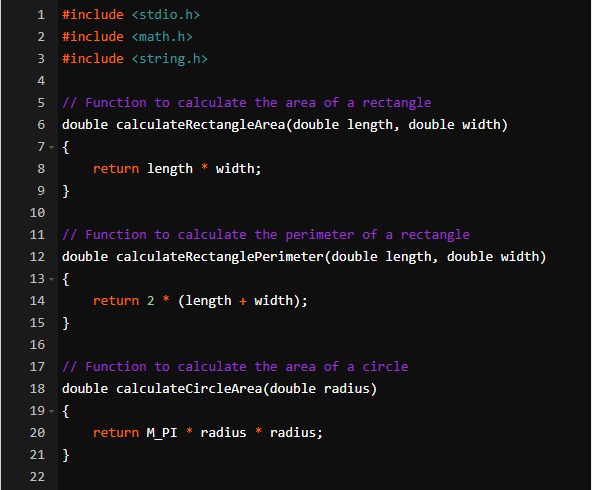
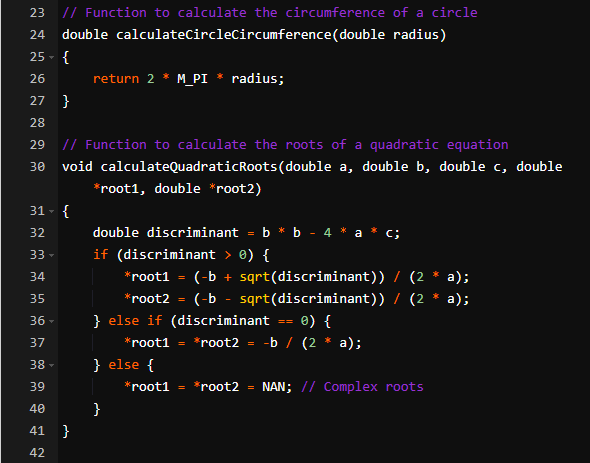
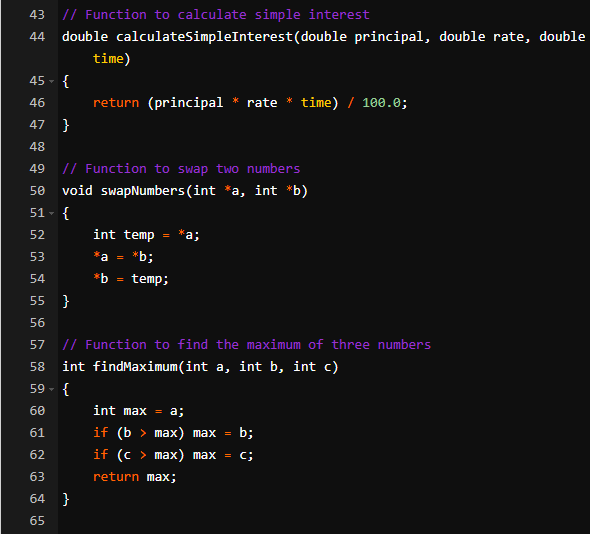
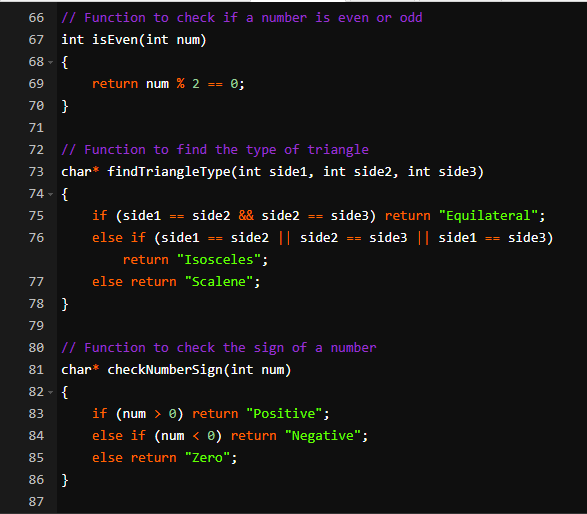
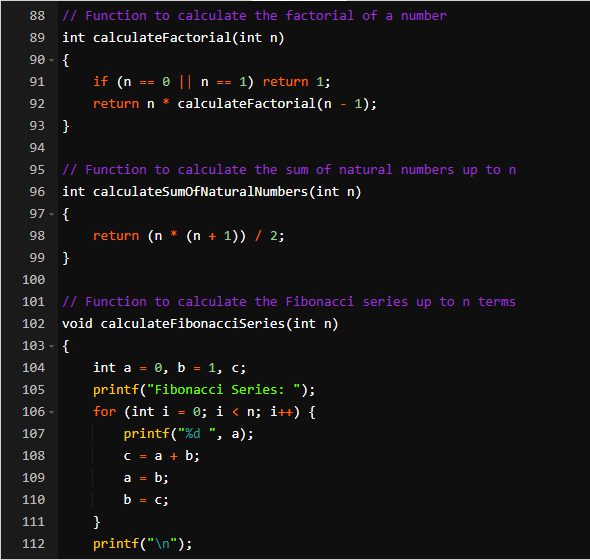
FUNCTIONS

