: %d\n",b);
else
    printf("Largest no. is: %d\n",c);
return 0;
}
```

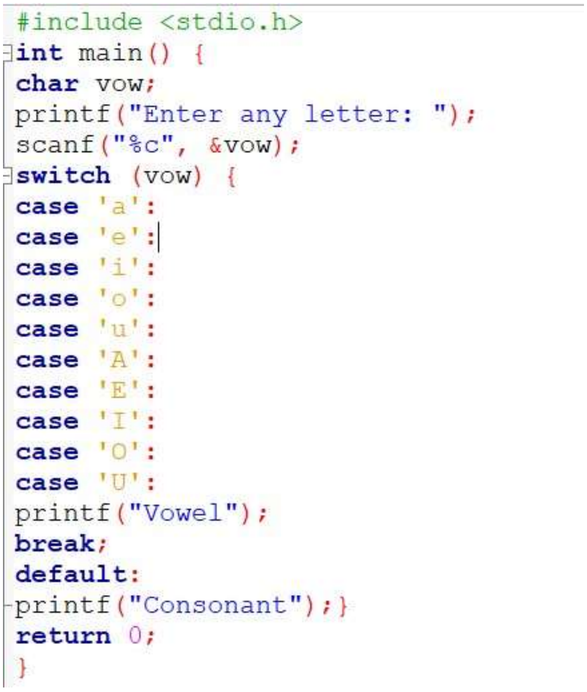
Output:



```
Enter any three no.s:
12 13 14
Largest no. is: 14
```

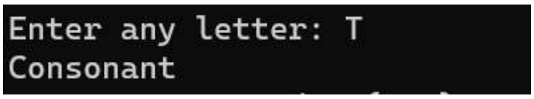
Program 8: Write the following programs using switch case statement:

- **To check that an input alphabet is vowel or consonant Solution**:



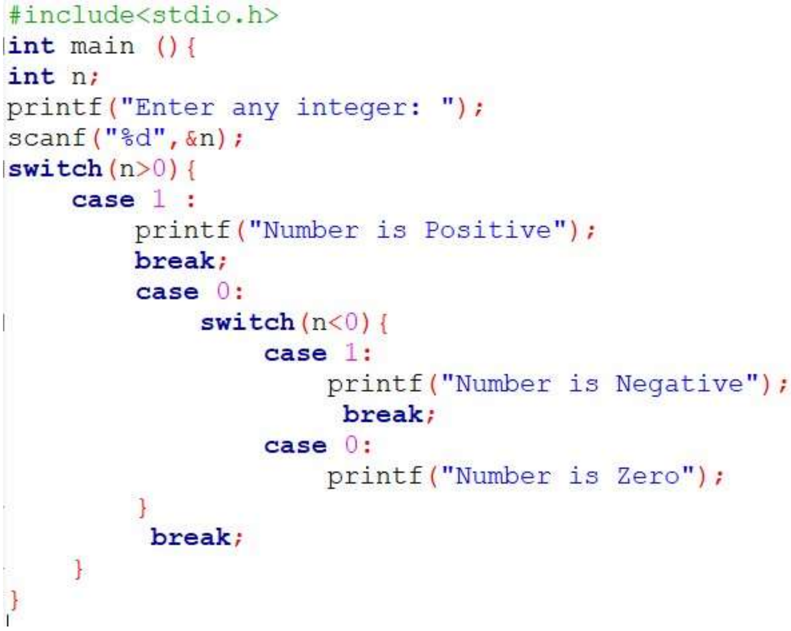
```c
#include <stdio.h>
int main() {
char vow;
printf("Enter any letter: ");
scanf("%c", &vow);
switch (vow) {
case 'a':
case 'e':
case 'i':
case 'o':
case 'u':
case 'A':
case 'E':
case 'I':
case 'O':
case 'U':
printf("Vowel");
break;
default:
printf("Consonant");}
return 0;
}
```

<u>Output</u>:



```
Enter any letter: T
Consonant
```

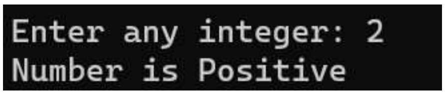
- **To check whether a number is positive, negative or zero <u>Solution</u>**:



```c
#include<stdio.h>
int main (){
int n;
printf("Enter any integer: ");
scanf("%d",&n);
switch(n>0){
    case 1 :
        printf("Number is Positive");
        break;
        case 0:
            switch(n<0){
                case 1:
                    printf("Number is Negative");
                     break;
                case 0:
                    printf("Number is Zero");
        }
         break;
    }
}
```

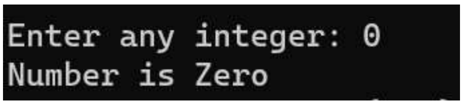
Output:

Case 1:



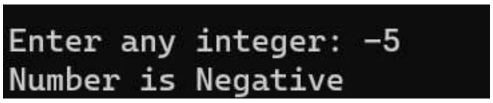
```
Enter any integer: 2
Number is Positive
```

Case 2:



```
Enter any integer: 0
Number is Zero
```

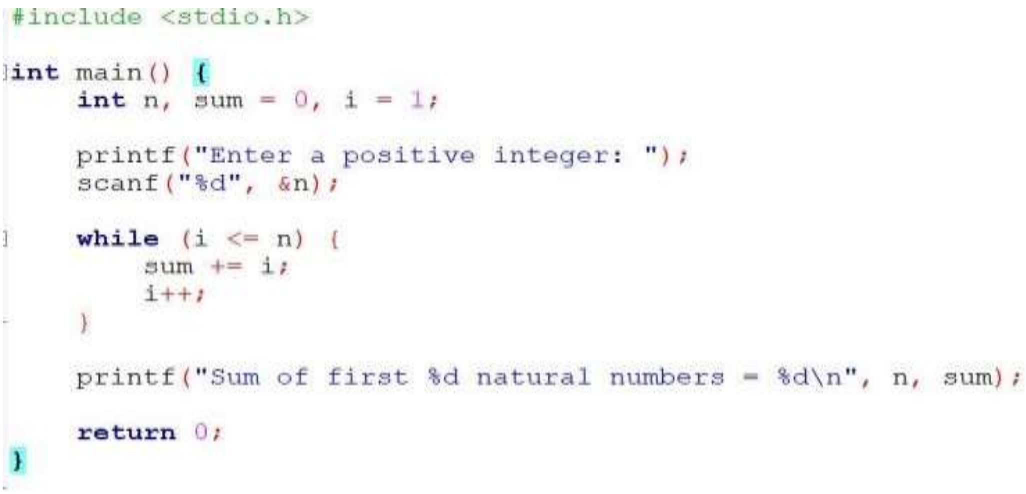
Case 3:



```
Enter any integer: -5
Number is Negative
```

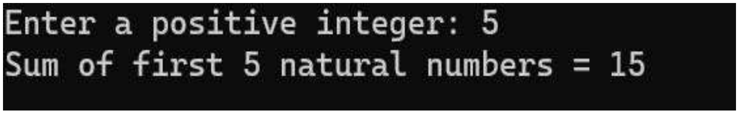
Program 9: Write a program using while loop to print the sum of first n natural numbers.

Solution:



```
#include <stdio.h>

int main() {
    int n, sum = 0, i = 1;

    printf("Enter a positive integer: ");
    scanf("%d", &n);

    while (i <= n) {
        sum += i;
        i++;
    }

    printf("Sum of first %d natural numbers = %d\n", n, sum);

    return 0;
}
```

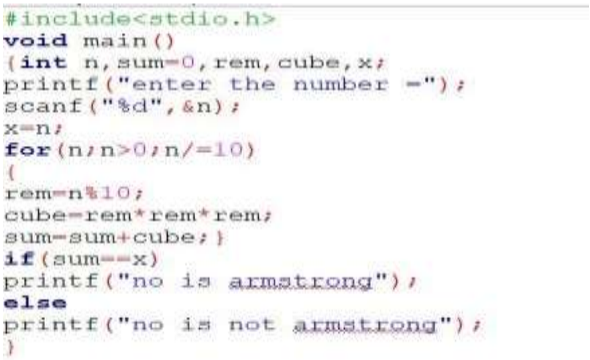
Output:



```
Enter a positive integer: 5
Sum of first 5 natural numbers = 15
```

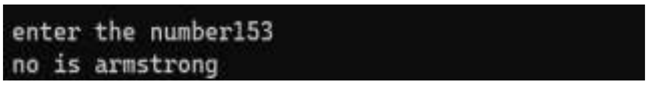
Program 10: Write a Program to check a number is Armstrong or not using For loop.

Solution:



```
#include<stdio.h>
void main()
{int n,sum=0,rem,cube,x;
printf("enter the number -");
scanf("%d",&n);
x=n;
for(n;n>0;n/=10)
{
rem=n%10;
cube=rem*rem*rem;
sum=sum+cube;}
if(sum==x)
printf("no is armstrong");
else
printf("no is not armstrong");
}
```

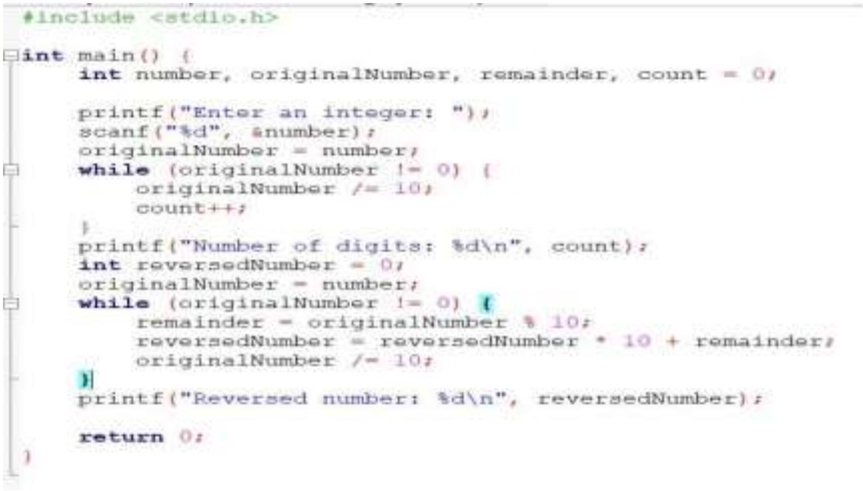
Output:



```
enter the number153
no is armstrong
```

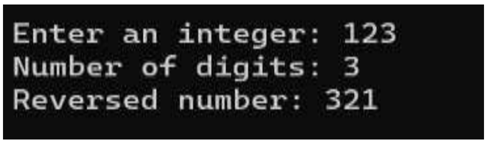
Program 1

1: Write a Program to count the digits in a number and then print the reverse of the number also. Solution:



```c
#include <stdio.h>

int main() {
    int number, originalNumber, remainder, count = 0;

    printf("Enter an integer: ");
    scanf("%d", &number);
    originalNumber = number;
    while (originalNumber != 0) {
        originalNumber /= 10;
        count++;
    }
    printf("Number of digits: %d\n", count);
    int reversedNumber = 0;
    originalNumber = number;
    while (originalNumber != 0) {
        remainder = originalNumber % 10;
        reversedNumber = reversedNumber * 10 + remainder;
        originalNumber /= 10;
    }
    printf("Reversed number: %d\n", reversedNumber);

    return 0;
}
```

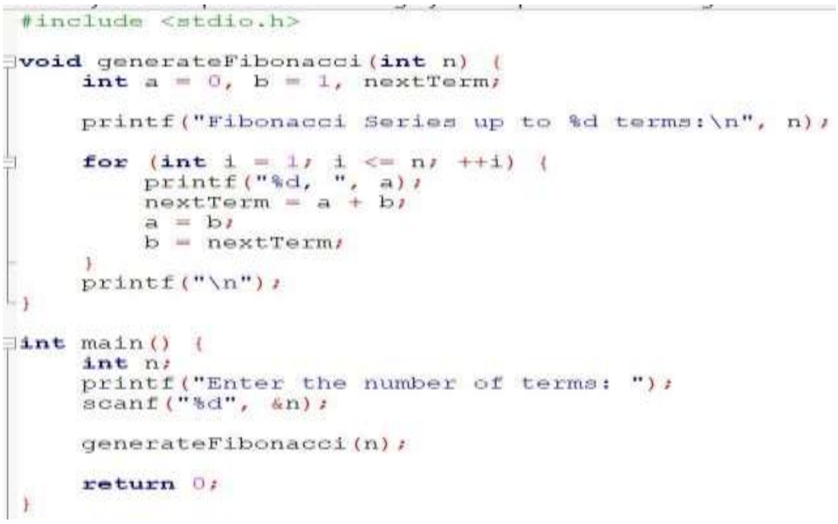
Output:



```
Enter an integer: 123
Number of digits: 3
Reversed number: 321
```

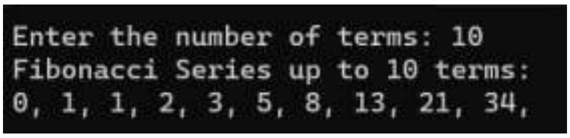
Program 12: Write a program to generate the Fibonacci series. Solution:

Program 1



