**C-programming**

**G.E.U**



Batch

2023-2027

BCA (hons.) with Ai and Ds

Submitted by: Submitted to:

Navjot Jangra Mr. Rishi kumar

231601096 Astt Prof. CSIT , G.E.U

**Index**

|  |  |  |  |
| --- | --- | --- | --- |
| S. No. | Programme | Faculty sign | Remarks |
| 1. | Hello World |  |  |
| 2. | Add two numbers |  |  |
| 3. | Find area of circle |  |  |
| 4. | Divide two numbers |  |  |
| 5. | Print ASCII value |  |  |
| 6. | Multiplying floating numbers |  |  |
| 7. | Swap two variables number by using third variable |  |  |
| 8. | Swap two variables number without using third variable |  |  |
| 9. | Swap three variables number without using fourth variable |  |  |
| 10. | Area of Rectangle |  |  |
| 11. | Area of Square |  |  |
| 12. | Area of Triangle |  |  |
| 13. | Area and Volume of Cube |  |  |
| 14. | Area and Volume of Cuboid |  |  |
| 15. | Find the largest number using Logical AND operator |  |  |
| 16. | Validate the username and password entered by the user is correct or not using the predefined username and password. |  |  |
| 17. | Input the positive number from the user to perform the left shift operator. |  |  |
| 18. | Input the positive number from the user to perform the right shift operator. |  |  |
| 19. | Perform the pre increment and pre decrement operator on two integers and print both original value and updated value. |  |  |
| 20. | Perform the post increment and post decrement operator on two integers and print both original value and updated value. |  |  |
| 21. | Integer number and to check whether it is divisible by 9 or 7 using OR logical operator. |  |  |
| 22. | Identify gender in single character and print full gender (Ex: if input is ‘M’ or ‘m’ – it should print “Male”) |  |  |
| 23. | Write a C program to print all natural numbers in reverse (from n to 1) |  |  |
| 24. | Write a C program to print all alphabets from a to z. |  |  |
| 25. | Write a C program to print all natural numbers from 1 to n. |  |  |
| 26. | program to print all even numbers between 1 to 100 |  |  |
| 27. | Write a C program to print all odd number between 1 to 100. |  |  |
| 28. | Write a C program to find sum of all natural numbers between 1 to n. |  |  |
| 29. | Write a C program to find sum of all even numbers between 1 to n |  |  |
| 30. | Write a C program to find sum of all odd numbers between 1 to n. |  |  |
| 31. | Write a C program to print multiplication table of any number. |  |  |
| 32. | Write a C program to count number of digits in a number. |  |  |
| 33. | Write a C program to find first and last digit of a number. |  |  |
| 34. | Write a C program to find sum of first and last digit of a number. |  |  |
| 35. | Write a C program to swap first and last digits of a number. |  |  |
| 36. | Write a C program to calculate sum of digits of a number. |  |  |
| 37. | Write a C program to calculate product of digits of a number. |  |  |
| 38. | Write a C program to enter a number and print its reverse. |  |  |
| 39. | Write a C program to check whether a number is palindrome or not. |  |  |
| 40. | Write a C program to find frequency of each digit in a given integer. |  |  |
| 41. | Write a C program to enter a number and print it in words. |  |  |
| 42. | Write a C program to print all ASCII character with their values |  |  |
| 43. | Write a C program to find power of a number using for loop |  |  |
| 44. | Write a C program to find all factors of a number |  |  |
| 45. | Write a C program to calculate factorial of a number |  |  |
| 46. | Write a C program to find HCF (GCD) of two numbers. |  |  |
| 47. | Write a C program to find LCM of two numbers. |  |  |
| 48. | Write a C program to check whether a number is Prime number or not. |  |  |
| 49. | Write a C program to print all Prime numbers between 1 to n. |  |  |
| 50. | Write a C program to find sum of all prime numbers between 1 to n. |  |  |
| 51. | Write a C program to find all prime factors of a number. |  |  |
| 52. | Write a C program to check whether a number is Armstrong number or not. |  |  |
| 53. | Write a C program to print all Armstrong numbers between 1 to n. |  |  |
| 54. | Write a C program to check whether a number is Perfect number or not. |  |  |
| 55. | Write a C program to print all Perfect numbers between 1 to n. |  |  |
| 56. | Write a C program to check whether a number is Strong number or not. |  |  |
| 57. | Write a C program to print all Strong numbers between 1 to n. |  |  |
| 58. | Write a C program to print Fibonacci series up to n terms. |  |  |
| 59. | Write a C program to find one's complement of a binary number. |  |  |
| 60. | Write a C program to find two's complement of a binary number. |  |  |
| 61. | Write a C program to convert Binary to Octal number system. |  |  |
| 62. | Write a C program to convert Binary to Decimal number system. |  |  |
| 63. | Write a C program to convert Binary to Hexadecimal number system |  |  |
| 64. | Write a C program to convert Octal to Binary number system. |  |  |
| 65. | Write a C program to convert Octal to Decimal number system. |  |  |
| 66. | Write a C program to convert Octal to Hexadecimal number system. |  |  |
| 67. | Write a C program to convert Decimal to Binary number system. |  |  |
| 68. | Write a C program to convert Decimal to Octal number system. |  |  |
| 69. | Write a C program to convert Decimal to Hexadecimal number system. |  |  |
| 70. | Write a C program to convert Hexadecimal to Binary number system. |  |  |
| 71. | Write a C program to convert Hexadecimal to Octal number system |  |  |
| 72. | Write a C program to convert Hexadecimal to Decimal number system. |  |  |
| 73. | Star pattern programs - Write a C program to print the given star patterns |  |  |
| 74. | Number pattern programs - Write a C program to print the given number patterns |  |  |
| 75. | Write a C program to find maximum between two numbers |  |  |
| 76. | Write a C program to find maximum between three numbers. |  |  |
| 77. | Write a C program to check whether a number is negative, positive or zero. |  |  |
| 78. | Write a C program to check whether a number is divisible by 5 and 11 or not. |  |  |
| 79. | Write a C program to check whether a number is even or odd. |  |  |
| 80. | Write a C program to check whether a year is leap year or not. |  |  |
| 81. | Write a C program to check whether a character is alphabet or not. |  |  |
| 82. | Write a C program to input any alphabet and check whether it is vowel or consonant. |  |  |
| 83. | Write a C program to input any character and check whether it is alphabet, digit or special character |  |  |
| 84. | Write a C program to check whether a character is uppercase or lowercase alphabet. |  |  |
| 85. | Write a C program to input week number and print week day. |  |  |
| 86. | Write a C program to input month number and print number of days in that month. |  |  |
| 87. | Write a C program to count total number of notes in given amount. |  |  |
| 88. | Write a C program to input month number and print number of days in that month. |  |  |
| 89. | Write a C program to count total number of notes in given amount. |  |  |
| 90. | Write a C program to input angles of a triangle and check whether triangle is valid or not. |  |  |
| 91. | Write a C program to input all sides of a triangle and check whether triangle is valid or not |  |  |
| 92. | Write a C program to check whether the triangle is equilateral, isosceles or scalene triangle. |  |  |
| 93. | Write a C program to find all roots of a quadratic equation. |  |  |
| 94. | Write a C program to calculate profit or loss. |  |  |
| 95. | Write a C program to input marks of five subjects Physics, Chemistry, Biology, Mathematics and Computer. Calculate percentage and grade |  |  |
| 96. | Write a C program to input basic salary of an employee and calculate its Gross salary according to following: |  |  |
| 97. | Write a C program to input electricity unit charges and calculate total electricity bill according to the given condition: |  |  |
| 98. | Write a C program to convert specified days into years, weeks and days. |  |  |
| 99. | Write a program in C to read n number of values in an array and display them in reverse order. |  |  |
| 100. | Write a program in C to find the sum of all elements of the array. |  |  |
| 101. | Write a program in C to copy the elements of one array into another array. |  |  |
| 102. | Write a program in C to find the sum of all elements of the array |  |  |
| 103. | Write a program in C to copy the elements of one array into another array. |  |  |
| 104. | Write a program in C to count the total number of duplicate elements in an array |  |  |
| 105. | Write a program in C to find the maximum and minimum elements in an array. |  |  |
| 106. | Write a C program to sort the elements of an array in descending order. |  |  |
| 107. | Write a program in C to separate odd and even integers into separate arrays. |  |  |
| 108. | Write a program in C to merge two arrays of the same size sorted in descending/ascending order. |  |  |
| 109. | Write a program in C to merge two arrays of the same size sorted in descending order. |  |  |
| 110. | WAP using Switch case: Consider two matrices of the size m and n. Implement matrix operation and display. Show these things in program 1) Read matrix elements and display 2) Matrix Multiplication and display 3) addition of matrix and display 4)Subtraction of Matrix and display 5)Transpose of Matrix and display. |  |  |
| 111. | Write a program in C to read n number of values in an array and display them in reverse order: |  |  |
| 112. | Write a program in C to find the sum of all elements of the array: |  |  |
| 113. | Write a program in C to copy the elements of one array into another array: |  |  |
| 114. | Write a program in C to count the total number of duplicate elements in an array: |  |  |
| 115. | Write a program in C to find the maximum and minimum elements in an array: |  |  |
| 116. | Write a C program to sort the elements of an array in descending order: |  |  |
| 117. | Write a program in C to separate odd and even integers into separate arrays: |  |  |
| 118. | Write a program in C to merge two arrays of the same size sorted in descending/ascending order: |  |  |