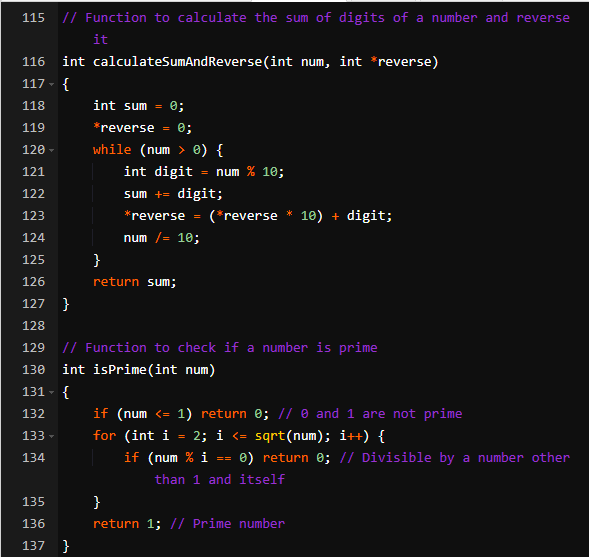


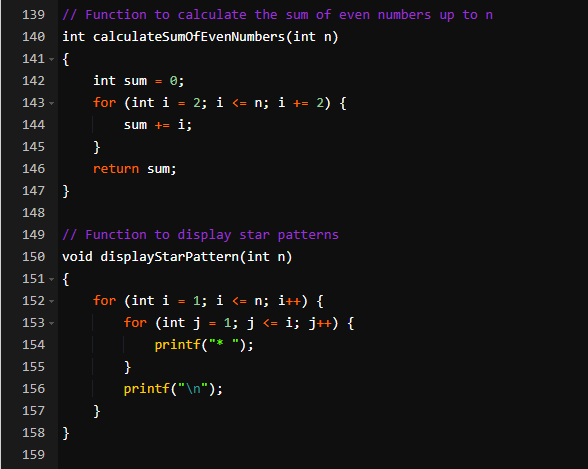


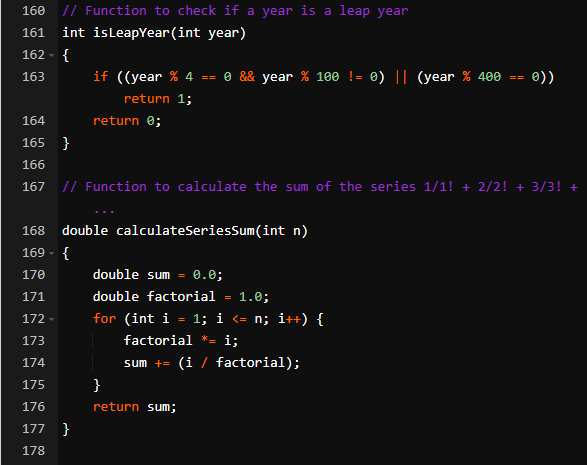


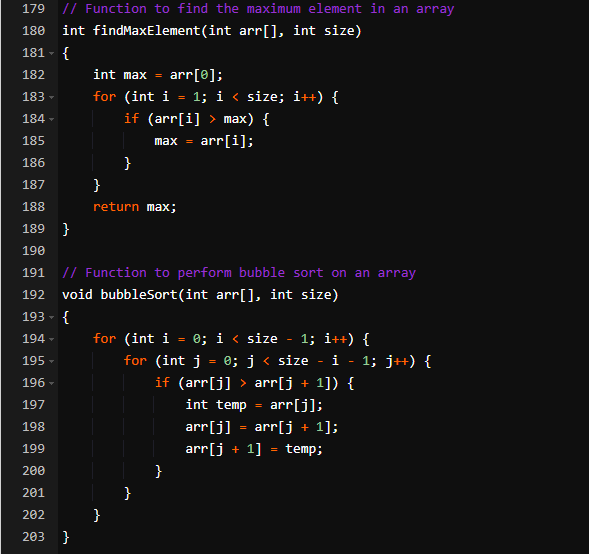


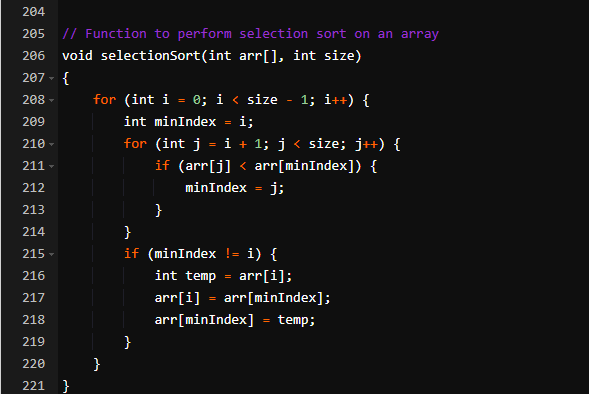












1.

#include <stdio.h>

double calculateRectangleArea(double length, double width) {

return length \* width;

}

double calculateRectanglePerimeter(double length, double width) {

return 2 \* (length + width);

}

int main() {

double length, width;

printf("Enter length and width of the rectangle: ");

scanf("%lf %lf", &length, &width);

double area = calculateRectangleArea(length, width);

double perimeter = calculateRectanglePerimeter(length, width);

printf("Area: %lf\n", area);

printf("Perimeter: %lf\n", perimeter);

return 0;

}

2.

#include <stdio.h>

#include <math.h>

double calculateCircleArea(double radius) {

return M\_PI \* radius \* radius;

}

double calculateCircleCircumference(double radius) {

return 2 \* M\_PI \* radius;

}

int main() {

double radius;

printf("Enter the radius of the circle: ");

scanf("%lf", &radius);

double area = calculateCircleArea(radius);

double circumference = calculateCircleCircumference(radius);

printf("Area: %lf\n", area);

printf("Circumference: %lf\n", circumference);

return 0;

}

3.

#include <stdio.h>

#include <math.h>

void calculateQuadraticRoots(double a, double b, double c, double \*root1, double \*root2) {