```
#include <stdio.h>

void generateFibonacci(int n) {
    int a = 0, b = 1, nextTerm;

    printf("Fibonacci Series up to %d terms:\n", n);

    for (int i = 1; i <= n; ++i) {
        printf("%d, ", a);
        nextTerm = a + b;
        a = b;
        b = nextTerm;
    }
    printf("\n");
}

int main() {
    int n;
    printf("Enter the number of terms: ");
    scanf("%d", &n);

    generateFibonacci(n);

    return 0;
}
```

Output:



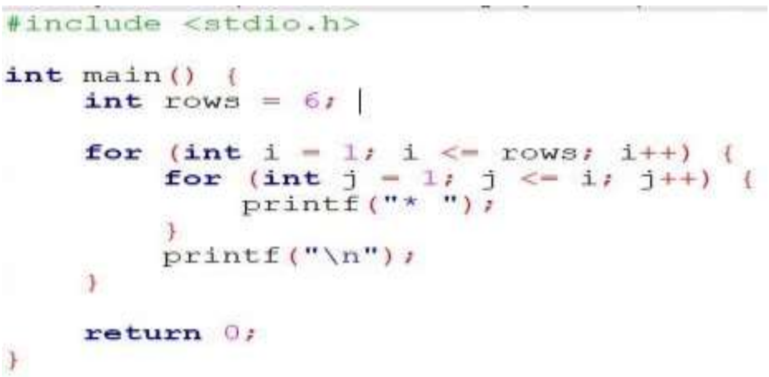
```
Enter the number of terms: 10
Fibonacci Series up to 10 terms:
0, 1, 1, 2, 3, 5, 8, 13, 21, 34,
```

3: Write a program to print the following patterns:

```
a) *
* *
* * *
* * * *
* * * * *
* * * * * *
```

```
b)         *
* *
  *      * *
 *      * * *
      * * * * *
*     * * * * *
```

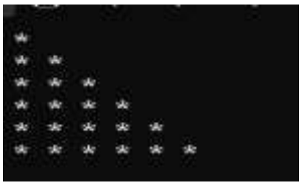
Solution:



```
#include <stdio.h>

int main() {
    int rows = 6; |

    for (int i = 1; i <= rows; i++) {
        for (int j = 1; j <= i; j++) {
            printf("* ");
        }
        printf("\n");
    }

    return 0;
}
```

Program 1

Output:



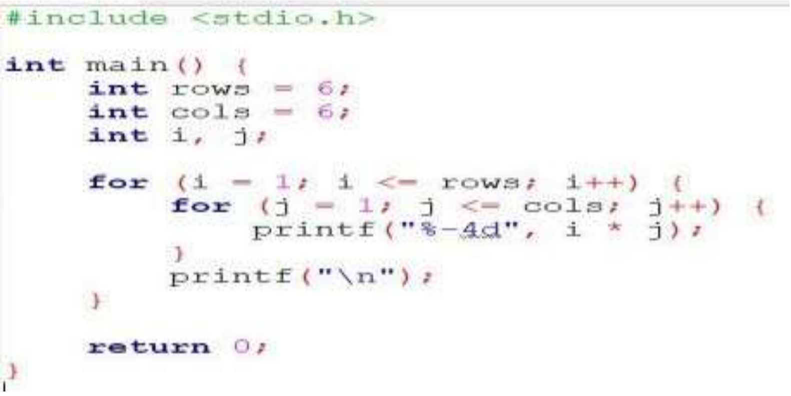
```
*
*  *
*  *  *
*  *  *  *
*  *  *  *  *
*  *  *  *  *  *
```

4: Write the program to print the following pattern:

1 2 3 4 5 6
2 4 6 8 10 12
3 6 9 12 15 18
4 8 12 16 20 24
5 10 15 20 25 30
6 12 18 24 30 36

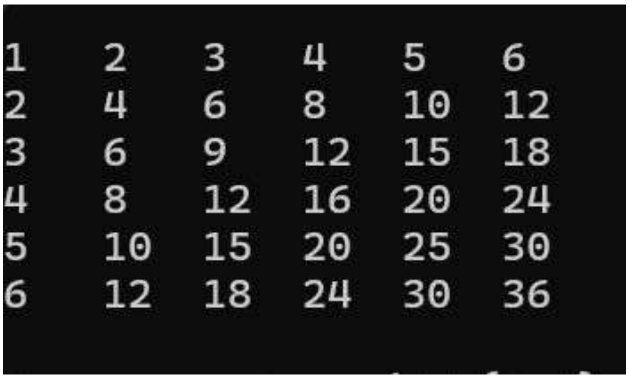
Solution:

Program 1



```c
#include <stdio.h>

int main() {
    int rows = 6;
    int cols = 6;
    int i, j;

    for (i = 1; i <= rows; i++) {
        for (j = 1; j <= cols; j++) {
            printf("%-4d", i * j);
        }
        printf("\n");
    }

    return 0;
}
```

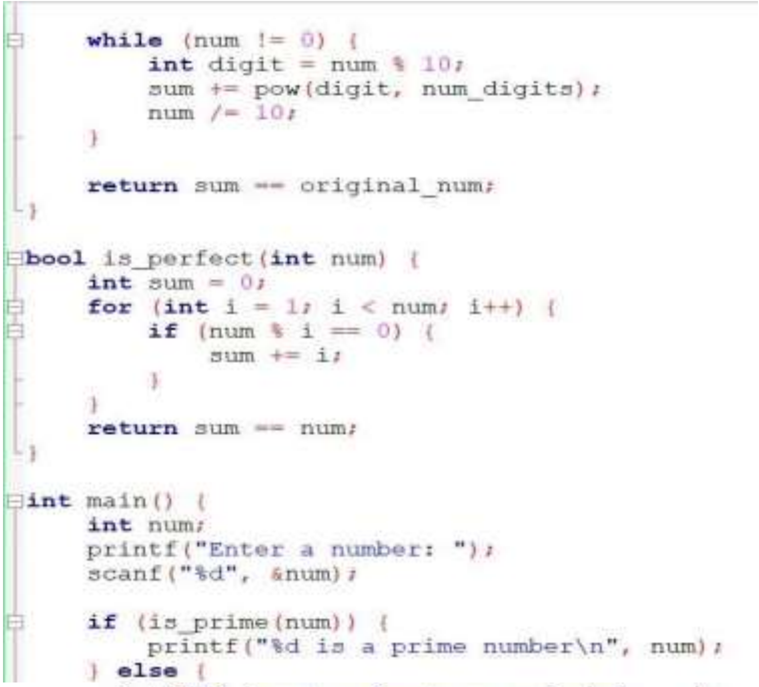
<u>Output</u>:



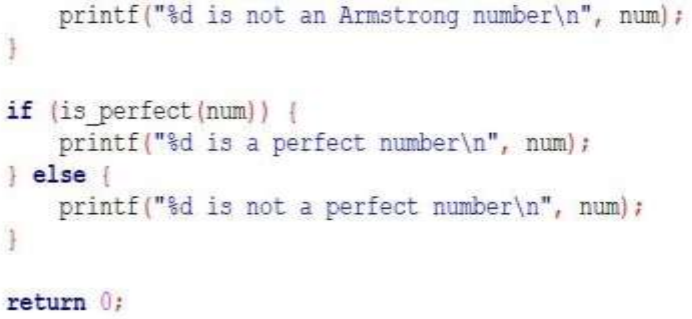
```
1   2   3   4   5   6
2   4   6   8   10  12
3   6   9   12  15  18
4   8   12  16  20  24
5   10  15  20  25  30
6   12  18  24  30  36
```

5: Write a program to check that the given number is prime, Armstrong or perfect using the concept of functions. <u>Solution</u>:

Program 1

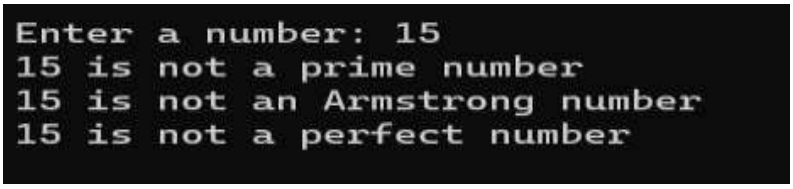


```
    while (num != 0) {
        int digit = num % 10;
        sum += pow(digit, num_digits);
        num /= 10;
    }

    return sum == original_num;
}

bool is_perfect(int num) {
    int sum = 0;
    for (int i = 1; i < num; i++) {
        if (num % i == 0) {
            sum += i;
        }
    }
    return sum == num;
}

int main() {
    int num;
    printf("Enter a number: ");
    scanf("%d", &num);

    if (is_prime(num)) {
        printf("%d is a prime number\n", num);
    } else {
```



```
        printf("%d is not an Armstrong number\n", num);
    }

    if (is_perfect(num)) {
        printf("%d is a perfect number\n", num);
    } else {
        printf("%d is not a perfect number\n", num);
    }

    return 0;
```

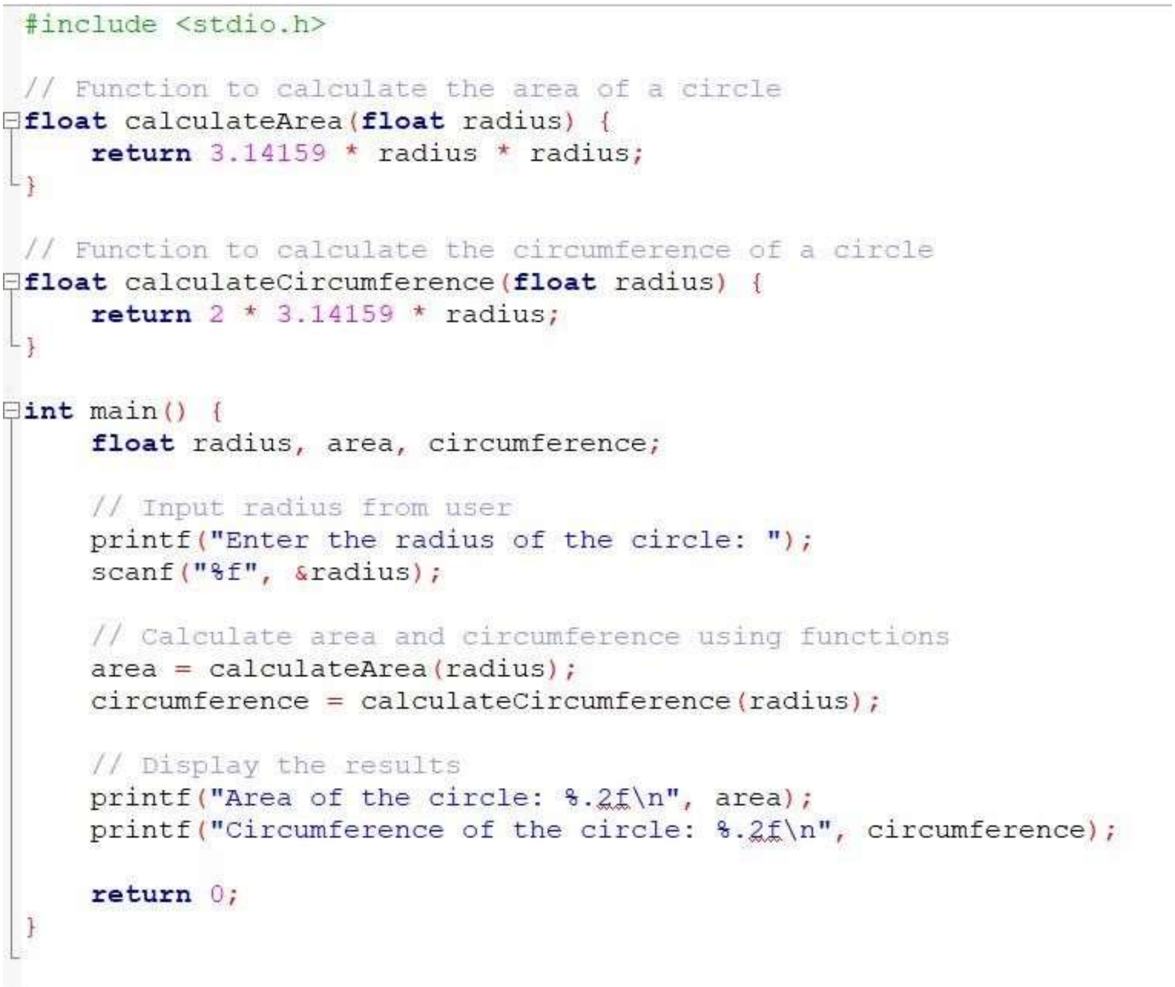
Output:



```
Enter a number: 15
15 is not a prime number
15 is not an Armstrong number
15 is not a perfect number
```

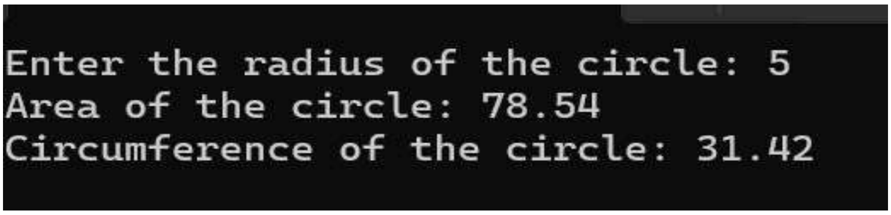
Program 1

6: Write a program to calculate the area and circumference of a circle using functions. <u>Solution</u>:



```c
#include <stdio.h>

// Function to calculate the area of a circle
float calculateArea(float radius) {
    return 3.14159 * radius * radius;
}

// Function to calculate the circumference of a circle
float calculateCircumference(float radius) {
    return 2 * 3.14159 * radius;
}

int main() {
    float radius, area, circumference;

    // Input radius from user
    printf("Enter the radius of the circle: ");
    scanf("%f", &radius);

    // Calculate area and circumference using functions
    area = calculateArea(radius);
    circumference = calculateCircumference(radius);

    // Display the results
    printf("Area of the circle: %.2f\n", area);
    printf("Circumference of the circle: %.2f\n", circumference);

    return 0;
}
```

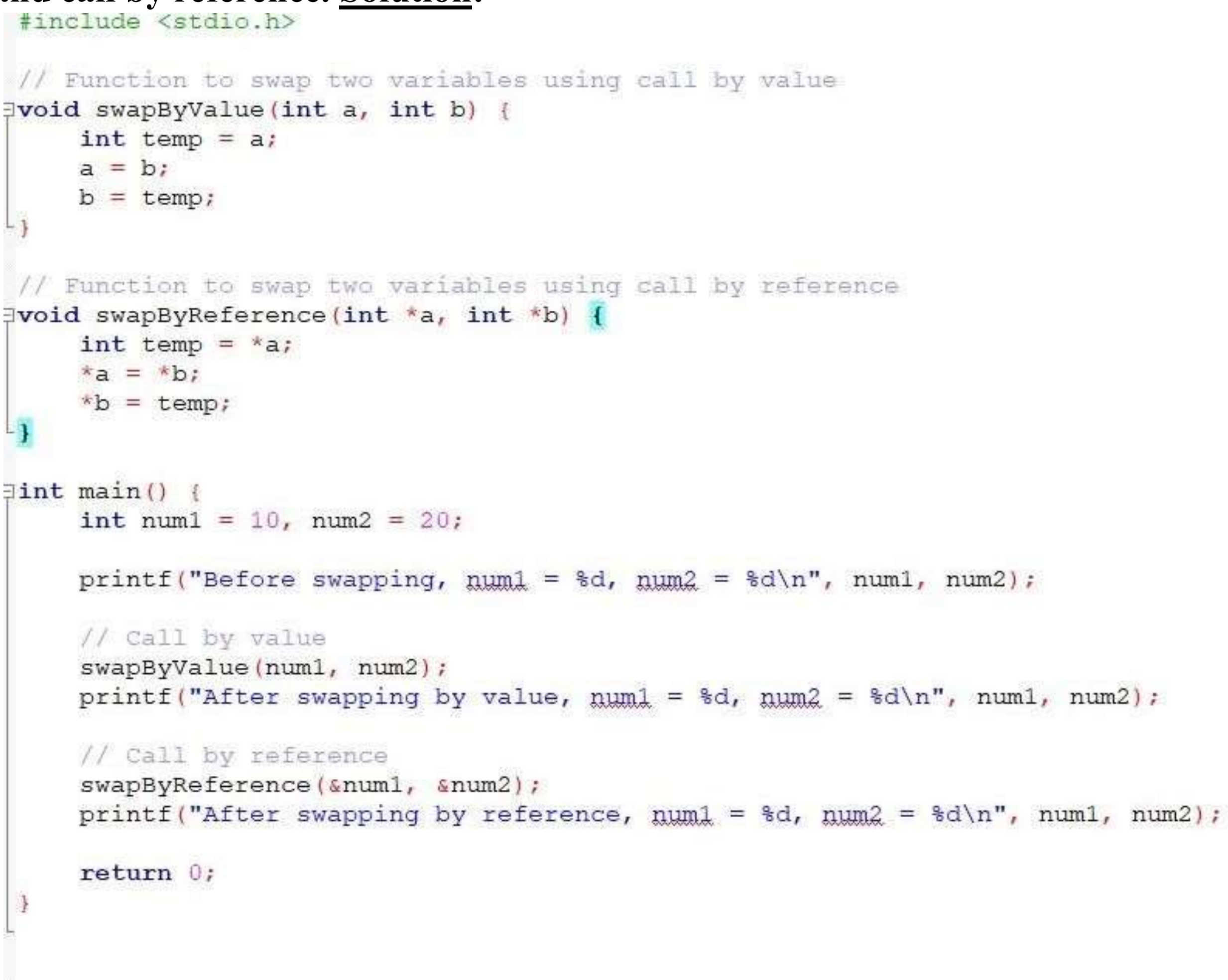
<u>Output</u>:



```
Enter the radius of the circle: 5
Area of the circle: 78.54
Circumference of the circle: 31.42
```

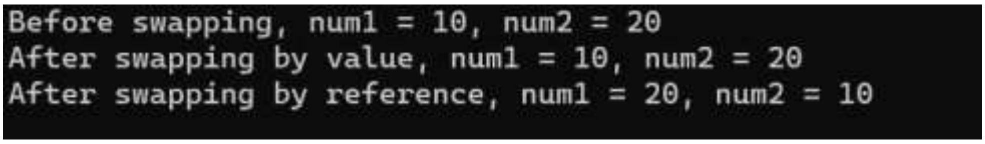
Program 1

7: Write a program to swap two variables using the concept of call by value and call by reference. Solution:



```
#include <stdio.h>

// Function to swap two variables using call by value
void swapByValue(int a, int b) {
    int temp = a;
    a = b;
    b = temp;
}

// Function to swap two variables using call by reference
void swapByReference(int *a, int *b) {
    int temp = *a;
    *a = *b;
    *b = temp;
}

int main() {
    int num1 = 10, num2 = 20;

    printf("Before swapping, num1 = %d, num2 = %d\n", num1, num2);

    // Call by value
    swapByValue(num1, num2);
    printf("After swapping by value, num1 = %d, num2 = %d\n", num1, num2);

    // Call by reference
    swapByReference(&num1, &num2);
    printf("After swapping by reference, num1 = %d, num2 = %d\n", num1, num2);

    return 0;
}
```

Output:

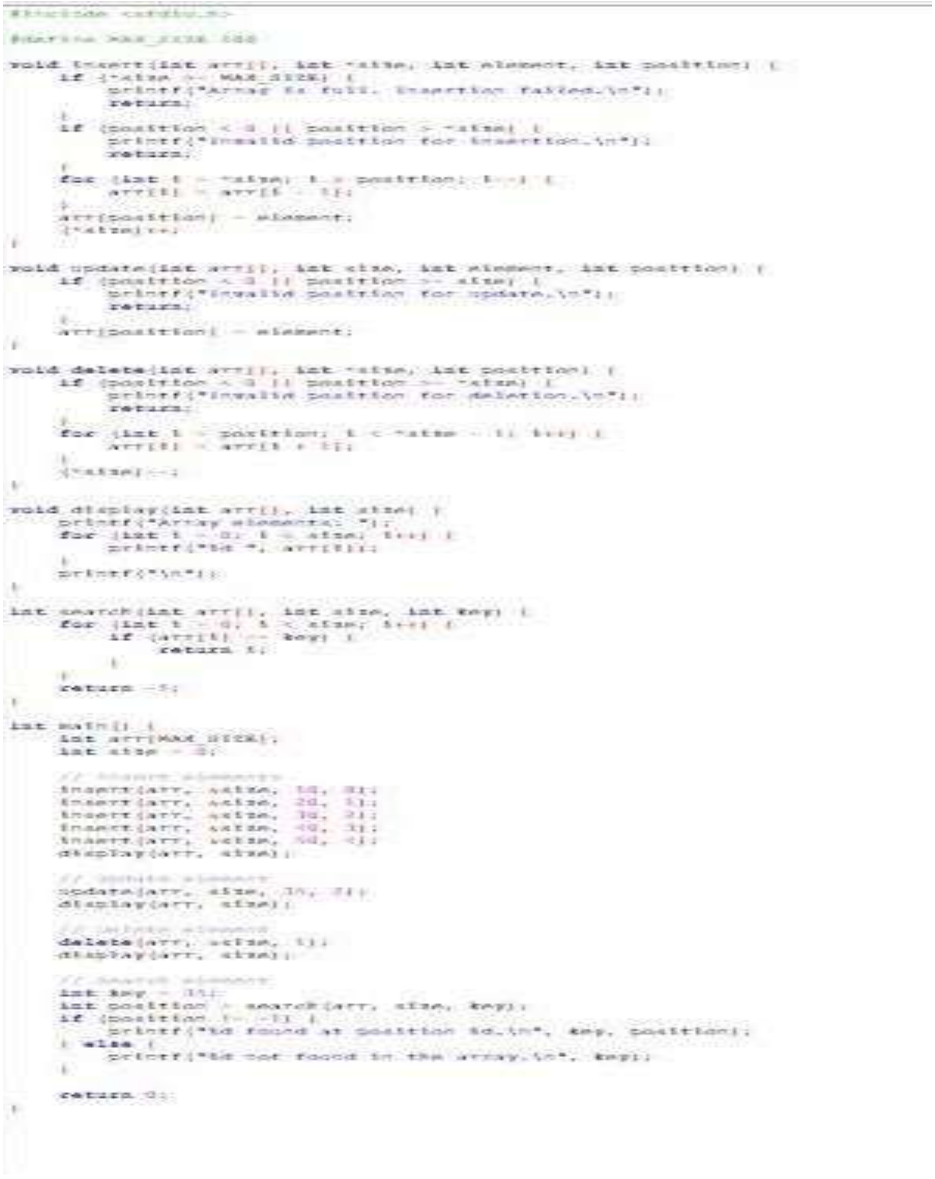


```
Before swapping, num1 = 10, num2 = 20
After swapping by value, num1 = 10, num2 = 20
After swapping by reference, num1 = 20, num2 = 10
```

Program 1

8: Write a program to perform the following operations on 1D-Array:

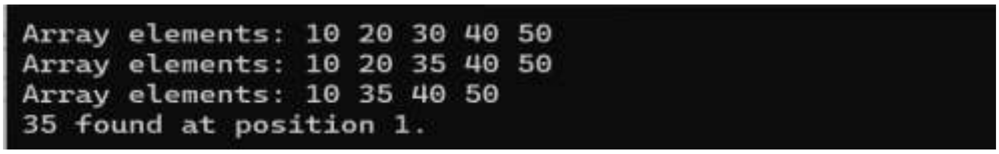
• Insert • Update • Delete • Display • Search Solution:



```c
#include <stdio.h>

#define MAX_SIZE 100

void insert(int arr[], int *size, int element, int position) {
    if (*size >= MAX_SIZE) {
        printf("Array is full. Insertion failed.\n");
        return;
    }
    if (position < 0 || position > *size) {
        printf("Invalid position for insertion.\n");
        return;
    }
    for (int i = *size; i > position; i--) {
        arr[i] = arr[i - 1];
    }
    arr[position] = element;
    (*size)++;
}

void update(int arr[], int size, int element, int position) {
    if (position < 0 || position >= size) {
        printf("Invalid position for update.\n");
        return;
    }
    arr[position] = element;
}

void delete(int arr[], int *size, int position) {
    if (position < 0 || position >= *size) {
        printf("Invalid position for deletion.\n");
        return;
    }
    for (int i = position; i < *size - 1; i++) {
        arr[i] = arr[i + 1];
    }
    (*size)--;
}

void display(int arr[], int size) {
    printf("Array elements: ");
    for (int i = 0; i < size; i++) {
        printf("%d ", arr[i]);
    }
    printf("\n");
}

int search(int arr[], int size, int key) {
    for (int i = 0; i < size; i++) {
        if (arr[i] == key) {
            return i;
        }
    }
    return -1;
}

int main() {
    int arr[MAX_SIZE];
    int size = 0;

    // Insert elements
    insert(arr, &size, 10, 0);
    insert(arr, &size, 20, 1);
    insert(arr, &size, 30, 2);
    insert(arr, &size, 40, 3);
    insert(arr, &size, 50, 4);
    display(arr, size);

    // Update element
    update(arr, size, 35, 2);
    display(arr, size);

    // Delete element
    delete(arr, &size, 1);
    display(arr, size);

    // Search element
    int key = 35;
    int position = search(arr, size, key);
    if (position != -1) {
        printf("%d found at position %d.\n", key, position);
    } else {
        printf("%d not found in the array.\n", key);
    }

    return 0;
}
```

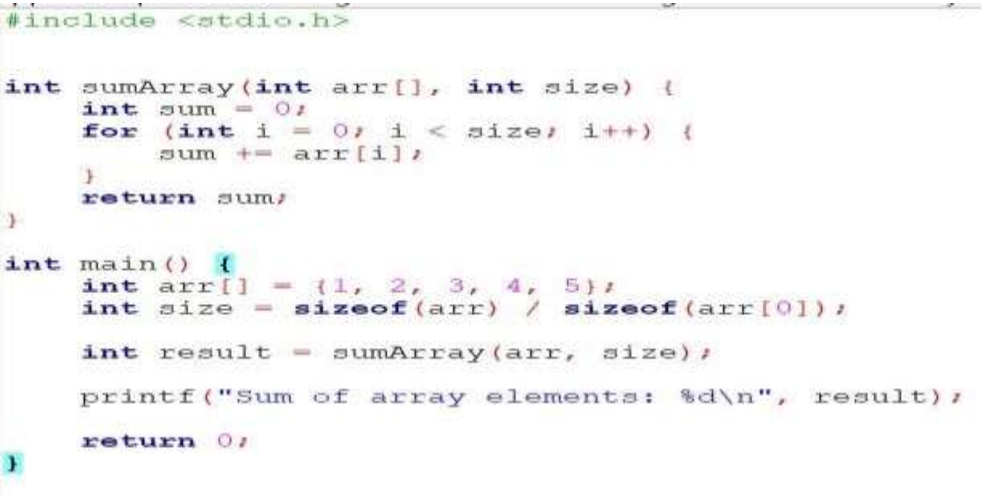
Output:

Program 1



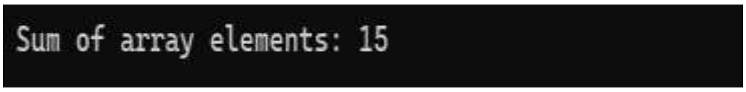
```
Array elements: 10 20 30 40 50
Array elements: 10 20 35 40 50
Array elements: 10 35 40 50
35 found at position 1.
```

9: Write a program to calculate the sum of array elements by passing it to a function Solution:



```
#include <stdio.h>

int sumArray(int arr[], int size) {
    int sum = 0;
    for (int i = 0; i < size; i++) {
        sum += arr[i];
    }
    return sum;
}

int main() {
    int arr[] = {1, 2, 3, 4, 5};
    int size = sizeof(arr) / sizeof(arr[0]);

    int result = sumArray(arr, size);

    printf("Sum of array elements: %d\n", result);

    return 0;
}
```

Output:

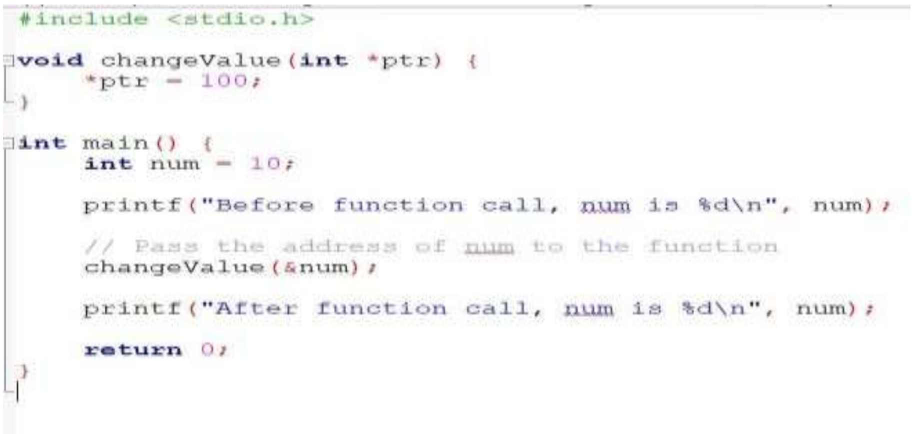


```
Sum of array elements: 15
```

Program 20: Write a program to show the use of passing pointer as arguments to the functions.

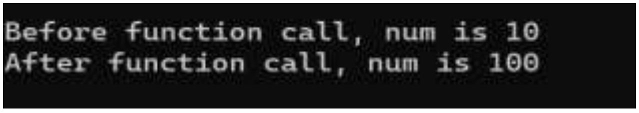
Solution:

Program 1



```
#include <stdio.h>

void changeValue(int *ptr) {
    *ptr = 100;
}

int main() {
    int num = 10;

    printf("Before function call, num is %d\n", num);

    // Pass the address of num to the function
    changeValue(&num);

    printf("After function call, num is %d\n", num);

    return 0;
}
```

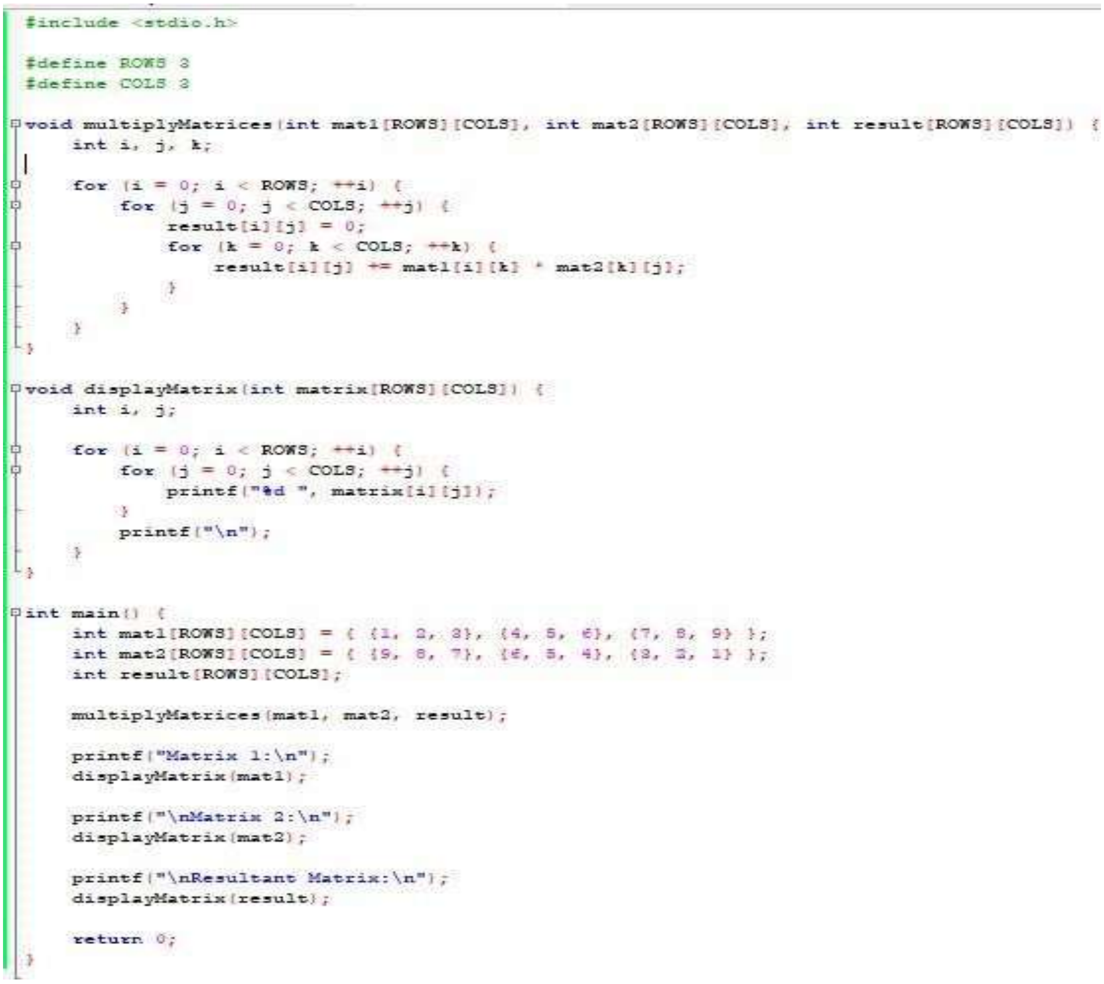
Output:



```
Before function call, num is 10
After function call, num is 100
```

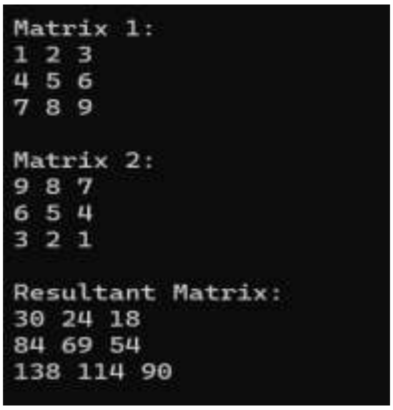
Program 21: Write a program matrix multiplication using the concept of 2D array

Solution:



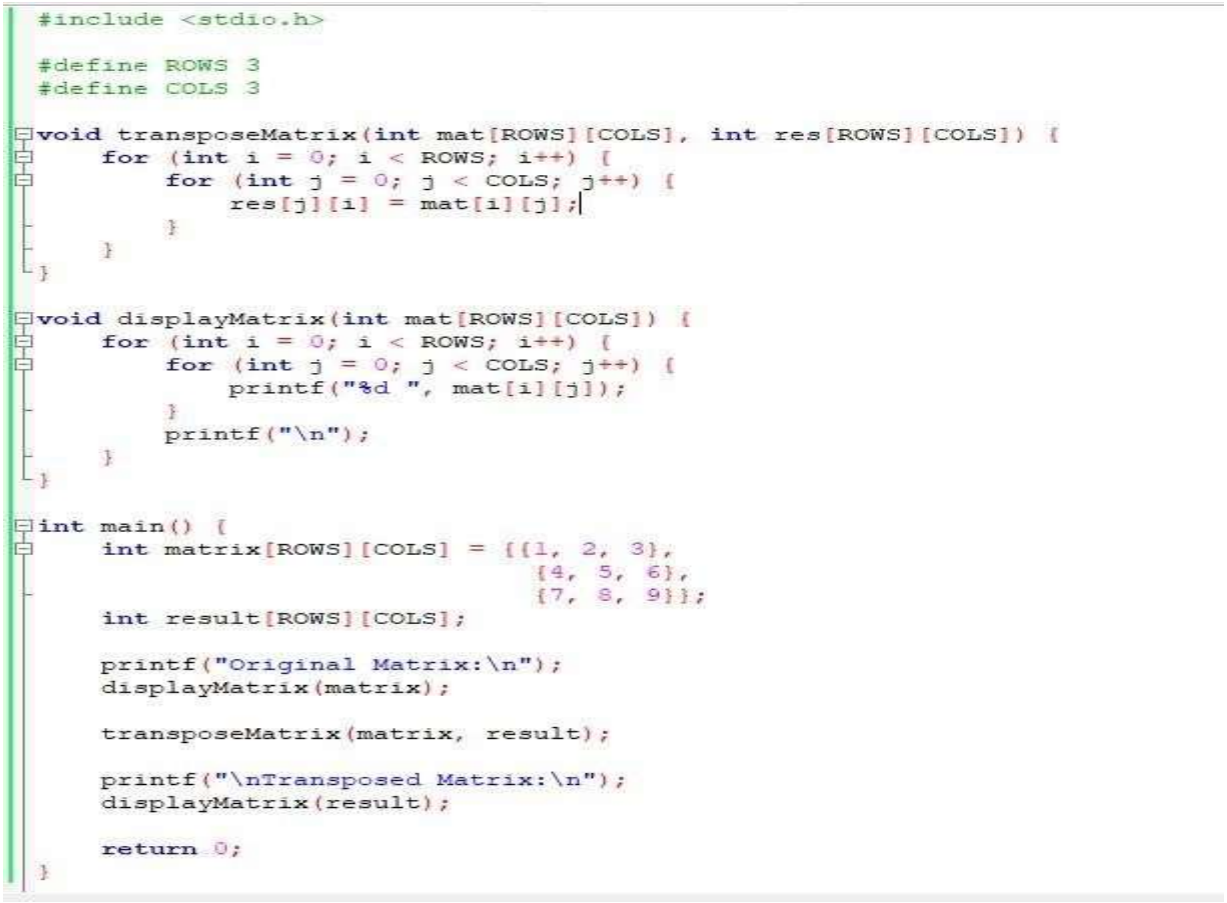
```
#include <stdio.h>

#define ROWS 3
#define COLS 3

void multiplyMatrices(int mat1[ROWS][COLS], int mat2[ROWS][COLS], int result[ROWS][COLS]) {
    int i, j, k;

    for (i = 0; i < ROWS; ++i) {
        for (j = 0; j < COLS; ++j) {
            result[i][j] = 0;
            for (k = 0; k < COLS; ++k) {
                result[i][j] += mat1[i][k] * mat2[k][j];
            }
        }
    }
}

void displayMatrix(int matrix[ROWS][COLS]) {
    int i, j;

    for (i = 0; i < ROWS; ++i) {
        for (j = 0; j < COLS; ++j) {
            printf("%d ", matrix[i][j]);
        }
        printf("\n");
    }
}

int main() {
    int mat1[ROWS][COLS] = { {1, 2, 3}, {4, 5, 6}, {7, 8, 9} };
    int mat2[ROWS][COLS] = { {9, 8, 7}, {6, 5, 4}, {3, 2, 1} };
    int result[ROWS][COLS];

    multiplyMatrices(mat1, mat2, result);

    printf("Matrix 1:\n");
    displayMatrix(mat1);

    printf("\nMatrix 2:\n");
    displayMatrix(mat2);

    printf("\nResultant Matrix:\n");
    displayMatrix(result);

    return 0;
}
```

Output:



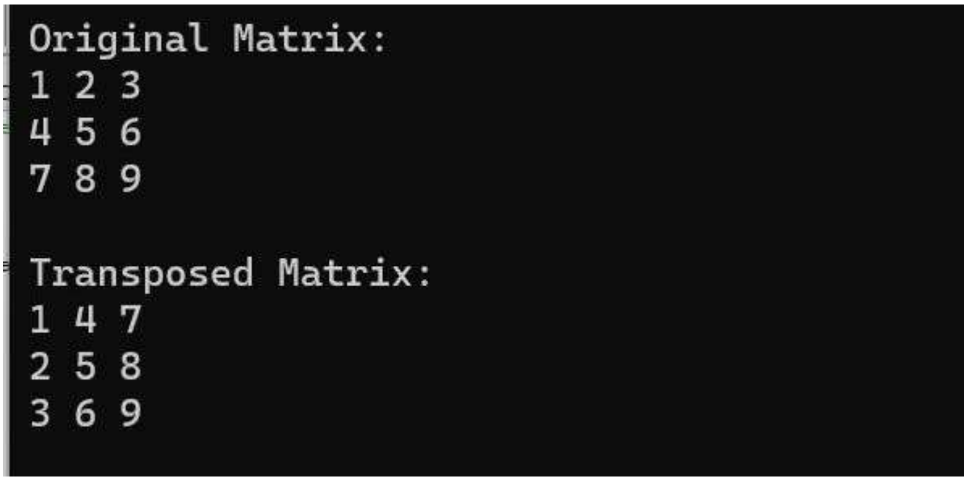
```
Matrix 1:
1 2 3
4 5 6
7 8 9