1. Hello world

#include <stdio.h>

int main() { printf(“Navjot”);

printf("Hello, World!\n");

return 0;

}

1. Add two numbers

#include <stdio.h>

int main() { printf(“Navjot”);

double num1, num2, sum;

// Ask the user to enter two numbers

printf("Enter the first number: ");

scanf("%lf", &num1);

printf("Enter the second number: ");

scanf("%lf", &num2);

// Add the two numbers

sum = num1 + num2;

// Display the result

printf("Sum: %.2lf\n", sum);

return 0;

}

1. Find area of Circle

#include <stdio.h>

#include <math.h>

int main() { printf(“Navjot”);

double radius, area;

// Ask the user to enter the radius of the circle

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

scanf("%lf", &radius);

// Calculate the area of the circle using the formula: area = π \* radius \* radius

area = M\_PI \* pow(radius, 2);

// Display the result

printf("Area of the circle: %.2lf\n", area);

return 0;

}

1. Divide two numbers

#include <stdio.h>

int main() { printf(“Navjot”);

// Declare variables to store numbers

double num1, num2, result;

// Ask the user for the first number

printf("Enter the first number: ");

scanf("%lf", &num1);

// Ask the user for the second number

printf("Enter the second number: ");

scanf("%lf", &num2);

// Calculate the division, assuming num2 is not zero

result = num1 / num2;

// Display the result

printf("Result of division: %.2lf\n", result);

return 0;

}

1. Print ASCII value

#include <stdio.h>

int main() { printf(“Navjot”);

char character;

// Ask the user to enter a character

printf("Enter a character: ");

scanf(" %c", &character);

// Display the ASCII value of the character

printf("ASCII value of '%c' is: %d\n", character, character);

return 0;

}

1. Multiplying floating numbers

#include <stdio.h>

int main() { printf(“Navjot”);

float num1, num2, result;

// Ask the user to enter the first floating-point number

printf("Enter the first number: ");

scanf("%f", &num1);

// Ask the user to enter the second floating-point number

printf("Enter the second number: ");

scanf("%f", &num2);

// Calculate the multiplication

result = num1 \* num2;

// Display the result

printf("Result of multiplication: %.2f\n", result);

return 0;

}

1. Swap two variables by using third variable

#include <stdio.h>

int main() { printf(“Navjot”);

// Declare three variables to store numbers

float num1, num2, temp;

// Ask the user to enter the first number

printf("Enter the first number: ");

scanf("%f", &num1);

// Ask the user to enter the second number

printf("Enter the second number: ");

scanf("%f", &num2);

// Display the original values

printf("Original values:\n");

printf("num1 = %.2f\n", num1);

printf("num2 = %.2f\n", num2);

// Swap the values using a third variable

temp = num1;

num1 = num2;

num2 = temp;

// Display the swapped values

printf("\nAfter swapping:\n");

printf("num1 = %.2f\n", num1);

printf("num2 = %.2f\n", num2);

return 0;

}

1. Swap two variable without using third variable

#include <stdio.h>

int main() { printf(“Navjot”);

printf(“Navjot”);

// Declare two variables to store numbers

float num1, num2;

// Ask the user to enter the first number

printf("Enter the first number: ");

scanf("%f", &num1);

// Ask the user to enter the second number

printf("Enter the second number: ");

scanf("%f", &num2);

// Display the original values

printf("Original values:\n");

printf("num1 = %.2f\n", num1);

printf("num2 = %.2f\n", num2);

// Swap the values without a third variable

num1 = num1 + num2;

num2 = num1 - num2;

num1 = num1 - num2;

// Display the swapped values

printf("\nAfter swapping:\n");

printf("num1 = %.2f\n", num1);

printf("num2 = %.2f\n", num2);

return 0;

}

1. Swap three variables without using fourth variable

#include <stdio.h>

int main() { printf(“Navjot”);

// Declare three variables to store numbers

int num1, num2, num3;

// Ask the user to enter the first number

printf("Enter the first number: ");

scanf("%d", &num1);

// Ask the user to enter the second number

printf("Enter the second number: ");

scanf("%d", &num2);

// Ask the user to enter the third number

printf("Enter the third number: ");

scanf("%d", &num3);

// Display the original values

printf("Original values:\n");

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

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

printf("num3 = %d\n", num3);

// Swap the values without a fourth variable

num1 = num1 + num2 + num3;

num2 = num1 - (num2 + num3);

num3 = num1 - (num2 + num3);

num1 = num1 - (num2 + num3);

// Display the swapped values

printf("\nAfter swapping:\n");

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

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

printf("num3 = %d\n", num3);

return 0;

}

1. Area of Rectangle

#include<stdio.h>

int main(){

printf(“Navjot”);

float length, width, area;

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

scanf("%f", &length);

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

scanf("%f", &width);

//Calculate the area of the rectangle

area= length \* width;

printf("The area of the rectangle is: %.2f square units\n",area);

return 0;

}

1. Area of Square

#include<stdio.h>

int main(){

printf(“Navjot”);

float side, area;

printf("Enter the side of the square:");

scanf("%f", &side);

//Calculate the area of the square

area= side \* side;

printf("The area of the square is: %.2f square units\n",area);

return 0;

}

1. Area of Triangle

#include<stdio.h>

int main(){

printf(“Navjot”);

float base, height, area;

printf("Enter the base legth of the triangle:");

scanf("%f", &base);

printf("Enter the height of the triangle:");

scanf("%f", &height);

//Calculate the area of the triangle

area= (base \* height)/2;

printf("The area of the triangle is: %.2f square units\n",area);

return 0;

}

1. Area and Volume of Cube

#include<stdio.h>

int main(){

printf(“Navjot”);

float side, total\_surface\_area, volume;

printf("Enter the side of the cube:");

scanf("%f", &side);

//Calculate the total\_surface\_area of the cube

total\_surface\_area= 6 \*(side \* side);

//Calculate the volume of the cube

volume = side \* side \* side;

printf("The total\_surface\_area of the cube is: %.2f square units\n",total\_surface\_area);

printf("The volume of the cube is: %.2f cubic units\n",volume);

return 0;

}

1. Area and Volume of Cuboid

#include<stdio.h>

int main(){

printf(“Navjot”);

float length, width, height, total\_surface\_area, volume;

printf("Enter the length of the cuboid:");

scanf("%f", &length);

printf("Enter the width of the cuboid:");

scanf("%f", &width);

printf("Enter the height of the cuboid:");

scanf("%f", &height);

//Calculate the total\_surface\_area of the cuboid

total\_surface\_area= 2 \*(length \* width + width \* height + height \* length);

//Calculate the volume of the cuboid

volume = length \* width \* height;

printf("The total\_surface\_area of the cuboid is: %.2f square units\n",total\_surface\_area);

printf("The volume of the cuboid is: %.2f cubic units\n",volume);

return 0;

}

1. Find the largest number using Logical AND operator

#include <stdio.h>

int main() { printf(“Navjot”);

printf(“Navjot”);

int a, b;

// Input values for a and b

printf("Enter two numbers: ");

scanf("%d %d", &a, &b);

int largest = (a > b && a != b) ? a : b;

printf("%d is the largest number.\n", largest);

return 0;

}

1. Validate the username and password entered by the user is correct or not using the predefined username and password.

#include <stdio.h>

#include <string.h>

int main() { printf(“Navjot”);

// Predefined username and password

char predefinedUsername[] = "myuser";

char predefinedPassword[] = "mypassword";

// Variables for user input

char enteredUsername[20];

char enteredPassword[20];

// Input username and password from the user

printf("Enter username: ");

scanf("%s", enteredUsername);

printf("Enter password: ");

scanf("%s", enteredPassword);

// Check if the entered username and password match the predefined credentials

int usernameIsValid = strcmp(enteredUsername, predefinedUsername) == 0;

int passwordIsValid = strcmp(enteredPassword, predefinedPassword) == 0;

// Use the results to print the appropriate message

if (usernameIsValid && passwordIsValid) {

printf("Login successful!\n");

}

if (!usernameIsValid || !passwordIsValid) {

printf("Login failed. Invalid username or password.\n");

}

return 0;

}

1. Input the positive number from the user to perform the left shift operator.

#include <stdio.h>

int main() { printf(“Navjot”);

int number, shift;

printf(“Navjot”);

// Input a positive number from the user

do {

printf("Enter a positive number: ");

scanf("%d", &number);

} while (number <= 0);

// Input the number of positions to shift

printf("Enter the number of positions to left-shift: ");

scanf("%d", &shift);

// Perform the left shift operation

int result = number << shift;

// Display the result

printf("Result of left-shift: %d\n", result);

return 0;

}

1. Input the positive number from the user to perform the right shift operator.

#include <stdio.h>

int main() { printf(“Navjot”);

int number, shift;

printf(“Navjot”);

// Input a positive number from the user

do {

printf("Enter a positive number: ");

scanf("%d", &number);

} while (number <= 0);

// Input the number of positions to right-shift

printf("Enter the number of positions to right-shift: ");

scanf("%d", &shift);

// Perform the right shift operation

int result = number >> shift;

// Display the result

printf("Result of right-shift: %d\n", result);

return 0;

}

1. Perform the pre increment and pre decrement operator on two integers and print both original value and updated value.

#include <stdio.h>

int main() { printf(“Navjot”);

int num1, num2;

printf(“Navjot”);

// Input two integers from the user

printf("Enter the first integer: ");

scanf("%d", &num1);

printf("Enter the second integer: ");

scanf("%d", &num2);

// Perform pre-increment and pre-decrement operations

int preIncrementResult1 = ++num1;

int preDecrementResult2 = --num2;

// Print original and updated values

printf("Original num1: %d, Updated num1 (after pre-increment): %d\n", num1 - 1, preIncrementResult1);

printf("Original num2: %d, Updated num2 (after pre-decrement): %d\n", num2 + 1, preDecrementResult2);

return 0;

}

1. Perform the post increment and post decrement operator on two integers and print both original value and updated value.