double discriminant = b \* b - 4 \* a \* c;

if (discriminant > 0) {

\*root1 = (-b + sqrt(discriminant)) / (2 \* a);

\*root2 = (-b - sqrt(discriminant)) / (2 \* a);

} else if (discriminant == 0) {

\*root1 = \*root2 = -b / (2 \* a);

} else {

\*root1 = \*root2 = NAN; // Complex roots

}

}

int main() {

double a, b, c;

printf("Enter coefficients a, b, and c: ");

scanf("%lf %lf %lf", &a, &b, &c);

double root1, root2;

calculateQuadraticRoots(a, b, c, &root1, &root2);

if (!isnan(root1) && !isnan(root2)) {

printf("Root 1: %lf\n", root1);

printf("Root 2: %lf\n", root2);

} else {

printf("Complex roots\n");

}

return 0;

}

4.

#include <stdio.h>

double calculateSimpleInterest(double principal, double rate, double time) {

return (principal \* rate \* time) / 100.0;

}

int main() {

double principal, rate, time;

printf("Enter principal amount, rate of interest, and time (in years): ");

scanf("%lf %lf %lf", &principal, &rate, &time);

double interest = calculateSimpleInterest(principal, rate, time);

printf("Simple Interest: %lf\n", interest);

return 0;

}

5.

#include <stdio.h>

void swapNumbers(int \*a, int \*b) {

int temp = \*a;

\*a = \*b;

\*b = temp;

}

int main() {

int num1, num2;

printf("Enter two numbers: ");

scanf("%d %d", &num1, &num2);

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

swapNumbers(&num1, &num2);

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

return 0;

}

6.