Matrix 2:
9 8 7
6 5 4
3 2 1

Resultant Matrix:
30 24 18
84 69 54
138 114 90
```

Program 22: Write a program to transpose a given matrix. **Solution:**



```
#include <stdio.h>

#define ROWS 3
#define COLS 3

void transposeMatrix(int mat[ROWS][COLS], int res[ROWS][COLS]) {
    for (int i = 0; i < ROWS; i++) {
        for (int j = 0; j < COLS; j++) {
            res[j][i] = mat[i][j];
        }
    }
}

void displayMatrix(int mat[ROWS][COLS]) {
    for (int i = 0; i < ROWS; i++) {
        for (int j = 0; j < COLS; j++) {
            printf("%d ", mat[i][j]);
        }
        printf("\n");
    }
}

int main() {
    int matrix[ROWS][COLS] = {{1, 2, 3},
                              {4, 5, 6},
                              {7, 8, 9}};
    int result[ROWS][COLS];

    printf("Original Matrix:\n");
    displayMatrix(matrix);

    transposeMatrix(matrix, result);

    printf("\nTransposed Matrix:\n");
    displayMatrix(result);

    return 0;
}
```

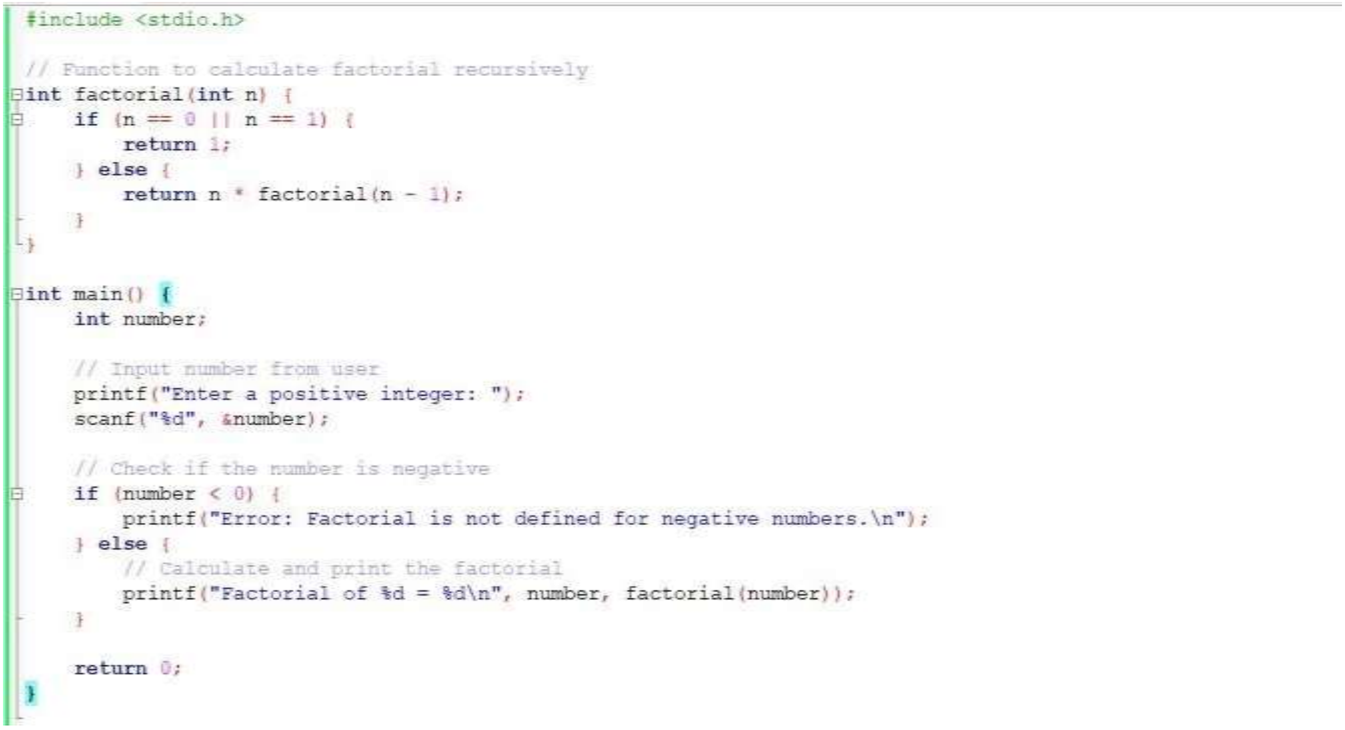
Output:



```
Original Matrix:
1 2 3
4 5 6
7 8 9

Transposed Matrix:
1 4 7
2 5 8
3 6 9
```

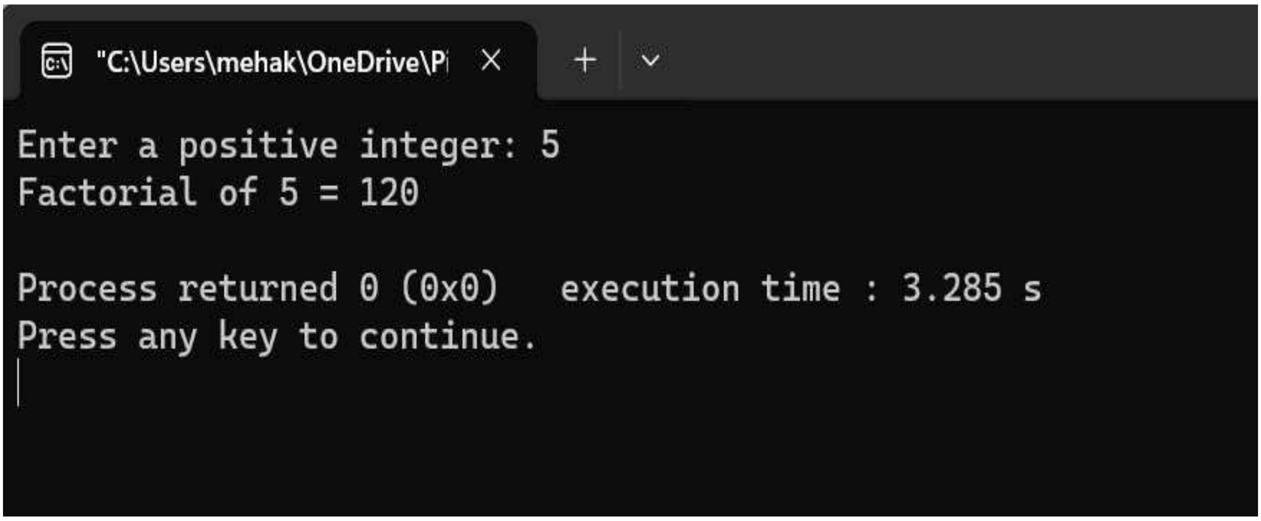
Program 23: Write a program to find the factorial of a number by using the concept of recursion.

<u>Solution</u>:



```
#include <stdio.h>

// Function to calculate factorial recursively
int factorial(int n) {
    if (n == 0 || n == 1) {
        return 1;
    } else {
        return n * factorial(n - 1);
    }
}

int main() {
    int number;

    // Input number from user
    printf("Enter a positive integer: ");
    scanf("%d", &number);

    // Check if the number is negative
    if (number < 0) {
        printf("Error: Factorial is not defined for negative numbers.\n");
    } else {
        // Calculate and print the factorial
        printf("Factorial of %d = %d\n", number, factorial(number));
    }

    return 0;
}
```

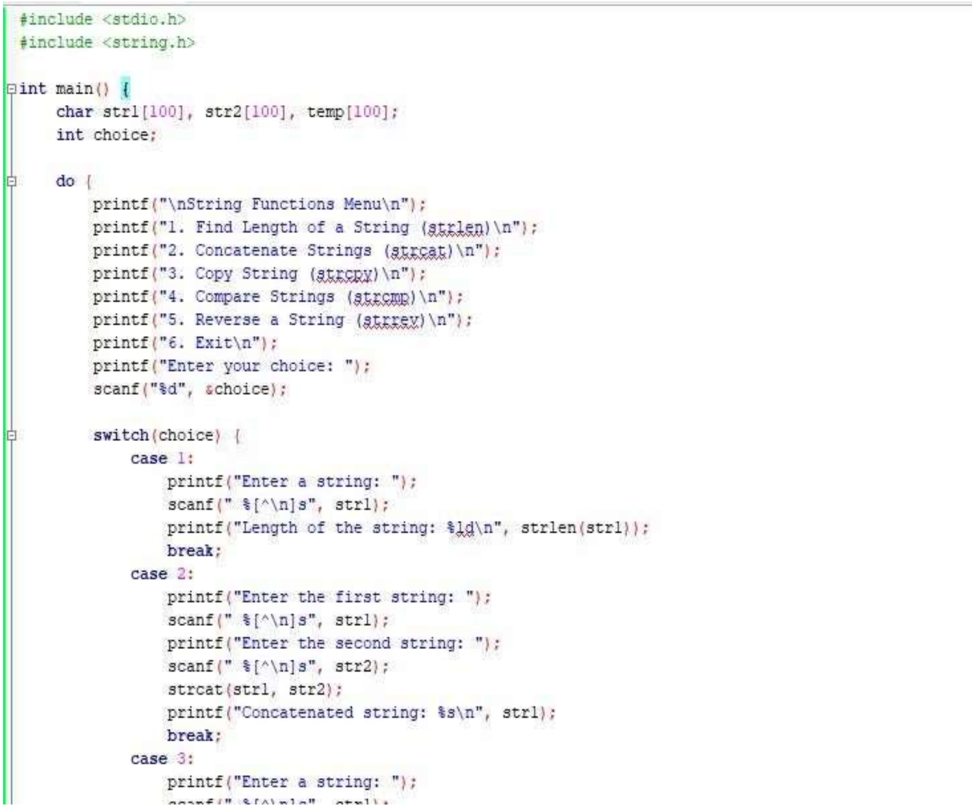
Output:



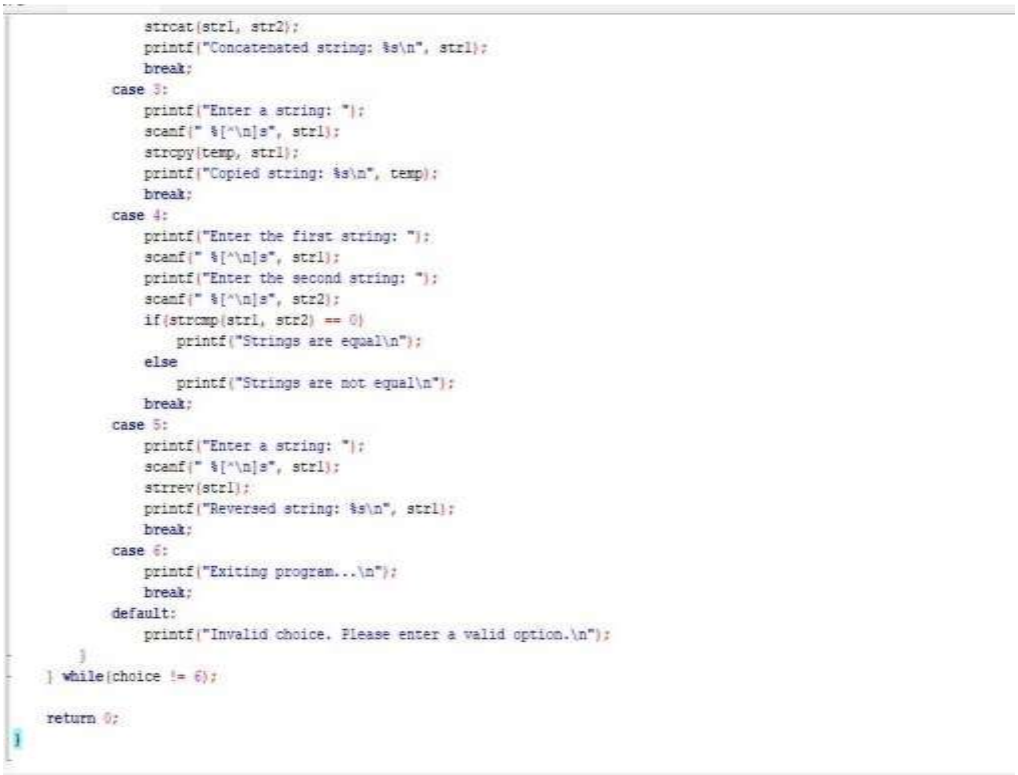
```
Enter a positive integer: 5
Factorial of 5 = 120