#include <stdio.h>

int main() { printf(“Navjot”);

int num1, num2;

printf(“Navjot”);

// Input two integers from the user

printf("Enter the first integer: ");

scanf("%d", &num1);

printf("Enter the second integer: ");

scanf("%d", &num2);

// Perform post-increment and post-decrement operations

int postIncrementResult1 = num1++;

int postDecrementResult2 = num2--;

// Print original and updated values

printf("Original num1: %d, Updated num1 (after post-increment): %d\n", num1 - 1, postIncrementResult1);

printf("Original num2: %d, Updated num2 (after post-decrement): %d\n", num2 + 1, postDecrementResult2);

return 0;

}

1. Integer number and to check whether it is divisible by 9 or 7 using OR logical operator.

#include <stdio.h>

int main() { printf(“Navjot”);

int number;

// Input an integer from the user

printf("Enter an integer: ");

scanf("%d", &number);

// Check if the number is divisible by 9 or 7

if (number % 9 == 0 || number % 7 == 0) {

printf("%d is divisible by either 9 or 7.\n", number);

} else {

printf("%d is not divisible by either 9 or 7.\n", number);

}

return 0;

}

1. Identify gender in single character and print full gender (Ex: if input is ‘M’ or ‘m’ – it should print “Male”).

#include <stdio.h>

int main() { printf(“Navjot”);

char gender;

printf(“Navjot”);

// Input a single character representing gender

printf("Enter a single character (M/F): ");

scanf(" %c", &gender); // Note the space before %c to skip leading whitespace

// Check the input and print the corresponding gender

if (gender == 'M' || gender == 'm') {

printf("Male\n");

}

if (gender == 'F' || gender == 'f') {

printf("Female\n");

}

return 0;

}

1. Write a C programme to print all natural numbers in reverse.

#include <stdio.h>

int main() { printf(“Navjot”);

int n;

printf(“Navjot”);

printf("Enter a natural number (n): ");

scanf("%d", &n);

if (n < 1) {

printf("Please enter a valid natural number (greater than 0).\n");

} else {

printf("Natural numbers in reverse order from %d to 1 are:\n", n);

for (int i = n; i >= 1; i--) {

printf("%d\n", i);

}

}

return 0;

}

1. Write a C program to print all alphabets from a to z.

#include <stdio.h>

int main() { printf(“Navjot”);

printf(“Navjot”);

char letter; // Declare a character variable

printf("Alphabets from 'a' to 'z' are:\n");

for (letter = 'a'; letter <= 'z'; letter++) {

printf("%c ", letter);

}

printf("\n");

return 0;

}

1. Write a C program to print all natural numbers from 1 to n.

#include <stdio.h>

int main() { printf(“Navjot”);

int n;

printf(“Navjot”);

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

scanf("%d", &n);

if (n <= 0) {

printf("Please enter a valid positive integer.\n");

} else {

printf("Natural numbers from 1 to %d are:\n", n);

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

printf("%d\n", i);

}

}

return 0;

}

1. program to print all even numbers between 1 to 100.

#include <stdio.h>

int main() { printf(“Navjot”);

printf(“Navjot”);

printf("Even numbers between 1 and 100 are:\n");

for (int i = 2; i <= 100; i += 2) {

printf("%d\n", i);

}

return 0;

}

1. Write a C program to print all odd number between 1 to 100.

#include <stdio.h>

int main() { printf(“Navjot”);

printf("Odd numbers between 1 and 100 are:\n");

for (int i = 1; i <= 100; i += 2) {

printf("%d\n", i);

}

return 0;

}

1. Write a C program to find sum of all natural numbers between 1 to n.

#include <stdio.h>

int main() { printf(“Navjot”);

int n, sum = 0;

printf(“Navjot”);

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

scanf("%d", &n);

if (n <= 0) {

printf("Please enter a valid positive integer.\n");

} else {

// Calculate the sum of natural numbers from 1 to n

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

sum += i;

}

printf("The sum of natural numbers from 1 to %d is: %d\n", n, sum);

}

return 0;

}

1. Write a C program to find sum of all even numbers between 1 to n.

#include <stdio.h>

int main() { printf(“Navjot”);

int n, sum = 0;

printf(“Navjot”);

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

scanf("%d", &n);

if (n <= 0) {

printf("Please enter a valid positive integer.\n");

} else {

// Calculate the sum of even numbers from 1 to n

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

sum += i;

}

printf("The sum of even numbers from 1 to %d is: %d\n", n, sum);

}

return 0;

}

1. Write a C program to find sum of all odd numbers between 1 to n.

#include <stdio.h>

int main() { printf(“Navjot”);

int n, sum = 0;

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

scanf("%d", &n);

if (n <= 0) {

printf("Please enter a valid positive integer.\n");

} else {

// Calculate the sum of odd numbers from 1 to n

for (int i = 1; i <= n; i += 2) {

sum += i;

}

printf("The sum of odd numbers from 1 to %d is: %d\n", n, sum);

}

return 0;

}

1. Write a C program to print multiplication table of any number.

#include <stdio.h>

int main() { printf(“Navjot”);

int number;

printf("Enter the number for the multiplication table: ");

scanf("%d", &number);

printf("Multiplication table for %d:\n", number);

for (int i = 1; i <= 10; i++) {

printf("%d x %d = %d\n", number, i, number \* i);

}

return 0;

}

1. Write a C program to count number of digits in a number.

#include <stdio.h>

int main() { printf(“Navjot”);

int number;

int count = 0;

printf("Enter an integer: ");

scanf("%d", &number);

// Ensure the number is positive (absolute value)

if (number < 0) {

number = -number;

}

// Count the number of digits

do {

count++;

number /= 10;

} while (number > 0);

printf("The number of digits in the entered integer is: %d\n", count);

return 0;

}

1. Write a C program to find first and last digit of a number.

#include <stdio.h>

int main() { printf(“Navjot”);

int number, firstDigit, lastDigit;

printf("Enter an integer: ");

scanf("%d", &number);

// To find the last digit, take the remainder when divided by 10

lastDigit = number % 10;

// To find the first digit, repeatedly divide by 10 until only one digit remains

firstDigit = number;

while (firstDigit >= 10) {

firstDigit /= 10;

}

printf("First digit: %d\n", firstDigit);

printf("Last digit: %d\n", lastDigit);

return 0;

}

1. Write a C program to find sum of first and last digit of a number.

#include <stdio.h>

int main() { printf(“Navjot”);

int number, firstDigit, lastDigit, sum;

printf("Enter an integer: ");

scanf("%d", &number);

// To find the last digit, take the remainder when divided by 10

lastDigit = number % 10;

// To find the first digit, repeatedly divide by 10 until only one digit remains

firstDigit = number;

while (firstDigit >= 10) {

firstDigit /= 10;

}

// Calculate the sum of the first and last digits

sum = firstDigit + lastDigit;

printf("First digit: %d\n", firstDigit);

printf("Last digit: %d\n", lastDigit);

printf("Sum of first and last digit: %d\n", sum);

return 0;

}

1. Write a C program to swap first and last digits of a number.

#include <stdio.h>

int main() { printf(“Navjot”);

int number, originalNumber, firstDigit, lastDigit,

swappedNumber = 0;

int placeValue = 1;

printf("Enter an integer: ");

scanf("%d", &number);

originalNumber = number; // Save the original number

// To find the last digit, take the remainder when divided by 10

lastDigit = number % 10;

// To find the first digit, repeatedly divide by 10 until only one digit remains

while (number >= 10) {

number /= 10;

}

firstDigit = number;

// Construct the swapped number

number = originalNumber;

while (number >= 10) {

number /= 10;

placeValue \*= 10;

}

swappedNumber = lastDigit \* placeValue; // Add last digit

swappedNumber += originalNumber % placeValue; // Add middle digits

swappedNumber += firstDigit; // Add first digit

printf("Original number: %d\n", originalNumber);

printf("Number after swapping first and last digits: %d\n", swappedNumber);

return 0;