#include <stdio.h>

int findMaximum(int a, int b, int c) {

int max = a;

if (b > max) max = b;

if (c > max) max = c;

return max;

}

int main() {

int num1, num2, num3;

printf("Enter three numbers: ");

scanf("%d %d %d", &num1, &num2, &num3);

int max = findMaximum(num1, num2, num3);

printf("Maximum: %d\n", max);

return 0;

}

7.

#include <stdio.h>

int isEven(int num) {

return num % 2 == 0;

}

int main() {

int num;

printf("Enter a number: ");

scanf("%d", &num);

if (isEven(num)) {

printf("Even\n");

} else {

printf("Odd\n");

}

return 0;

}

8.

#include <stdio.h>

char\* findTriangleType(int side1, int side2, int side3) {

if (side1 == side2 && side2 == side3) return "Equilateral";

else if (side1 == side2 || side2 == side3 || side1 == side3) return "Isosceles";

else return "Scalene";

}

int main() {

int side1, side2, side3;

printf("Enter three side lengths of a triangle: ");

scanf("%d %d %d", &side1, &side2, &side3);

char\* type = findTriangleType(side1, side2, side3);

printf("Triangle is %s\n", type);

return 0;

}

9.

#include <stdio.h>

char\* checkNumberSign(int num) {

if (num > 0) return "Positive";

else if (num < 0) return "Negative";

else return "Zero";

}

int main() {

int num;

printf("Enter a number: ");

scanf("%d", &num);

char\* sign = checkNumberSign(num);

printf("Number is %s\n", sign);

return 0;

}

10.

#include <stdio.h>

int isLeapYear(int year) {

if ((year % 4 == 0 && year % 100 != 0) || (year % 400 == 0)) return 1;

return 0;

}

int main() {

int year;

printf("Enter a year: ");

scanf("%d", &year);

if (isLeapYear(year)) {

printf("Leap Year\n");

} else {

printf("Not a Leap Year\n");

}

return 0;

}

11

#include <stdio.h>

int calculateSumOfNaturalNumbers(int n) {

return (n \* (n + 1)) / 2;

}

int main() {

int n;

printf("Enter a positive integer n: ");

scanf("%d", &n);

int sum = calculateSumOfNaturalNumbers(n);

printf("Sum of natural numbers up to %d: %d\n", n, sum);

return 0;

}

12.

#include <stdio.h>

int calculateFactorial(int n) {

if (n == 0 || n == 1) return 1;

return n \* calculateFactorial(n - 1);

}

int main() {

int n;

printf("Enter a non-negative integer n: ");

scanf("%d", &n);

int factorial = calculateFactorial(n);

printf("Factorial of %d: %d\n", n, factorial);

return 0;

}

13.

#include <stdio.h>

void calculateFibonacciSeries(int n) {

int a = 0, b = 1, c;

printf("Fibonacci Series: ");

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

printf("%d ", a);

c = a + b;

a = b;

b = c;

}

printf("\n");

}

int main() {

int n;

printf("Enter the number of terms in the Fibonacci series: ");

scanf("%d", &n);

calculateFibonacciSeries(n);

return 0;

}

14.

#include <stdio.h>

int calculateSumAndReverse(int num, int \*reverse) {

int sum = 0;

\*reverse = 0;

while (num > 0) {

int digit = num % 10;

sum += digit;

\*reverse = (\*reverse \* 10) + digit;

num /= 10;

}

return sum;

}

int main() {

int num;

printf("Enter a positive integer: ");

scanf("%d", &num);

int reverse, sum;

sum = calculateSumAndReverse(num, &reverse);

printf("Sum of digits: %d\n", sum);

printf("Reverse of the number: %d\n", reverse);

return 0;

}

15.

#include <stdio.h>

#include <math.h>

int isPrime(int num) {

if (num <= 1) return 0; // 0 and 1 are not prime

for (int i = 2; i <= sqrt(num); i++) {

if (num % i == 0) return 0; // Divisible by a number other than 1 and itself

}

return 1; // Prime number

}

int main() {

int num;

printf("Enter a positive integer: ");

scanf("%d", &num);

if (isPrime(num)) {

printf("Prime Number\n");

} else {

printf("Not a Prime Number\n");

}

return 0;

}

16.

#include <stdio.h>

int calculateSumOfEvenNumbers(int n) {

int sum = 0;

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

sum += i;

}

return sum;

}

int main() {

int n;

printf("Enter a positive integer n: ");

scanf("%d", &n);

int sum = calculateSumOfEvenNumbers(n);

printf("Sum of even numbers up to %d: %d\n", n, sum);

return 0;

}

17.