Process returned 0 (0x0)   execution time : 3.285 s
Press any key to continue.
```

Program 24: Write a menu driven C program to show the use of in-built string functions like strlen, strcat, strcpy, strcmp, strrev etc.

Solution:

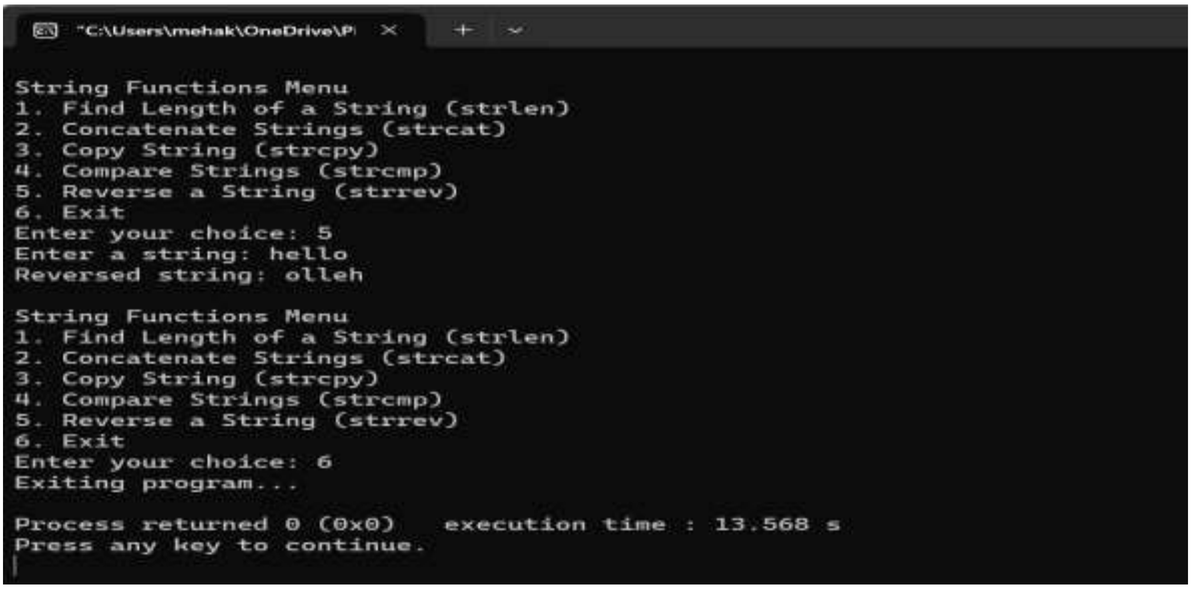


```
#include <stdio.h>
#include <string.h>

int main() {
    char str1[100], str2[100], temp[100];
    int choice;

    do {
        printf("\nString Functions Menu\n");
        printf("1. Find Length of a String (strlen)\n");
        printf("2. Concatenate Strings (strcat)\n");
        printf("3. Copy String (strcpy)\n");
        printf("4. Compare Strings (strcmp)\n");
        printf("5. Reverse a String (strrev)\n");
        printf("6. Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);

        switch(choice) {
            case 1:
                printf("Enter a string: ");
                scanf(" %[^\n]s", str1);
                printf("Length of the string: %ld\n", strlen(str1));
                break;
            case 2:
                printf("Enter the first string: ");
                scanf(" %[^\n]s", str1);
                printf("Enter the second string: ");
                scanf(" %[^\n]s", str2);
                strcat(str1, str2);
                printf("Concatenated string: %s\n", str1);
                break;
            case 3:
                printf("Enter a string: ");
```



```
                strcat(str1, str2);
                printf("Concatenated string: %s\n", str1);
                break;
            case 3:
                printf("Enter a string: ");
                scanf(" %[^\n]s", str1);
                strcpy(temp, str1);
                printf("Copied string: %s\n", temp);
                break;
            case 4:
                printf("Enter the first string: ");
                scanf(" %[^\n]s", str1);
                printf("Enter the second string: ");
                scanf(" %[^\n]s", str2);
                if(strcmp(str1, str2) == 0)
                    printf("Strings are equal\n");
                else
                    printf("Strings are not equal\n");
                break;
            case 5:
                printf("Enter a string: ");
                scanf(" %[^\n]s", str1);
                strrev(str1);
                printf("Reversed string: %s\n", str1);
                break;
            case 6:
                printf("Exiting program...\n");
                break;
            default:
                printf("Invalid choice. Please enter a valid option.\n");
        }
    } while(choice != 6);

    return 0;
}
```

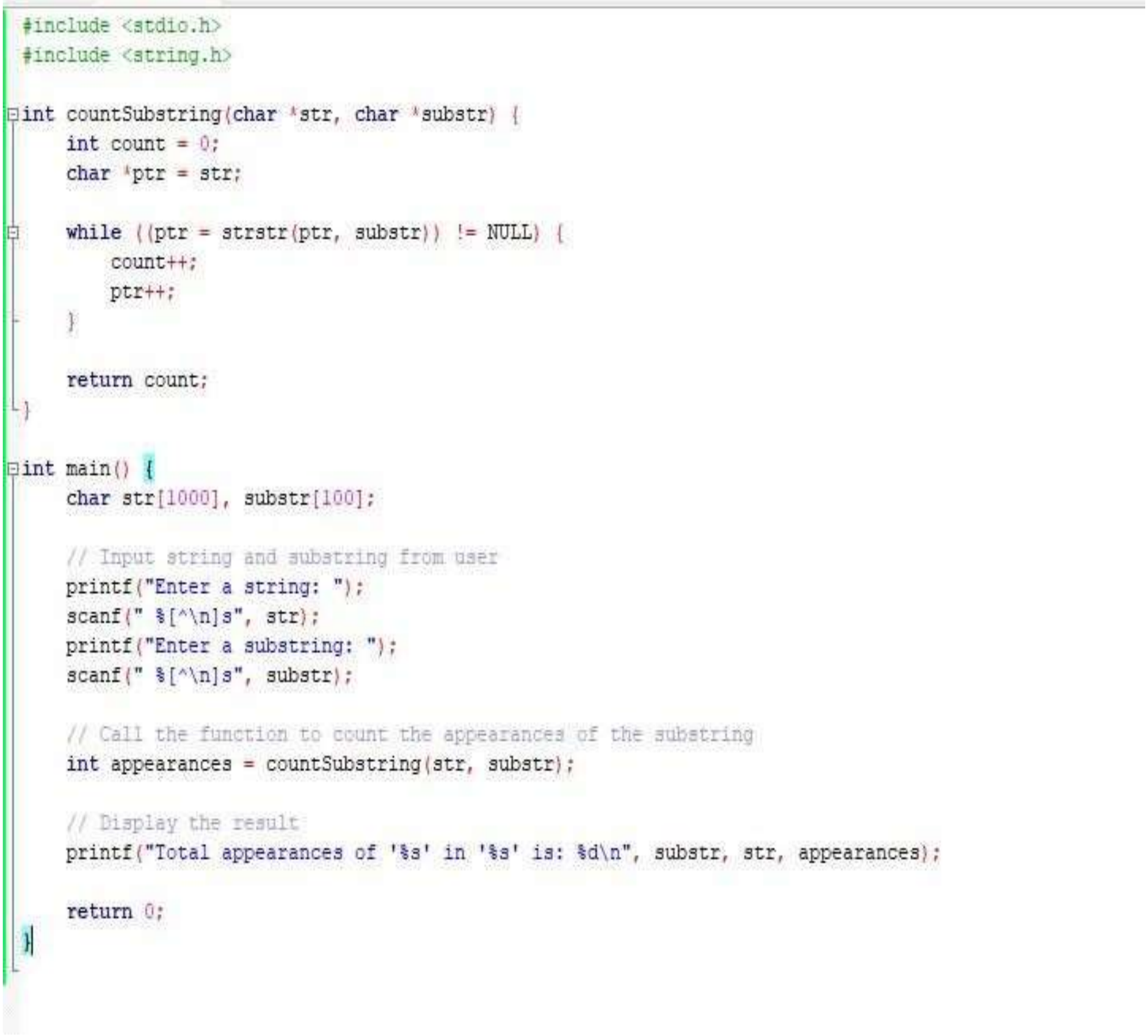
Output:



```
String Functions Menu
1. Find Length of a String (strlen)
2. Concatenate Strings (strcat)
3. Copy String (strcpy)
4. Compare Strings (strcmp)
5. Reverse a String (strrev)
6. Exit
Enter your choice: 5
Enter a string: hello
Reversed string: olleh

String Functions Menu
1. Find Length of a String (strlen)
2. Concatenate Strings (strcat)
3. Copy String (strcpy)
4. Compare Strings (strcmp)
5. Reverse a String (strrev)
6. Exit
Enter your choice: 6
Exiting program...

Process returned 0 (0x0)   execution time : 13.568 s
Press any key to continue.
```

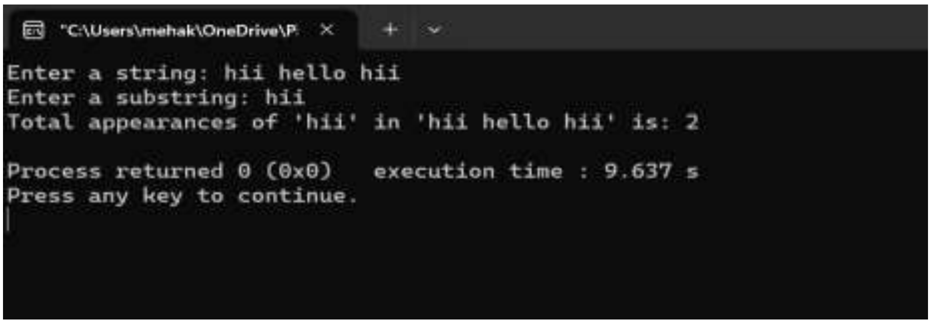
Program 25: Write a Program in C to display the total number of appearances of a substring provided as input by the user in a given string.

Solution:



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

int countSubstring(char *str, char *substr) {
    int count = 0;
    char *ptr = str;

    while ((ptr = strstr(ptr, substr)) != NULL) {
        count++;
        ptr++;
    }

    return count;
}

int main() {
    char str[1000], substr[100];

    // Input string and substring from user
    printf("Enter a string: ");
    scanf(" %[^\n]s", str);
    printf("Enter a substring: ");
    scanf(" %[^\n]s", substr);

    // Call the function to count the appearances of the substring
    int appearances = countSubstring(str, substr);

    // Display the result
    printf("Total appearances of '%s' in '%s' is: %d\n", substr, str, appearances);

    return 0;
}
```

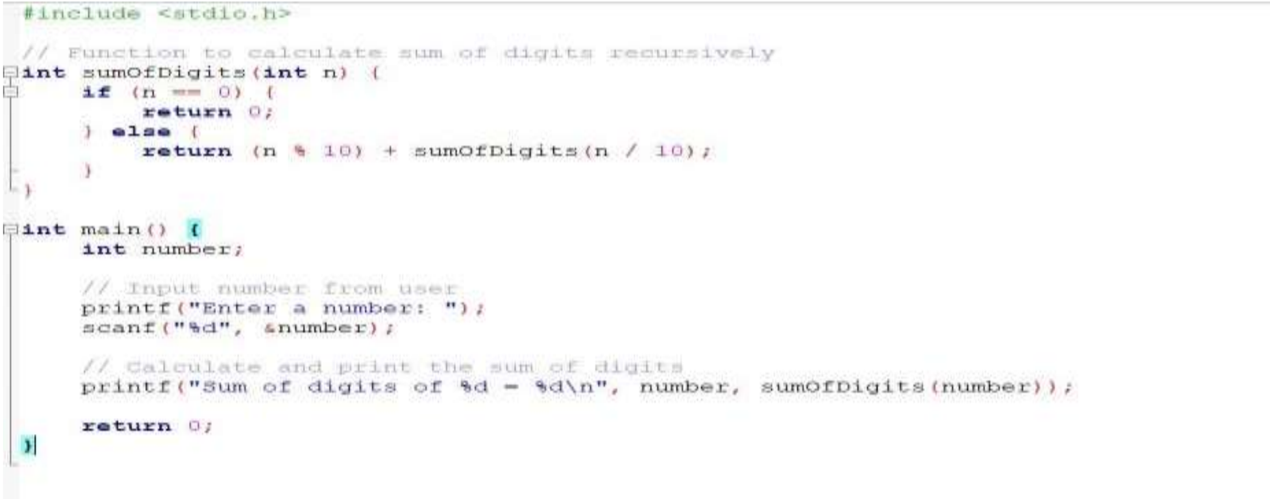
<u>Output</u>:



```
Enter a string: hii hello hii
Enter a substring: hii
Total appearances of 'hii' in 'hii hello hii' is: 2

Process returned 0 (0x0)   execution time : 9.637 s
Press any key to continue.
```

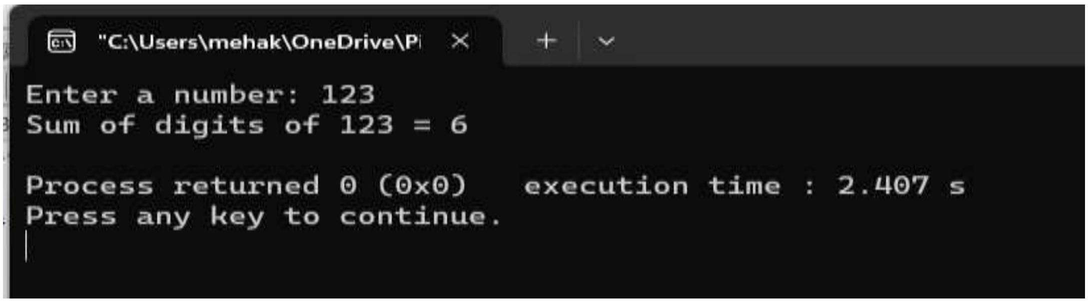
Program 26: Write a program to display the sum of the digits of a number by using the concept of recursion.