}

1. Write a C program to calculate sum of digits of a number.

#include <stdio.h>

int main() { printf(“Navjot”);

int number, sum = 0, digit;

printf("Enter an integer: ");

scanf("%d", &number);

// Calculate the sum of digits

while (number != 0) {

digit = number % 10; // Get the last digit

sum += digit; // Add it to the sum

number /= 10; // Remove the last digit

}

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

return 0;

}

1. Write a C program to calculate product of digits of a number.

#include <stdio.h>

int main() { printf(“Navjot”);

int number, product = 1, digit;

printf("Enter an integer: ");

scanf("%d", &number);

// Calculate the product of digits

while (number != 0) {

digit = number % 10; // Get the last digit

product \*= digit; // Multiply it by the product

number /= 10; // Remove the last digit

}

printf("Product of the digits: %d\n", product);

return 0;

}

1. Write a C program to enter a number and print its reverse.

#include <stdio.h>

int main() { printf(“Navjot”);

int number, reversedNumber = 0, originalNumber, remainder;

printf("Enter an integer: ");

scanf("%d", &number);

originalNumber = number; // Save the original number

// Reverse the number

while (number != 0) {

remainder = number % 10; // Get the last digit

reversedNumber = reversedNumber \* 10 + remainder; // Add the digit to the reversed number

number /= 10; // Remove the last digit

}

printf("Original number: %d\n", originalNumber);

printf("Reversed number: %d\n", reversedNumber);

return 0;

}

1. Write a C program to check whether a number is palindrome or not.

#include <stdio.h>

int main() { printf(“Navjot”);

int number, reversedNumber = 0, originalNumber, remainder;

printf("Enter an integer: ");

scanf("%d", &number);

originalNumber = number; // Save the original number

// Reverse the number

while (number != 0) {

remainder = number % 10; // Get the last digit

reversedNumber = reversedNumber \* 10 + remainder; // Add the digit to the reversed number

number /= 10; // Remove the last digit

}

// Check if the original number is the same as its reverse

if (originalNumber == reversedNumber) {

printf("%d is a palindrome.\n", originalNumber);

} else {

printf("%d is not a palindrome.\n", originalNumber);

}

return 0;

}

1. Write a C program to find frequency of each digit in a given integer.

#include <stdio.h>

int main() { printf(“Navjot”);

int number, digit, originalNumber;

int frequency[10] = {0}; // Array to store the frequency of each digit

printf("Enter an integer: ");

scanf("%d", &number);

originalNumber = number; // Save the original number

// Calculate the frequency of each digit

while (number != 0) {

digit = number % 10; // Get the last digit

frequency[digit]++; // Increment the corresponding element in the array

number /= 10; // Remove the last digit

}

printf("Digit frequencies in %d are:\n", originalNumber);

for (int i = 0; i < 10; i++) {

if (frequency[i] > 0) {

printf("Digit %d occurs %d time(s)\n", i, frequency[i]);

}

}

return 0;

}

1. Write a C program to enter a number and print it in words.

#include <stdio.h>

// Function to convert a two-digit number to words

void convertTwoDigitsToWords(int num) {

char\* ones[] = {"", "One", "Two", "Three", "Four", "Five", "Six", "Seven", "Eight", "Nine"};

char\* teens[] = {"Ten", "Eleven", "Twelve", "Thirteen", "Fourteen", "Fifteen", "Sixteen", "Seventeen", "Eighteen", "Nineteen"};

char\* tens[] = {"", "", "Twenty", "Thirty", "Forty", "Fifty", "Sixty", "Seventy", "Eighty", "Ninety"};

if (num < 10) {

printf("%s ", ones[num]);

} else if (num < 20) {

printf("%s ", teens[num - 10]);

} else {

int tensDigit = num / 10;

int onesDigit = num % 10;

printf("%s", tens[tensDigit]);

if (onesDigit != 0) {

printf("-%s", ones[onesDigit]);

}

}

}

int main() { printf(“Navjot”);

int number;

printf("Enter a number: ");

scanf("%d", &number);

if (number < 0) {

printf("Negative ");

number = -number; // Make it positive for processing

}

if (number == 0) {

printf("Zero\n");

return 0;

}

if (number >= 1000) {

printf("Please enter a number less than 1000.\n");

return 0;

}

if (number >= 100) {

printf("%s Hundred ", convertTwoDigitsToWords(number / 100));

number %= 100;

}

if (number >= 10) {

convertTwoDigitsToWords(number);

} else {

printf("%s ", ones[number]);

}

printf("\n");

printf(“Navjot”);

return 0;

}

1. Write a C program to print all ASCII character with their values.

#include <stdio.h>

int main() { printf(“Navjot”);

printf("ASCII Characters and Their Values (0-127):\n");

for (int i = 0; i < 128; i++) {

printf("ASCII value %d corresponds to character '%c'\n", i, (char)i);

}

return 0;

}

1. Write a C program to find power of a number using for loop.

#include <stdio.h>

int main() { printf(“Navjot”);

double base, exponent, result = 1.0;

printf("Enter the base: ");

scanf("%lf", &base);

printf("Enter the exponent: ");

scanf("%lf", &exponent);

if (exponent >= 0) {

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

result \*= base;

}

printf("%.2lf raised to the power %.2lf is %.2lf\n", base, exponent, result);

} else {

printf("Exponent should be a non-negative number for this example.\n");

}

return 0;

}

1. Write a C program to find all factors of a number.

#include <stdio.h>

int main() { printf(“Navjot”);

int number;

printf("Enter a positive integer: ");

scanf("%d", &number);

if (number <= 0) {

printf("Please enter a valid positive integer.\n");

} else {

printf("Factors of %d are: ", number);

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

if (number % i == 0) {

printf("%d ", i);

}

}

printf("\n");

}

return 0;

}

1. Write a C program to calculate factorial of a number.

#include <stdio.h>

int main() { printf(“Navjot”);

int num;

unsigned long long factorial = 1; // Use 'unsigned long long' to handle large factorials

printf("Enter a positive integer: ");

scanf("%d", &num);

if (num < 0) {

printf("Factorial is not defined for negative numbers.\n");

} else {

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

factorial \*= i;

}

printf("Factorial of %d = %llu\n", num, factorial);

}

return 0;

}

1. Write a C program to find HCF (GCD) of two numbers.

#include <stdio.h>

int findGCD(int a, int b) {

if (b == 0) {

return a;

} else {

return findGCD(b, a % b);

}

}

int main() { printf(“Navjot”);

int num1, num2;

printf("Enter two positive integers: ");

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

if (num1 <= 0 || num2 <= 0) {

printf("Please enter positive integers.\n");

} else {

int gcd = findGCD(num1, num2);

printf("The GCD of %d and %d is %d\n", num1, num2, gcd);

}

return 0;

}

1. Write a C program to find LCM of two numbers.

#include <stdio.h>

int findGCD(int a, int b) {

if (b == 0) {

return a;

} else {

return findGCD(b, a % b);

}

}

int findLCM(int a, int b) {

int gcd = findGCD(a, b);

int lcm = (a \* b) / gcd;

return lcm;

}

int main() { printf(“Navjot”);

int num1, num2;

printf("Enter two positive integers: ");

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

if (num1 <= 0 || num2 <= 0) {

printf("Please enter positive integers.\n");

} else {

int lcm = findLCM(num1, num2);

printf("The LCM of %d and %d is %d\n", num1, num2, lcm);

}

return 0;

}

1. Write a C program to check whether a number is Prime number or not.

#include <stdio.h>

#include <stdbool.h>

bool isPrime(int num) {

if (num <= 1) {

return false; // 1 and non-positive numbers are not prime

}

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

if (num % i == 0) {

return false; // If the number is divisible by any integer from 2 to sqrt(num), it's not prime

}

}

return true; // If no divisors are found, the number is prime

}

int main() { printf(“Navjot”);

int num;

printf("Enter a positive integer: ");

scanf("%d", &num);

if (isPrime(num)) {

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

} else {

printf("%d is not a prime number.\n");

}

return 0;

}

1. Write a C program to print all Prime numbers between 1 to n.

#include <stdio.h>

#include <stdbool.h>

bool isPrime(int num) {

if (num <= 1) {

return false; // 1 and non-positive numbers are not prime

}

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

if (num % i == 0) {

return false; // If the number is divisible by any integer from 2 to sqrt(num), it's not prime

}

}

return true; // If no divisors are found, the number is prime

}

int main() { printf(“Navjot”);

int n;

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

scanf("%d", &n);

if (n < 2) {

printf("There are no prime numbers in the range [1, %d]\n", n);

} else {

printf("Prime numbers in the range [1, %d]:\n", n);

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

if (isPrime(i)) {

printf("%d ", i);

}

}

printf("\n");

}

return 0;

}

1. Write a C program to find sum of all prime numbers between 1 to n.

#include <stdio.h>

#include <stdbool.h>

bool isPrime(int num) {

if (num <= 1) {

return false; // 1 and non-positive numbers are not prime

}

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

if (num % i == 0) {

return false; // If the number is divisible by any integer from 2 to sqrt(num), it's not prime