#include <stdio.h>

void displayStarPattern(int n) {

for (int i = 1; i <= n; i++) {

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

printf("\* ");

}

printf("\n");

}

}

int main() {

int n;

printf("Enter the number of rows: ");

scanf("%d", &n);

displayStarPattern(n);

return 0;

}

18.

#include <stdio.h>

void checkNumber(int num) {

switch (num) {

case 1:

printf("One\n");

break;

case 2:

printf("Two\n");

break;

case 3:

printf("Three\n");

break;

default:

printf("Number is not 1, 2, or 3\n");

}

}

int main() {

int num;

printf("Enter a number (1, 2, or 3): ");

scanf("%d", &num);

checkNumber(num);

return 0;

}

19.

#include <stdio.h>

#include <math.h>

int isArmstrong(int num) {

int originalNum = num;

int n = 0, sum = 0;

while (num != 0) {

num /= 10;

n++;

}

num = originalNum;

while (num != 0) {

int digit = num % 10;

sum += pow(digit, n);

num /= 10;

}

return sum == originalNum;

}

int main() {

int num;

printf("Enter a number: ");

scanf("%d", &num);

if (isArmstrong(num)) {

printf("%d is an Armstrong number.\n", num);

} else {

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

}

return 0;

}

20.

#include <stdio.h>

int calculateFactorial(int n) {

if (n == 0 || n == 1) return 1;

return n \* calculateFactorial(n - 1);

}

void findFactorialsInRange(int start, int end) {

for (int i = start; i <= end; i++) {

int factorial = calculateFactorial(i);

printf("Factorial of %d: %d\n", i, factorial);

}

}

int main() {

int start, end;

printf("Enter the range (start and end): ");

scanf("%d %d", &start, &end);

findFactorialsInRange(start, end);

return 0;

}

21.

#include <stdio.h>

#include <math.h>

int isPrime(int num) {

if (num <= 1) return 0; // 0 and 1 are not prime

for (int i = 2; i <= sqrt(num); i++) {

if (num % i == 0) return 0; // Divisible by a number other than 1 and itself

}

return 1; // Prime number

}

void findPrimesInRange(int start, int end) {

for (int i = start; i <= end; i++) {

if (isPrime(i)) {

printf("%d is a prime number.\n", i);

}

}

}

int main() {

int start, end;

printf("Enter the range (start and end): ");

scanf("%d %d", &start, &end);

findPrimesInRange(start, end);

return 0;

}

22.

#include <stdio.h>

double calculateSeriesSum(int n) {

double sum = 0.0;

double factorial = 1.0;

for (int i = 1; i <= n; i++) {

factorial \*= i;

sum += (i / factorial);

}

return sum;

}

int main() {

int n;

printf("Enter the number of terms in the series: ");

scanf("%d", &n);

double sum = calculateSeriesSum(n);

printf("Sum of the series: %lf\n", sum);

return 0;

}

23.

#include <stdio.h>

int main() {

int num;

printf("Enter a number: ");

scanf("%d", &num);

// Ternary operator to check if the number is even or odd

(num % 2 == 0) ? printf("Even\n") : printf("Odd\n");

return 0;

}

24.

#include <stdio.h>

int main() {

int num1, num2;

printf("Enter two numbers: ");

scanf("%d %d", &num1, &num2);

// Bitwise AND

int resultAnd = num1 & num2;

printf("Bitwise AND: %d\n", resultAnd);

// Bitwise OR

int resultOr = num1 | num2;

printf("Bitwise OR: %d\n", resultOr);

// Bitwise XOR

int resultXor = num1 ^ num2;

printf("Bitwise XOR: %d\n", resultXor);

// Bitwise NOT

int resultNot1 = ~num1;

int resultNot2 = ~num2;

printf("Bitwise NOT of num1: %d\n", resultNot1);

printf("Bitwise NOT of num2: %d\n", resultNot2);

return 0;

}

25.

#include <stdio.h>

int findMaxElement(int arr[], int size) {

int max = arr[0];

for (int i = 1; i < size; i++) {

if (arr[i] > max) {

max = arr[i];

}

}

return max;

}

int main() {

int size;

printf("Enter the size of the array: ");

scanf("%d", &size);

int arr[size];

printf("Enter %d elements:\n", size);

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

scanf("%d", &arr[i]);

}

int max = findMaxElement(arr, size);

printf("Maximum element in the array: %d\n", max);

return 0;

}

26.