Solution:



```
#include <stdio.h>

// Function to calculate sum of digits recursively
int sumOfDigits(int n) {
    if (n == 0) {
        return 0;
    } else {
        return (n % 10) + sumOfDigits(n / 10);
    }
}

int main() {
    int number;

    // Input number from user
    printf("Enter a number: ");
    scanf("%d", &number);

    // Calculate and print the sum of digits
    printf("Sum of digits of %d = %d\n", number, sumOfDigits(number));

    return 0;
}
```

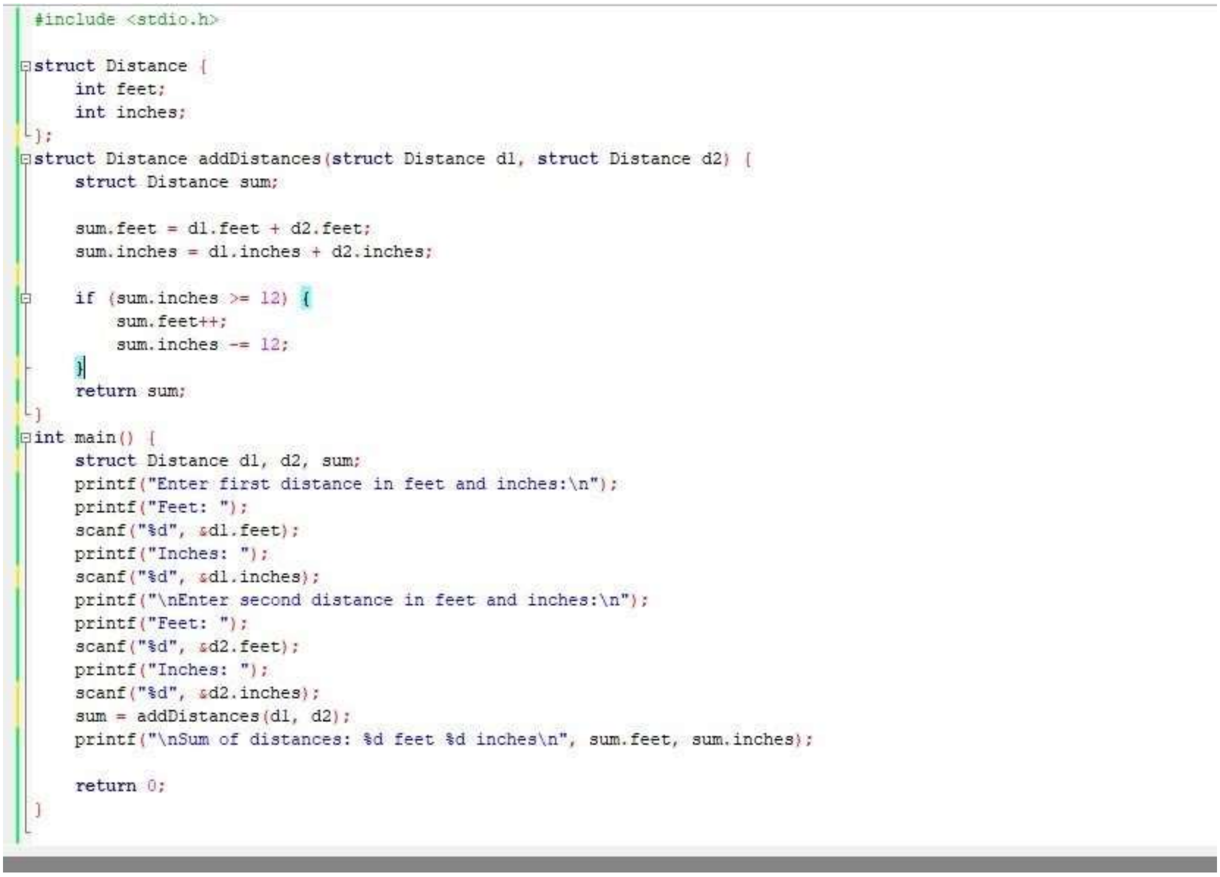
Output:



```
Enter a number: 123
Sum of digits of 123 = 6

Process returned 0 (0x0)   execution time : 2.407 s
Press any key to continue.
```

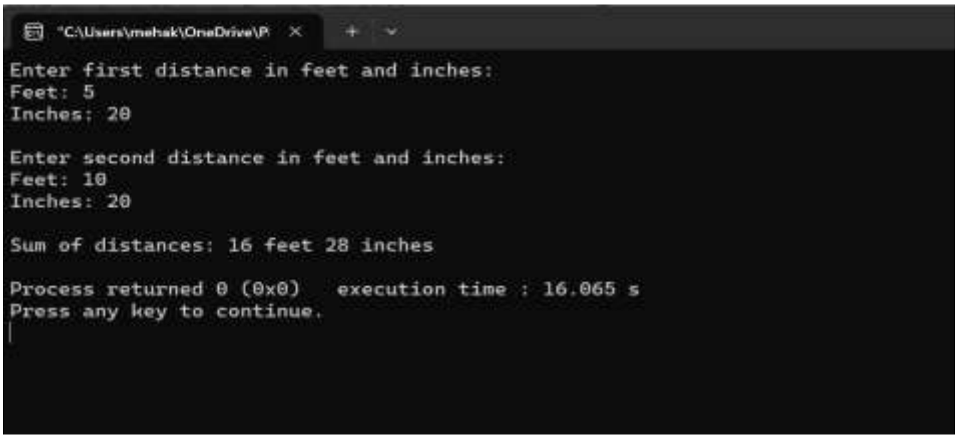
Program 27: Write a C program to add two distances in inch & feet using the concept of structures.

Solution:



```
#include <stdio.h>

struct Distance {
    int feet;
    int inches;
};
struct Distance addDistances(struct Distance d1, struct Distance d2) {
    struct Distance sum;

    sum.feet = d1.feet + d2.feet;
    sum.inches = d1.inches + d2.inches;

    if (sum.inches >= 12) {
        sum.feet++;
        sum.inches -= 12;
    }
    return sum;
}
int main() {
    struct Distance d1, d2, sum;
    printf("Enter first distance in feet and inches:\n");
    printf("Feet: ");
    scanf("%d", &d1.feet);
    printf("Inches: ");
    scanf("%d", &d1.inches);
    printf("\nEnter second distance in feet and inches:\n");
    printf("Feet: ");
    scanf("%d", &d2.feet);
    printf("Inches: ");
    scanf("%d", &d2.inches);
    sum = addDistances(d1, d2);
    printf("\nSum of distances: %d feet %d inches\n", sum.feet, sum.inches);

    return 0;
}
```

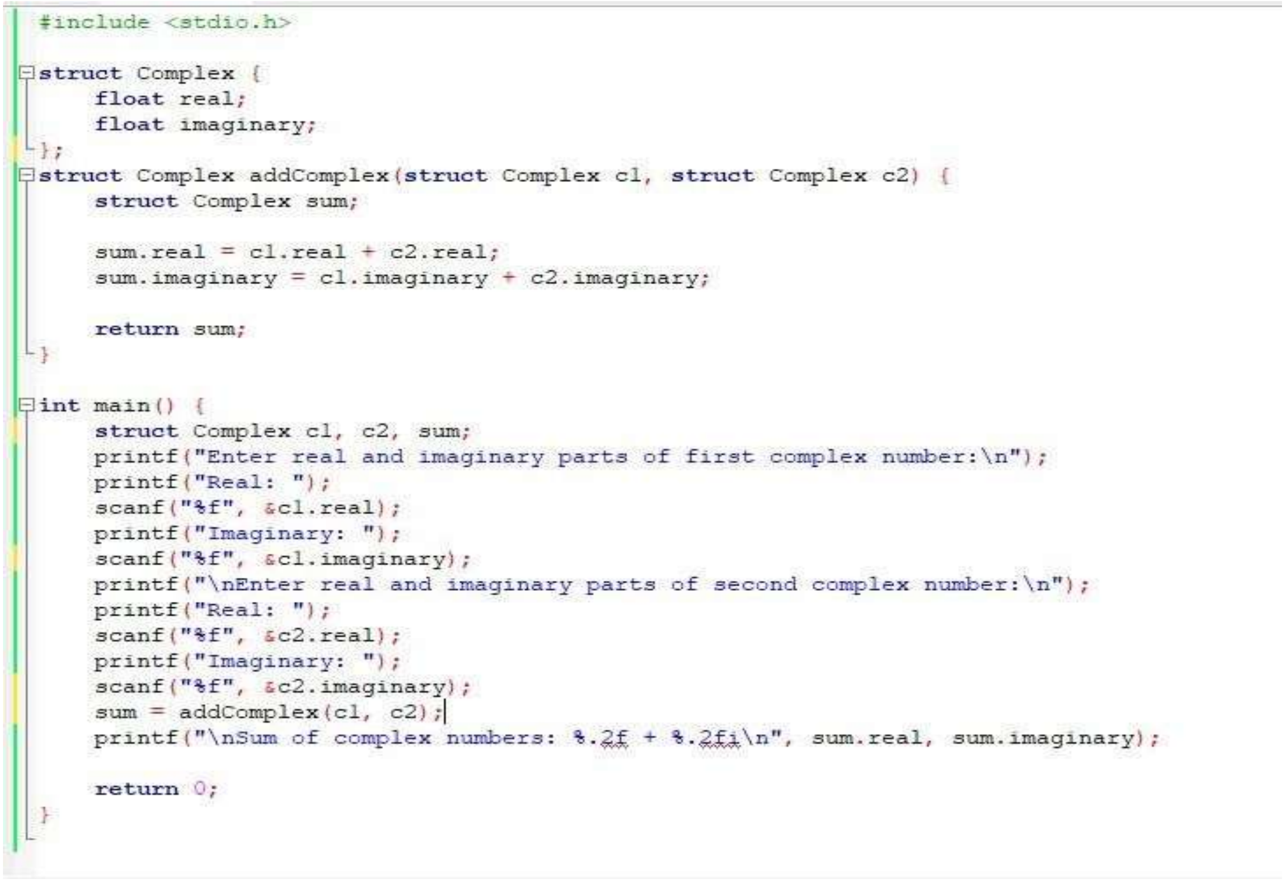
Output:



```
Enter first distance in feet and inches:
Feet: 5
Inches: 20

Enter second distance in feet and inches:
Feet: 10
Inches: 20

Sum of distances: 16 feet 28 inches

Process returned 0 (0x0)   execution time : 16.065 s
Press any key to continue.
```

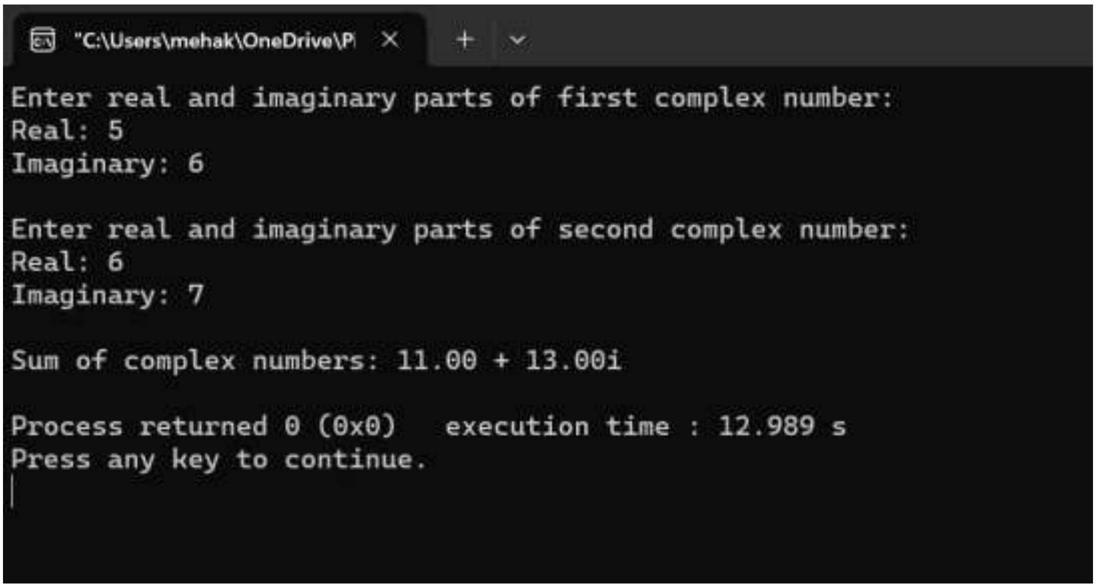
Program 28: Write a C program to add two complex numbers using the concept of structures in C.