}

}

return true; // If no divisors are found, the number is prime

}

int main() { printf(“Navjot”);

int n;

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

scanf("%d", &n);

if (n < 2) {

printf("There are no prime numbers in the range [1, %d]\n", n);

} else {

int sum = 0;

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

if (isPrime(i)) {

sum += i;

}

}

printf("Sum of prime numbers in the range [1, %d]: %d\n", n, sum);

}

return 0;

}

1. Write a C program to find all prime factors of a number.

#include <stdio.h>

void primeFactors(int num) {

while (num % 2 == 0) {

printf("2 ");

num /= 2;

}

for (int i = 3; i \* i <= num; i += 2) {

while (num % i == 0) {

printf("%d ", i);

num /= i;

}

}

if (num > 2) {

printf("%d ", num);

}

}

int main() { printf(“Navjot”);

int num;

printf("Enter a positive integer: ");

scanf("%d", &num);

if (num <= 1) {

printf("Prime factors are not defined for numbers less than 2.\n");

} else {

printf("Prime factors of %d are: ", num);

primeFactors(num);

printf("\n");

}

return 0;

}

1. Write a C program to check whether a number is Armstrong number or not.

#include <stdio.h>

#include <math.h>

int countDigits(int num) {

int count = 0;

while (num != 0) {

num /= 10;

count++;

}

return count;

}

int isArmstrong(int num) {

int originalNum = num;

int n = countDigits(num);

int sum = 0;

while (num > 0) {

int digit = num % 10;

sum += pow(digit, n);

num /= 10;

}

return (sum == originalNum);

}

int main() { printf(“Navjot”);

int num;

printf("Enter a positive integer: ");

scanf("%d", &num);

if (num < 0) {

printf("Please enter a positive integer.\n");

} else {

if (isArmstrong(num)) {

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

} else {

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

}

}

return 0;

}

1. Write a C program to print all Armstrong numbers between 1 to n.

#include <stdio.h>

#include <math.h>

int countDigits(int num) {

int count = 0;

while (num != 0) {

num /= 10;

count++;

}

return count;

}

int isArmstrong(int num) {

int originalNum = num;

int n = countDigits(num);

int sum = 0;

while (num > 0) {

int digit = num % 10;

sum += pow(digit, n);

num /= 10;

}

return (sum == originalNum);

}

int main() { printf(“Navjot”);

int n;

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

scanf("%d", &n);

if (n < 1) {

printf("There are no Armstrong numbers in the range [1, %d]\n", n);

} else {

printf("Armstrong numbers in the range [1, %d]:\n", n);

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

if (isArmstrong(i)) {

printf("%d ", i);

}

}

printf("\n");

}

return 0;

}

1. Write a C program to check whether a number is Perfect number or not.

#include <stdio.h>

int isPerfect(int num) {

int sum = 1; // Initialize sum with 1, as all numbers are divisible by 1

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

if (num % i == 0) {

if (i \* i != num) {

// Add both divisors (i and num/i) to the sum

sum += i + num / i;

} else {

// Add only one divisor (i) to the sum if the number is a perfect square

sum += i;

}

}

}

return (sum == num);

}

int main() { printf(“Navjot”);

int num;

printf("Enter a positive integer: ");

scanf("%d", &num);

if (num <= 0) {

printf("Please enter a positive integer.\n");

} else {

if (isPerfect(num)) {

printf("%d is a perfect number.\n", num);

} else {

printf("%d is not a perfect number.\n", num);

}

}

return 0;

}

1. Write a C program to print all Perfect numbers between 1 to n.

#include <stdio.h>

int isPerfect(int num) {

int sum = 1; // Initialize sum with 1, as all numbers are divisible by 1

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

if (num % i == 0) {

if (i \* i != num) {

// Add both divisors (i and num/i) to the sum

sum += i + num / i;

} else {

// Add only one divisor (i) to the sum if the number is a perfect square

sum += i;

}

}

}

return (sum == num);

}

int main() { printf(“Navjot”);

int n;

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

scanf("%d", &n);

if (n < 1) {

printf("There are no perfect numbers in the range [1, %d]\n", n);

} else {

printf("Perfect numbers in the range [1, %d]:\n", n);

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

if (isPerfect(i)) {

printf("%d ", i);

}

}

printf("\n");

}

return 0;

}

1. Write a C program to check whether a number is Strong number or not.

#include <stdio.h>

int factorial(int n) {

if (n == 0) {

return 1;

} else {

int fact = 1;

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

fact \*= i;

}

return fact;

}

}

int isStrongNumber(int num) {

int originalNum = num;

int sum = 0;

while (num > 0) {

int digit = num % 10;

sum += factorial(digit);

num /= 10;

}

return (sum == originalNum);

}

int main() { printf(“Navjot”);

int num;

printf("Enter a positive integer: ");

scanf("%d", &num);

if (num < 0) {

printf("Please enter a positive integer.\n");

} else {

if (isStrongNumber(num)) {

printf("%d is a strong number.\n", num);

} else {

printf("%d is not a strong number.\n", num);

}

}

return 0;

}

1. Write a C program to print all Strong numbers between 1 to n.

#include <stdio.h>

int factorial(int n) {

if (n == 0) {

return 1;

} else {

int fact = 1;

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

fact \*= i;

}

return fact;

}

}

int isStrongNumber(int num) {

int originalNum = num;

int sum = 0;

while (num > 0) {

int digit = num % 10;

sum += factorial(digit);

num /= 10;

}

return (sum == originalNum);

}

int main() { printf(“Navjot”);

int n;

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

scanf("%d", &n);

if (n < 1) {

printf("There are no strong numbers in the range [1, %d]\n", n);

} else {

printf("Strong numbers in the range [1, %d]:\n", n);

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

if (isStrongNumber(i)) {

printf("%d ", i);

}

}

printf("\n");

}

return 0;

}

1. Write a C program to print Fibonacci series up to n terms.

#include <stdio.h>

int main() { printf(“Navjot”);

int n, first = 0, second = 1, next;

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

scanf("%d", &n);

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

if (n >= 1) {

printf("%d", first);

}

if (n >= 2) {

printf(", %d", second);

}

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

next = first + second;

printf(", %d", next);

first = second;

second = next;

}

printf("\n");

return 0;

}

1. Write a C program to find one's complement of a binary number.

#include <stdio.h>

#include <string.h>

int main() { printf(“Navjot”);

char binary[32];

int i;

printf("Enter a binary number: ");

scanf("%s", binary);

int length = strlen(binary);

for (i = 0; i < length; i++) {

if (binary[i] == '0') {

binary[i] = '1';

} else if (binary[i] == '1') {

binary[i] = '0';

} else {

printf("Invalid binary number: It should consist of only 0s and 1s.\n");

return 1;

}

}

printf("One's complement of the binary number is: %s\n", binary);

return 0;

}

1. Write a C program to find two's complement of a binary number.

#include <stdio.h>

#include <string.h>

void findOnesComplement(char binary[]) {

int length = strlen(binary);

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

if (binary[i] == '0') {

binary[i] = '1';

} else if (binary[i] == '1') {

binary[i] = '0';

} else {

printf("Invalid binary number: It should consist of only 0s and 1s.\n");

return;

}

}

}

void addOne(char binary[]) {

int length = strlen(binary);

int carry = 1;

for (int i = length - 1; i >= 0; i--) {

if (binary[i] == '0' && carry == 1) {

binary[i] = '1';

carry = 0;

} else if (binary[i] == '1' && carry == 1) {

binary[i] = '0';

carry = 1;

}

}

}

int main() { printf(“Navjot”);

char binary[32];

printf("Enter a binary number: ");

scanf("%s", binary);

int length = strlen(binary);

findOnesComplement(binary);

addOne(binary);

printf("Two's complement of the binary number is: %s\n", binary);

return 0;

}

1. Write a C program to convert Binary to Octal number system.

#include <stdio.h>

#include <math.h>

int binaryToOctal(int binary) {

int octal = 0, decimal = 0, i = 0;

// Convert binary to decimal

while (binary != 0) {

int remainder = binary % 10;

decimal += remainder \* pow(2, i);

binary /= 10;

i++;

}

i = 1;

// Convert decimal to octal

while (decimal != 0) {

int remainder = decimal % 8;

octal += remainder \* i;

decimal /= 8;

i \*= 10;

}

return octal;

}

int main() { printf(“Navjot”);

int binary;

printf("Enter a binary number: ");

scanf("%d", &binary);

if (binary < 0) {

printf("Please enter a non-negative binary number.\n");

} else {

int octal = binaryToOctal(binary);

printf("Octal equivalent: %d\n", octal);

}

return 0;

}

1. Write a C program to convert Binary to Decimal number system.

#include <stdio.h>

int binaryToDecimal(int binary) {

int decimal = 0, base = 1;

while (binary > 0) {

int lastDigit = binary % 10;

decimal += lastDigit \* base;

binary /= 10;

base \*= 2;

}

return decimal;

}

int main() { printf(“Navjot”);

int binary;

printf("Enter a binary number: ");

scanf("%d", &binary);

if (binary < 0) {

printf("Please enter a non-negative binary number.\n");