#include <stdio.h>

void bubbleSort(int arr[], int size) {

for (int i = 0; i < size - 1; i++) {

for (int j = 0; j < size - i - 1; j++) {

if (arr[j] > arr[j + 1]) {

int temp = arr[j];

arr[j] = arr[j + 1];

arr[j + 1] = temp;

}

}

}

}

int main() {

int size;

printf("Enter the size of the array: ");

scanf("%d", &size);

int arr[size];

printf("Enter %d elements:\n", size);

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

scanf("%d", &arr[i]);

}

bubbleSort(arr, size);

printf("Sorted array using Bubble Sort: ");

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

printf("%d ", arr[i]);

}

printf("\n");

return 0;

}

27.

#include <stdio.h>

void selectionSort(int arr[], int size) {

for (int i = 0; i < size - 1; i++) {

int minIndex = i;

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

if (arr[j] < arr[minIndex]) {

minIndex = j;

}

}

if (minIndex != i) {

int temp = arr[i];

arr[i] = arr[minIndex];

arr[minIndex] = temp;

}

}

}

int main() {

int size;

printf("Enter the size of the array: ");

scanf("%d", &size);

int arr[size];

printf("Enter %d elements:\n", size);

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

scanf("%d", &arr[i]);

}

selectionSort(arr, size);

printf("Sorted array using Selection Sort: ");

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

printf("%d ", arr[i]);

}

printf("\n");

return 0;

}

28.

#include <stdio.h>

void addArrays(int arr1[], int arr2[], int result[], int size) {

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

result[i] = arr1[i] + arr2[i];

}

}

int main() {

int size;

printf("Enter the size of the arrays: ");

scanf("%d", &size);

int arr1[size], arr2[size], result[size];

printf("Enter %d elements for the first array:\n", size);

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

scanf("%d", &arr1[i]);

}

printf("Enter %d elements for the second array:\n", size);

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

scanf("%d", &arr2[i]);

}

addArrays(arr1, arr2, result, size);

printf("Resultant array after addition: ");

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

printf("%d ", result[i]);

}

printf("\n");

return 0;

}

29.

#include <stdio.h>

#include <string.h>

int calculateStringLength(char str[]) {

int length = 0;

while (str[length] != '\0') {

length++;

}

return length;

}

int main() {

char str[100];

printf("Enter a string: ");

scanf("%s", str);

int length = calculateStringLength(str);

printf("Length of the string: %d\n", length);

return 0;

}

30.

#include <stdio.h>

#include <string.h>

int main() {

char str1[100], str2[100];

printf("Enter the first string: ");

scanf("%s", str1);

printf("Enter the second string: ");

scanf("%s", str2);

// String Comparison

int cmpResult = strcmp(str1, str2);

if (cmpResult == 0) {

printf("Strings are equal.\n");

} else if (cmpResult < 0) {

printf("First string is lexicographically smaller.\n");

} else {

printf("Second string is lexicographically smaller.\n");

}

// String Copy

strcpy(str1, str2);

printf("First string after copying the second string: %s\n", str1);

// String Concatenation

strcat(str1, str2);

printf("Concatenated string: %s\n", str1);

return 0;

}

31.

#include <stdio.h>

#include <string.h>

int main() {

char str[100];

printf("Enter a string: ");

scanf("%s", str);

int length = strlen(str);

printf("Length of the string: %d\n", length);

return 0;

}

32.

#include <stdio.h>

#include <string.h>

int main() {

char str1[100], str2[100];

printf("Enter the first string: ");

scanf("%s", str1);

printf("Enter the second string: ");

scanf("%s", str2);

int cmpResult = strcmp(str1, str2);

if (cmpResult == 0) {

printf("Strings are equal.\n");

} else {

printf("Strings are not equal.\n");

}

return 0;

}

33.

#include <stdio.h>

#include <string.h>

int main() {

char source[100], destination[100];

printf("Enter a string to copy from: ");

scanf("%s", source);

strcpy(destination, source);

printf("Copied string: %s\n", destination);

return 0;

}

34.