Solution:



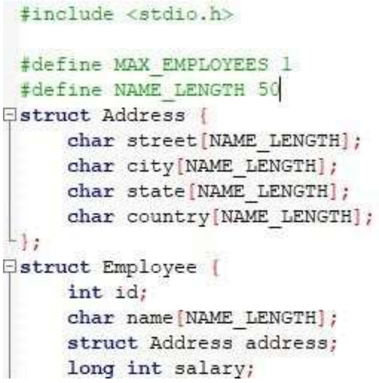
```c
#include <stdio.h>

struct Complex {
    float real;
    float imaginary;
};
struct Complex addComplex(struct Complex c1, struct Complex c2) {
    struct Complex sum;

    sum.real = c1.real + c2.real;
    sum.imaginary = c1.imaginary + c2.imaginary;

    return sum;
}

int main() {
    struct Complex c1, c2, sum;
    printf("Enter real and imaginary parts of first complex number:\n");
    printf("Real: ");
    scanf("%f", &c1.real);
    printf("Imaginary: ");
    scanf("%f", &c1.imaginary);
    printf("\nEnter real and imaginary parts of second complex number:\n");
    printf("Real: ");
    scanf("%f", &c2.real);
    printf("Imaginary: ");
    scanf("%f", &c2.imaginary);
    sum = addComplex(c1, c2);
    printf("\nSum of complex numbers: %.2f + %.2fi\n", sum.real, sum.imaginary);

    return 0;
}
```

Output:

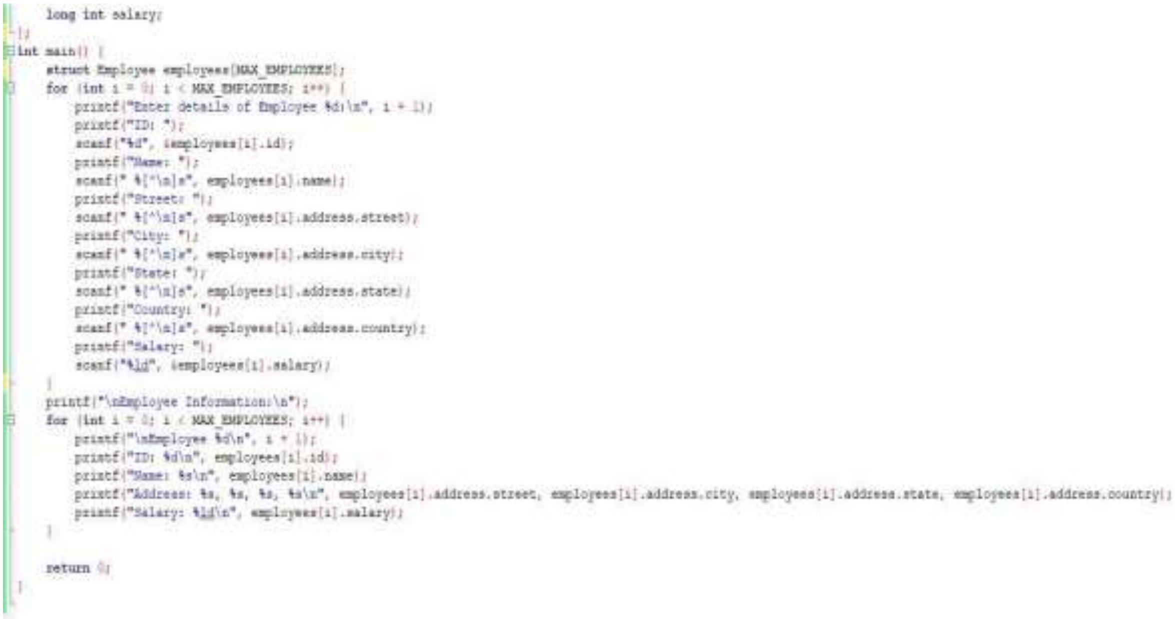


```
Enter real and imaginary parts of first complex number:
Real: 5
Imaginary: 6

Enter real and imaginary parts of second complex number:
Real: 6
Imaginary: 7

Sum of complex numbers: 11.00 + 13.00i

Process returned 0 (0x0)   execution time : 12.989 s
Press any key to continue.
```

Program 29: Write a program in C to store the information of five employees using both concepts i.e. array of structure and array within structure. Solution:

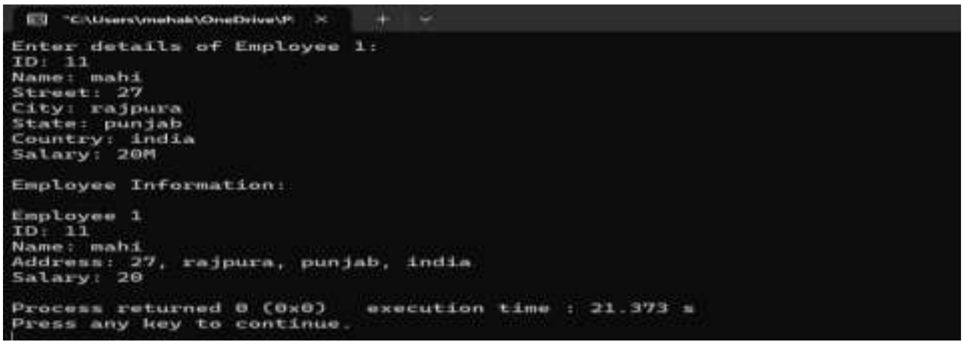


```
#include <stdio.h>

#define MAX_EMPLOYEES 1
#define NAME_LENGTH 50
struct Address {
    char street[NAME_LENGTH];
    char city[NAME_LENGTH];
    char state[NAME_LENGTH];
    char country[NAME_LENGTH];
};
struct Employee {
    int id;
    char name[NAME_LENGTH];
    struct Address address;
    long int salary;
```



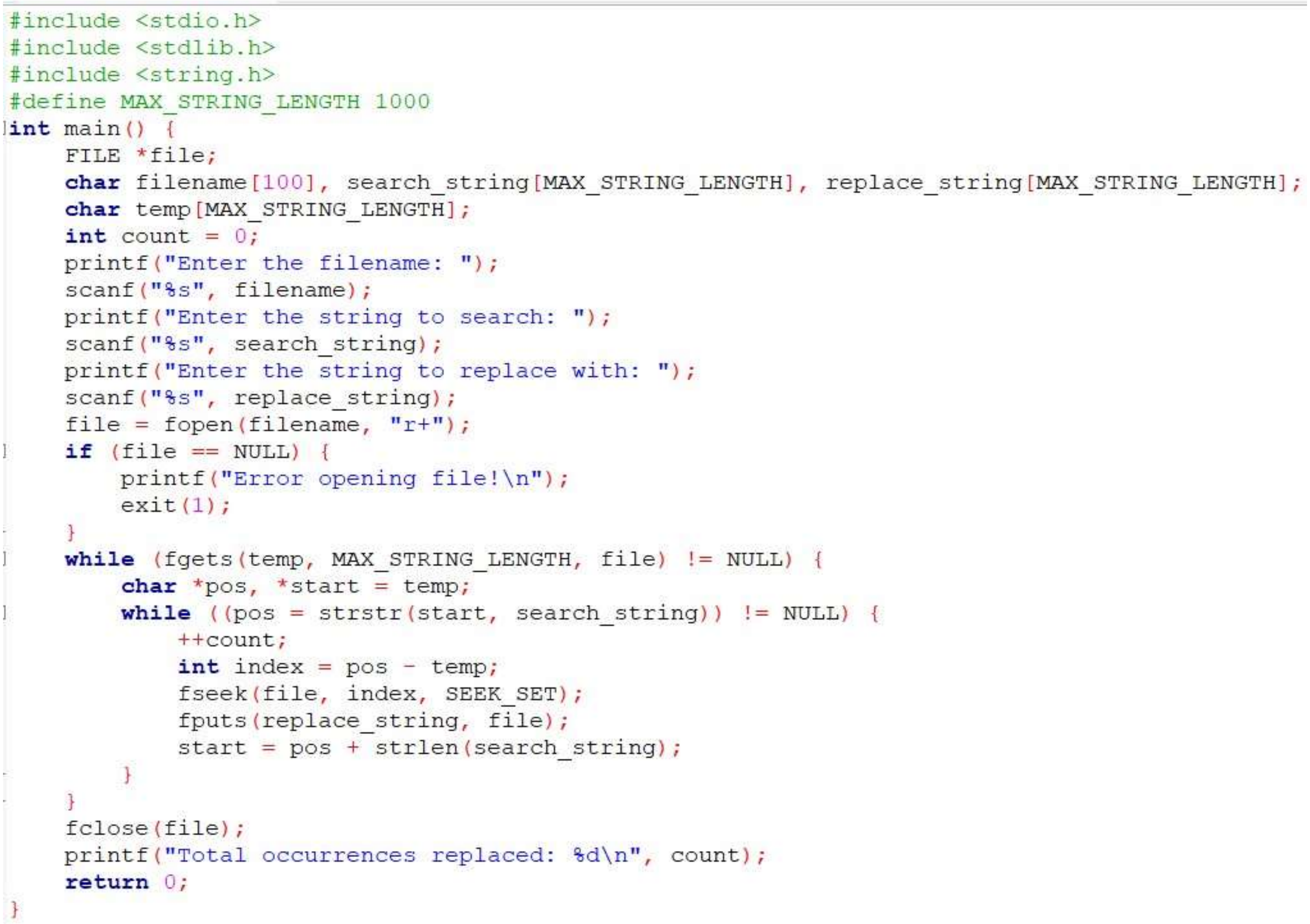
```
    long int salary;
};
int main() {
    struct Employee employees[MAX_EMPLOYEES];
    for (int i = 0; i < MAX_EMPLOYEES; i++) {
        printf("Enter details of Employee %d:\n", i + 1);
        printf("ID: ");
        scanf("%d", &employees[i].id);
        printf("Name: ");
        scanf(" %[^\n]s", employees[i].name);
        printf("Street: ");
        scanf(" %[^\n]s", employees[i].address.street);
        printf("City: ");
        scanf(" %[^\n]s", employees[i].address.city);
        printf("State: ");
        scanf(" %[^\n]s", employees[i].address.state);
        printf("Country: ");
        scanf(" %[^\n]s", employees[i].address.country);
        printf("Salary: ");
        scanf("%ld", &employees[i].salary);
    }
    printf("\nEmployee Information:\n");
    for (int i = 0; i < MAX_EMPLOYEES; i++) {
        printf("\nEmployee %d\n", i + 1);
        printf("ID: %d\n", employees[i].id);
        printf("Name: %s\n", employees[i].name);
        printf("Address: %s, %s, %s, %s\n", employees[i].address.street, employees[i].address.city, employees[i].address.state, employees[i].address.country);
        printf("Salary: %ld\n", employees[i].salary);
    }

    return 0;
}
```

<u>Output</u>:



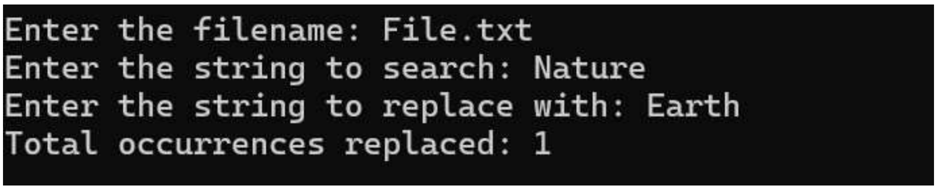
```
Enter details of Employee 1:
ID: 11
Name: mahi
Street: 27
City: rajpura
State: punjab
Country: india
Salary: 20M

Employee Information:

Employee 1
ID: 11
Name: mahi
Address: 27, rajpura, punjab, india
Salary: 20

Process returned 0 (0x0)   execution time : 21.373 s
Press any key to continue.
```

Program 30: Write a Program in C to find and replace a specific string in a file and also display the total number of appearances of that string. <u>Solution</u>:



```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#define MAX_STRING_LENGTH 1000
int main() {
    FILE *file;
    char filename[100], search_string[MAX_STRING_LENGTH], replace_string[MAX_STRING_LENGTH];
    char temp[MAX_STRING_LENGTH];
    int count = 0;
    printf("Enter the filename: ");
    scanf("%s", filename);
    printf("Enter the string to search: ");
    scanf("%s", search_string);
    printf("Enter the string to replace with: ");
    scanf("%s", replace_string);
    file = fopen(filename, "r+");
    if (file == NULL) {
        printf("Error opening file!\n");
        exit(1);
    }
    while (fgets(temp, MAX_STRING_LENGTH, file) != NULL) {
        char *pos, *start = temp;
        while ((pos = strstr(start, search_string)) != NULL) {
            ++count;
            int index = pos - temp;
            fseek(file, index, SEEK_SET);
            fputs(replace_string, file);
            start = pos + strlen(search_string);
        }
    }
    fclose(file);
    printf("Total occurrences replaced: %d\n", count);
    return 0;
}
```

<u>Output</u>:



```
Enter the filename: File.txt
Enter the string to search: Nature
Enter the string to replace with: Earth
Total occurrences replaced: 1
```