} else {

int decimal = binaryToDecimal(binary);

printf("Decimal equivalent: %d\n", decimal);

}

return 0;

}

1. Write a C program to convert Binary to Hexadecimal number system

#include <stdio.h>

#include <string.h>

char\* binaryToHexadecimal(char binary[]) {

// Pad the binary number with leading zeros to make its length a multiple of 4

int len = strlen(binary);

int pad\_len = len % 4 == 0 ? 0 : 4 - (len % 4);

char padded\_binary[32];

for (int i = 0; i < pad\_len; i++) {

padded\_binary[i] = '0';

}

strcpy(padded\_binary + pad\_len, binary);

int hex\_len = (len + pad\_len) / 4;

char hexadecimal[8];

char hex\_digits[] = "0123456789ABCDEF";

for (int i = 0; i < hex\_len; i++) {

int decimal = 0;

for (int j = 0; j < 4; j++) {

decimal = decimal \* 2 + (padded\_binary[i \* 4 + j] - '0');

}

hexadecimal[i] = hex\_digits[decimal];

}

hexadecimal[hex\_len] = '\0';

return strdup(hexadecimal);

}

int main() { printf(“Navjot”);

char binary[32];

printf("Enter a binary number: ");

scanf("%s", binary);

if (strspn(binary, "01") != strlen(binary)) {

printf("Invalid binary number: It should consist of only 0s and 1s.\n");

} else {

char\* hexadecimal = binaryToHexadecimal(binary);

printf("Hexadecimal equivalent: %s\n", hexadecimal);

free(hexadecimal);

}

return 0;

}

1. Write a C program to convert Octal to Binary number system.

#include <stdio.h>

#include <string.h>

char\* octalToBinary(char octal[]) {

int octal\_len = strlen(octal);

char binary[32];

char binary\_digits[8][4] = {"000", "001", "010", "011", "100", "101", "110", "111"};

strcpy(binary, "");

for (int i = 0; i < octal\_len; i++) {

int octal\_digit = octal[i] - '0';

if (octal\_digit >= 0 && octal\_digit <= 7) {

strcat(binary, binary\_digits[octal\_digit]);

} else {

printf("Invalid octal number: It should consist of digits 0-7.\n");

return NULL;

}

}

return strdup(binary);

}

int main() { printf(“Navjot”);

char octal[32];

printf("Enter an octal number: ");

scanf("%s", octal);

char\* binary = octalToBinary(octal);

if (binary != NULL) {

printf("Binary equivalent: %s\n", binary);

free(binary);

}

return 0;

}

1. Write a C program to convert Octal to Decimal number system.

#include <stdio.h>

#include <string.h>

int octalToDecimal(char octal[]) {

int octal\_len = strlen(octal);

int decimal = 0;

for (int i = 0; i < octal\_len; i++) {

int octal\_digit = octal[i] - '0';

if (octal\_digit >= 0 && octal\_digit <= 7) {

decimal = (decimal \* 8) + octal\_digit;

} else {

printf("Invalid octal number: It should consist of digits 0-7.\n");

return -1;

}

}

return decimal;

}

int main() { printf(“Navjot”);

char octal[32];

printf("Enter an octal number: ");

scanf("%s", octal);

int decimal = octalToDecimal(octal);

if (decimal != -1) {

printf("Decimal equivalent: %d\n", decimal);

}

return 0;

}

1. Write a C program to convert Octal to Hexadecimal number system.

#include <stdio.h>

#include <string.h>

char\* octalToHexadecimal(char octal[]) {

char binary[32], hexadecimal[8];

char\* binaryToHexadecimal(char binary[]) {

int binary\_len = strlen(binary);

char hexadecimal[8];

char binary\_digits[16][5] = {

"0000", "0001", "0010", "0011",

"0100", "0101", "0110", "0111",

"1000", "1001", "1010", "1011",

"1100", "1101", "1110", "1111"

};

strcpy(hexadecimal, "");

for (int i = 0; i < binary\_len; i += 4) {

char binary\_digit[5];

strncpy(binary\_digit, binary + i, 4);

binary\_digit[4] = '\0';

for (int j = 0; j < 16; j++) {

if (strcmp(binary\_digit, binary\_digits[j]) == 0) {

if (j < 10) {

char hex\_digit[2];

snprintf(hex\_digit, sizeof(hex\_digit), "%d", j);

strcat(hexadecimal, hex\_digit);

} else {

char hex\_digit[2];

snprintf(hex\_digit, sizeof(hex\_digit), "%c", 'A' + (j - 10));

strcat(hexadecimal, hex\_digit);

}

break;

}

}

}

return strdup(hexadecimal);

}

for (int i = 0; i < 32; i++) {

binary[i] = '\0';

}

for (int i = 0; i < strlen(octal); i++) {

int octal\_digit = octal[i] - '0';

if (octal\_digit >= 0 && octal\_digit <= 7) {

strcat(binary, binaryToHexadecimal(octal\_digit));

} else {

printf("Invalid octal number: It should consist of digits 0-7.\n");

return NULL;

}

}

return strdup(hexadecimal);

}

int main() { printf(“Navjot”);

char octal[32];

printf("Enter an octal number: ");

scanf("%s", octal);

char\* hexadecimal = octalToHexadecimal(octal);

if (hexadecimal != NULL) {

printf("Hexadecimal equivalent: %s\n", hexadecimal);

free(hexadecimal);

}

return 0;

}

1. Write a C program to convert Decimal to Binary number system.

#include <stdio.h>

void decimalToBinary(int decimal) {

int binary[32];

int i = 0;

if (decimal == 0) {

printf("Binary equivalent: 0\n");

return;

}

while (decimal > 0) {

binary[i] = decimal % 2;

decimal /= 2;

i++;

}

printf("Binary equivalent: ");

for (int j = i - 1; j >= 0; j--) {

printf("%d", binary[j]);

}

printf("\n");

}

int main() { printf(“Navjot”);

int decimal;

printf("Enter a decimal number: ");

scanf("%d", &decimal);

if (decimal < 0) {

printf("Please enter a non-negative decimal number.\n");

} else {

decimalToBinary(decimal);

}

return 0;

}

1. Write a C program to convert Decimal to Octal number system.

#include <stdio.h>

void decimalToOctal(int decimal) {

int octal[32];

int i = 0;

if (decimal == 0) {

printf("Octal equivalent: 0\n");

return;

}

while (decimal > 0) {

octal[i] = decimal % 8;

decimal /= 8;

i++;

}

printf("Octal equivalent: ");

for (int j = i - 1; j >= 0; j--) {

printf("%d", octal[j]);

}

printf("\n");

}

int main() { printf(“Navjot”);

int decimal;

printf("Enter a decimal number: ");

scanf("%d", &decimal);

if (decimal < 0) {

printf("Please enter a non-negative decimal number.\n");

} else {

decimalToOctal(decimal);

}

return 0;

}

1. Write a C program to convert Decimal to Hexadecimal number system.

#include <stdio.h>

void decimalToHexadecimal(int decimal) {

char hexadecimal[32];

int i = 0;

if (decimal == 0) {

printf("Hexadecimal equivalent: 0\n");

return;

}

while (decimal > 0) {

int remainder = decimal % 16;

if (remainder < 10) {

hexadecimal[i] = remainder + '0';

} else {

hexadecimal[i] = remainder - 10 + 'A';

}

decimal /= 16;

i++;

}

printf("Hexadecimal equivalent: ");

for (int j = i - 1; j >= 0; j--) {

printf("%c", hexadecimal[j]);

}

printf("\n");

}

int main() { printf(“Navjot”);

int decimal;

printf("Enter a decimal number: ");

scanf("%d", &decimal);

if (decimal < 0) {

printf("Please enter a non-negative decimal number.\n");

} else {

decimalToHexadecimal(decimal);

}

return 0;

}

1. Write a C program to convert Hexadecimal to Binary number system.

#include <stdio.h>

#include <string.h>

int hexToDecimal(char hexChar) {

if (hexChar >= '0' && hexChar <= '9') {

return hexChar - '0';

} else if (hexChar >= 'A' && hexChar <= 'F') {

return hexChar - 'A' + 10;

} else if (hexChar >= 'a' && hexChar <= 'f') {

return hexChar - 'a' + 10;

} else {

printf("Invalid hexadecimal digit: %c\n", hexChar);

return -1; // Return -1 for invalid input

}

}

void hexadecimalToBinary(char hex[]) {

int len = strlen(hex);

char binary[32];

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

int decimal = hexToDecimal(hex[i]);

if (decimal == -1) {

return; // Exit if there's an invalid input

}

// Convert the decimal to a 4-digit binary representation

for (int j = 3; j >= 0; j--) {

binary[i \* 4 + j] = (decimal % 2) + '0';

decimal /= 2;

}

}

binary[len \* 4] = '\0';

printf("Binary equivalent: %s\n", binary);

}

int main() { printf(“Navjot”);

char hex[32];

printf("Enter a hexadecimal number: ");

scanf("%s", hex);

hexadecimalToBinary(hex);

return 0;