#include <stdio.h>

#include <string.h>

int main() {

char str1[100], str2[100];

printf("Enter the first string: ");

scanf("%s", str1);

printf("Enter the second string: ");

scanf("%s", str2);

strcat(str1, str2);

printf("Concatenated string: %s\n", str1);

return 0;

}

35.

#include <stdio.h>

#include <string.h>

void reverseString(char str[]) {

int length = strlen(str);

for (int i = 0, j = length - 1; i < j; i++, j--) {

char temp = str[i];

str[i] = str[j];

str[j] = temp;

}

}

int main() {

char str[100];

printf("Enter a string: ");

scanf("%s", str);

reverseString(str);

printf("Reversed string: %s\n", str);

return 0;

}

36.

#include <stdio.h>

void displayArray(int arr[][3], int rows, int cols) {

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

for (int j = 0; j < cols; j++) {

printf("%d ", arr[i][j]);

}

printf("\n");

}

}

int main() {

int rows, cols;

printf("Enter the number of rows and columns: ");

scanf("%d %d", &rows, &cols);

int arr[rows][cols];

printf("Enter elements for the array:\n");

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

for (int j = 0; j < cols; j++) {

scanf("%d", &arr[i][j]);

}

}

printf("Array:\n");

displayArray(arr, rows, cols);

return 0;

}

37.

#include <stdio.h>

void addArrays(int arr1[][3], int arr2[][3], int result[][3], int rows, int cols) {

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

for (int j = 0; j < cols; j++) {

result[i][j] = arr1[i][j] + arr2[i][j];

}

}

}

void subtractArrays(int arr1[][3], int arr2[][3], int result[][3], int rows, int cols) {

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

for (int j = 0; j < cols; j++) {

result[i][j] = arr1[i][j] - arr2[i][j];

}

}

}

void displayArray(int arr[][3], int rows, int cols) {

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

for (int j = 0; j < cols; j++) {

printf("%d ", arr[i][j]);

}

printf("\n");

}

}

int main() {

int rows, cols;

printf("Enter the number of rows and columns: ");

scanf("%d %d", &rows, &cols);

int arr1[rows][cols], arr2[rows][cols], result[rows][cols];

printf("Enter elements for the first array:\n");

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

for (int j = 0; j < cols; j++) {

scanf("%d", &arr1[i][j]);

}

}

printf("Enter elements for the second array:\n");

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

for (int j = 0; j < cols; j++) {

scanf("%d", &arr2[i][j]);

}

}

// Addition

addArrays(arr1, arr2, result, rows, cols);

printf("Resultant array after addition:\n");

displayArray(result, rows, cols);

// Subtraction

subtractArrays(arr1, arr2, result, rows, cols);

printf("Resultant array after subtraction:\n");

displayArray(result, rows, cols);

return 0;

}

38.

#include <stdio.h>

void multiplyArrays(int arr1[][3], int arr2[][3], int result[][3], int rows1, int cols1, int cols2) {

for (int i = 0; i < rows1; i++) {

for (int j = 0; j < cols2; j++) {

result[i][j] = 0;

for (int k = 0; k < cols1; k++) {

result[i][j] += arr1[i][k] \* arr2[k][j];

}

}

}

}

void displayArray(int arr[][3], int rows, int cols) {

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

for (int j = 0; j < cols; j++) {

printf("%d ", arr[i][j]);

}

printf("\n");

}

}

int main() {

int rows1, cols1, rows2, cols2;

printf("Enter the number of rows and columns for the first matrix: ");

scanf("%d %d", &rows1, &cols1);

printf("Enter the number of rows and columns for the second matrix: ");

scanf("%d %d", &rows2, &cols2);

if (cols1 != rows2) {

printf("Matrix multiplication not possible.\n");

return 1;

}

int arr1[rows1][cols1], arr2[rows2][cols2], result[rows1][cols2];

printf("Enter elements for the first matrix:\n");

for (int i = 0; i < rows1; i++) {

for (int j = 0; j < cols1; j++) {

scanf("%d", &arr1[i][j]);

}

}

printf("Enter elements for the second matrix:\n");

for (int i = 0; i < rows2; i++) {

for (int j = 0; j < cols2; j++) {

scanf("%d", &arr2[i][j]);

}

}

// Multiplication

multiplyArrays(arr1, arr2, result, rows1, cols1, cols2);

printf("Resultant array after multiplication:\n");

displayArray(result, rows1, cols2);

return 0;

}