}

1. Write a C program to convert Hexadecimal to Octal number system.

#include <stdio.h>

#include <string.h>

int hexToDecimal(char hexChar) {

if (hexChar >= '0' && hexChar <= '9') {

return hexChar - '0';

} else if (hexChar >= 'A' && hexChar <= 'F') {

return hexChar - 'A' + 10;

} else if (hexChar >= 'a' && hexChar <= 'f') {

return hexChar - 'a' + 10;

} else {

printf("Invalid hexadecimal digit: %c\n", hexChar);

return -1; // Return -1 for invalid input

}

}

void decimalToOctal(int decimal) {

int octal[32];

int i = 0;

if (decimal == 0) {

printf("Octal equivalent: 0\n");

return;

}

while (decimal > 0) {

octal[i] = decimal % 8;

decimal /= 8;

i++;

}

printf("Octal equivalent: ");

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

printf("%d", octal[j]);

}

printf("\n");

}

void hexToOctal(char hex[]) {

int decimal = 0;

int len = strlen(hex);

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

int hexDigit = hexToDecimal(hex[i]);

if (hexDigit == -1) {

return; // Exit if there's an invalid input

}

decimal = decimal \* 16 + hexDigit;

}

decimalToOctal(decimal);

}

int main() { printf(“Navjot”);

char hex[32];

printf("Enter a hexadecimal number: ");

scanf("%s", hex);

hexToOctal(hex);

return 0;

}

1. Write a C program to convert Hexadecimal to Decimal number system.

#include <stdio.h>

#include <string.h>

int hexToDecimal(char hexChar) {

if (hexChar >= '0' && hexChar <= '9') {

return hexChar - '0';

} else if (hexChar >= 'A' && hexChar <= 'F') {

return hexChar - 'A' + 10;

} else if (hexChar >= 'a' && hexChar <= 'f') {

return hexChar - 'a' + 10;

} else {

printf("Invalid hexadecimal digit: %c\n", hexChar);

return -1; // Return -1 for invalid input

}

}

int hexadecimalToDecimal(char hex[]) {

int decimal = 0;

int len = strlen(hex);

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

int hexDigit = hexToDecimal(hex[i]);

if (hexDigit == -1) {

return -1; // Exit if there's an invalid input

}

decimal = decimal \* 16 + hexDigit;

}

return decimal;

}

int main() { printf(“Navjot”);

char hex[32];

printf("Enter a hexadecimal number: ");

scanf("%s", hex);

int decimal = hexadecimalToDecimal(hex);

if (decimal != -1) {

printf("Decimal equivalent: %d\n", decimal);

}

return 0;

}

1. Star pattern programs - Write a C program to print the given star patterns.

A). #include <stdio.h>

int main() { printf(“Navjot”);

int rows, i, j, space;

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

scanf("%d", &rows);

for (i = 1; i <= rows; i++) {

// Print spaces before the stars

for (space = 1; space <= rows - i; space++) {

printf(" ");

}

// Print stars

for (j = 1; j <= 2 \* i - 1; j++) {

printf("\*");

}

printf("\n");

}

return 0;

}

B). #include <stdio.h>

int main() { printf(“Navjot”);

int rows, i, j, space;

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

scanf("%d", &rows);

for (i = 1; i <= rows; i++) {

// Print spaces before the stars

for (space = 1; space <= rows - i; space++) {

printf(" ");

}

// Print stars in the first and last rows, and spaces in between

if (i == 1 || i == rows) {

for (j = 1; j <= 2 \* i - 1; j++) {

printf("\*");

}

} else { // Print stars with spaces in between

printf("\*");

for (j = 1; j <= 2 \* i - 3; j++) {

printf(" ");

}

printf("\*");

}

printf("\n");

}

return 0;

}

C). #include <stdio.h>

int main() { printf(“Navjot”);

int rows, i, j, space;

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

scanf("%d", &rows);

for (i = rows; i >= 1; i--) {

// Print spaces before the stars

for (space = 1; space <= rows - i; space++) {

printf(" ");

}

// Print stars

for (j = 1; j <= 2 \* i - 1; j++) {

printf("\*");

}

printf("\n");

}

return 0;

}

D). #include <stdio.h>

int main() { printf(“Navjot”);

int rows, i, j, space;

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

scanf("%d", &rows);

for (i = 1; i <= rows; i++) {

// Print spaces before the stars

for (space = 1; space < i; space++) {

printf(" ");

}

// Print stars and spaces in the first and last rows

if (i == 1 || i == rows) {

for (j = 1; j <= 2 \* (rows - i) + 1; j++) {

printf("\*");

}

} else { // Print stars with spaces in between

printf("\*");

for (j = 1; j <= 2 \* (rows - i - 1) + 1; j++) {

printf(" ");

}

printf("\*");

}

printf("\n");

}

return 0;

}

E). #include <stdio.h>

int main() { printf(“Navjot”);

int rows, i, j;

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

scanf("%d", &rows);

for (i = 1; i <= rows; i++) {

// Print stars in the first half of the diamond

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

printf("\*");

}

printf("\n");

}

for (i = rows - 1; i >= 1; i--) {

// Print stars in the second half of the diamond

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

printf("\*");

}

printf("\n");

}

return 0;

}

F). #include <stdio.h>

int main() { printf(“Navjot”);

int rows, i, j, space;

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

scanf("%d", &rows);

for (i = 1; i <= rows; i++) {

// Print spaces before the stars

for (space = 1; space <= rows - i; space++) {

printf(" ");

}

// Print stars

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

printf("\*");

}

printf("\n");

}

for (i = rows - 1; i >= 1; i--) {

// Print spaces before the stars

for (space = 1; space <= rows - i; space++) {

printf(" ");

}

// Print stars

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

printf("\*");

}

printf("\n");

}

return 0;

}

1. Number pattern programs - Write a C program to print the given number patterns.

A). #include <stdio.h>

int main() { printf(“Navjot”);

int rows, cols;

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

scanf("%d", &rows);

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

scanf("%d", &cols);

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

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

printf("1");

}

printf("\n");

}

return 0;

}

B). #include <stdio.h>

int main() { printf(“Navjot”);

int rows, i, j;

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

scanf("%d", &rows);

for (i = 1; i <= rows; i++) {

// Print '1' in odd rows and '0' in even rows

for (j = 1; j <= rows; j++) {

if (i % 2 == 1) {

printf("1");

} else {

printf("0");

}

}

printf("\n");

}

return 0;

}

C). #include <stdio.h>

int main() { printf(“Navjot”);

int rows, i, j;

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

scanf("%d", &rows);

for (i = 1; i <= rows; i++) {

// Print '0' and '1' alternatively in each column

for (j = 1; j <= rows; j++) {

if (j % 2 == 0) {

printf("1");

} else {

printf("0");

}

}

printf("\n");

}

return 0;

}

D). #include <stdio.h>

int main() { printf(“Navjot”);

int rows, i, j;

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

scanf("%d", &rows);

for (i = 1; i <= rows; i++) {

// Print '1' at the beginning and end, '0' in between

for (j = 1; j <= rows; j++) {

if (j == 1 || j == rows || i == 1 || i == rows) {

printf("1");

} else {

printf("0");

}

}

printf("\n");

}

return 0;

}

E). #include <stdio.h>

int main() { printf(“Navjot”);

int rows, i, j;

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

scanf("%d", &rows);

for (i = 1; i <= rows; i++) {

// Print '1' at the middle, '0' elsewhere

for (j = 1; j <= rows; j++) {

if (i == (rows / 2) + 1 || j == (rows / 2) + 1) {

printf("1");

} else {

printf("0");

}

}

printf("\n");

}

return 0;

}

F). #include <stdio.h>

int main() { printf(“Navjot”);

int rows, i, j;

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

scanf("%d", &rows);

for (i = 1; i <= rows; i++) {

// Print alternating '1' and '0'

for (j = 1; j <= rows; j++) {

if ((i + j) % 2 == 0) {

printf("1");

} else {

printf("0");

}

}

printf("\n");

}

return 0;

}

1. Write a C program to find maximum between two numbers.

#include <stdio.h>

int main() { printf(“Navjot”);

int num1, num2;

// Input two numbers

printf("Enter the first number: ");

scanf("%d", &num1);

printf("Enter the second number: ");

scanf("%d", &num2);

// Compare and find the maximum

if (num1 > num2) {

printf("The maximum number is: %d\n", num1);

} else if (num2 > num1) {

printf("The maximum number is: %d\n", num2);

} else {

printf("Both numbers are equal.\n");

}

return 0;

}

1. Write a C program to find maximum between three numbers.

#include <stdio.h>

int main() { printf(“Navjot”);

int num1, num2, num3, max;

// Input three numbers

printf("Enter the first number: ");

scanf("%d", &num1);

printf("Enter the second number: ");

scanf("%d", &num2);

printf("Enter the third number: ");

scanf("%d", &num3);

// Assume the first number is the maximum

max = num1;

// Compare with the second number

if (num2 > max) {

max = num2;

}

// Compare with the third number

if (num3 > max) {

max = num3;

}

printf("The maximum number is: %d\n", max);

return 0;

}

1. Write a C program to check whether a number is negative, positive or zero.

#include <stdio.h>

int main() { printf(“Navjot”);

int number;

// Input a number

printf("Enter a number: ");

scanf("%d", &number);

if (number > 0) {

printf("The number is positive.\n");

} else if (number < 0) {

printf("The number is negative.\n");

} else {

printf("The number is zero.\n");

}

return 0;

}

1. Write a C program to check whether a number is divisible by 5 and 11 or not.

#include <stdio.h>

int main() { printf(“Navjot”);

int number;

// Input a number

printf("Enter a number: ");

scanf("%d", &number);

// Check if the number is divisible by 5 and 11

if (number % 5 == 0 && number % 11 == 0) {

printf("The number is divisible by both 5 and 11.\n");

} else {

printf("The number is not divisible by both 5 and 11.\n");

}

return 0;

}

1. Write a C program to check whether a number is even or odd.

#include <stdio.h>

int main() { printf(“Navjot”);

int number;

// Input a number

printf("Enter a number: ");

scanf("%d", &number);

// Check if the number is even or odd

if (number % 2 == 0) {

printf("The number is even.\n");

} else {

printf("The number is odd.\n");

}

return 0;

}

1. Write a C program to check whether a year is leap year or not.

#include <stdio.h>

int main() { printf(“Navjot”);

int year;

// Input a year

printf("Enter a year: ");

scanf("%d", &year);

// Check if the year is a leap year

if ((year % 4 == 0 && year % 100 != 0) || (year % 400 == 0)) {

printf("%d is a leap year.\n", year);

} else {

printf("%d is not a leap year.\n", year);

}

return 0;

}

1. Write a C program to check whether a year is leap year or not.

#include <stdio.h>

int main() { printf(“Navjot”);

int year;

// Input a year

printf("Enter a year: ");

scanf("%d", &year);

// Check if the year is a leap year

if ((year % 4 == 0 && year % 100 != 0) || (year % 400 == 0)) {

printf("%d is a leap year.\n", year);

} else {

printf("%d is not a leap year.\n", year);

}

return 0;

}

82. Write a C program to check whether a character is alphabet or not.

#include <stdio.h>

#include <ctype.h>

int main() { printf(“Navjot”);

char character;

// Input a character

printf("Enter a character: ");

scanf(" %c", &character);

// Check if the character is an alphabet

if (isalpha(character)) {

printf("%c is an alphabet.\n", character);

} else {

printf("%c is not an alphabet.\n", character);

}

return 0;

}

83. Write a C program to input any alphabet and check whether it is vowel or consonant.

#include <stdio.h>

#include <ctype.h>

int main() { printf(“Navjot”);

char character;

// Input a character

printf("Enter a character: ");

scanf(" %c", &character);

// Convert the character to uppercase for case-insensitive comparison

character = toupper(character);

// Check if the character is a vowel or consonant

if (isalpha(character)) {

if (character == 'A' || character == 'E' || character == 'I' || character == 'O' || character == 'U') {

printf("%c is a vowel.\n", character);

} else {

printf("%c is a consonant.\n", character);

}

} else {

printf("%c is not a valid alphabet character.\n", character);

}

return 0;

}

84.Write a C program to input any alphabet and check whether it is vowel or consonant.

#include <stdio.h>

#include <ctype.h>

int main() { printf(“Navjot”);

char character;

// Input a character

printf("Enter an alphabet: ");

scanf(" %c", &character);

// Convert the character to uppercase for case-insensitive comparison

character = toupper(character);

// Check if the character is a vowel or consonant

if ((character >= 'A' && character <= 'Z')) {

if (character == 'A' || character == 'E' || character == 'I' || character == 'O' || character == 'U') {

printf("%c is a vowel.\n", character);

} else {

printf("%c is a consonant.\n", character);

}

} else {

printf("%c is not a valid alphabet character.\n", character);

}

return 0;

}

85. Write a C program to input any character and check whether it is alphabet, digit or special character.

#include <stdio.h>

#include <ctype.h>

int main() { printf(“Navjot”);

char character;

// Input a character

printf("Enter a character: ");

scanf(" %c", &character);

// Check if the character is an alphabet, digit, or special character

if (isalpha(character)) {

printf("%c is an alphabet.\n", character);

} else if (isdigit(character)) {

printf("%c is a digit.\n", character);

} else {

printf("%c is a special character.\n", character);

}

return 0;

}

86. Write a C program to check whether a character is uppercase or lowercase alphabet.

#include <stdio.h>

#include <ctype.h>

int main() { printf(“Navjot”);

char character;

// Input a character

printf("Enter a character: ");

scanf(" %c", &character);

// Check if the character is uppercase or lowercase alphabet

if (isalpha(character)) {

if (isupper(character)) {

printf("%c is an uppercase alphabet.\n", character);

} else if (islower(character)) {

printf("%c is a lowercase alphabet.\n", character);

}

} else {

printf("%c is not an alphabet character.\n", character);

}

return 0;

}

87. Write a C program to input week number and print week day.

#include <stdio.h>

int main() { printf(“Navjot”);

int weekNumber;

// Input the week number

printf("Enter a week number (1-7): ");

scanf("%d", &weekNumber);

// Check and print the corresponding day of the week

switch (weekNumber) {

case 1:

printf("Day of the week: Sunday\n");

break;

case 2:

printf("Day of the week: Monday\n");

break;

case 3:

printf("Day of the week: Tuesday\n");

break;

case 4:

printf("Day of the week: Wednesday\n");

break;

case 5:

printf("Day of the week: Thursday\n");

break;

case 6:

printf("Day of the week: Friday\n");

break;

case 7:

printf("Day of the week: Saturday\n");

break;

default:

printf("Invalid week number. Please enter a number from 1 to 7.\n");

break;

}

return 0;

}

88. Write a C program to input month number and print number of days in that month.

#include <stdio.h>

int main() { printf(“Navjot”);

int month;

// Input the month number

printf("Enter a month number (1-12): ");

scanf("%d", &month);

// Check and print the number of days in the month

switch (month) {

case 1: case 3: case 5: case 7: case 8: case 10: case 12:

printf("Number of days in the month: 31 days\n");

break;

case 4: case 6: case 9: case 11:

printf("Number of days in the month: 30 days\n");

break;

case 2:

printf("Number of days in the month: 28 or 29 days (leap year)\n");

break;

default:

printf("Invalid month number. Please enter a number from 1 to 12.\n");

break;

}

return 0;

}

89. Write a C program to count total number of notes in given amount.

#include <stdio.h>

int main() { printf(“Navjot”);

int amount;

int notes[] = {2000, 500, 200, 100, 50, 20, 10, 5, 2, 1};

int count[10] = {0};

// Input the amount

printf("Enter the amount in INR: ");

scanf("%d", &amount);

if (amount <= 0) {

printf("Invalid amount. Please enter a positive amount.\n");

} else {

int remaining = amount;

for (int i = 0; i < 10; i++) {

count[i] = remaining / notes[i];

remaining = remaining % notes[i];

}

printf("Number of currency notes for %d INR:\n", amount);

for (int i = 0; i < 10; i++) {

if (count[i] > 0) {

printf("%d x %d INR\n", count[i], notes[i]);

}

}

}

return 0;

}

90. Write a C program to input month number and print number of days in that month.

#include <stdio.h>

int main() { printf(“Navjot”);

int month;

// Input the month number

printf("Enter a month number (1-12): ");

scanf("%d", &month);

if (month < 1 || month > 12) {

printf("Invalid month number. Please enter a number from 1 to 12.\n");

} else {

int days;

if (month == 4 || month == 6 || month == 9 || month == 11) {

days = 30;

} else if (month == 2) {

days = 28; // Assuming a non-leap year

} else {

days = 31;

}

printf("Number of days in month %d: %d days\n", month, days);

}

return 0;

}

91. Write a C program to count total number of notes in given amount.

#include <stdio.h>

int main() { printf(“Navjot”);

int amount;

int notes[] = {2000, 500, 200, 100, 50, 20, 10, 5, 2, 1};

int count[10] = {0};

// Input the amount

printf("Enter the amount in INR: ");

scanf("%d", &amount);

if (amount <= 0) {

printf("Invalid amount. Please enter a positive amount.\n");

} else {

int remaining = amount;

for (int i = 0; i < 10; i++) {

count[i] = remaining / notes[i];

remaining = remaining % notes[i];

}

printf("Number of currency notes for %d INR:\n", amount);

for (int i = 0; i < 10; i++) {

if (count[i] > 0) {

printf("%d x %d INR\n", count[i], notes[i]);

}

}

}

return 0;

}

92. Write a C program to input angles of a triangle and check whether triangle is valid or not.

#include <stdio.h>

int main() { printf(“Navjot”);

int angle1, angle2, angle3;

// Input the three angles of the triangle

printf("Enter the first angle: ");

scanf("%d", &angle1);

printf("Enter the second angle: ");

scanf("%d", &angle2);

printf("Enter the third angle: ");

scanf("%d", &angle3);

// Check if the triangle is valid

if (angle1 + angle2 + angle3 == 180 && angle1 > 0 && angle2 > 0 && angle3 > 0) {

printf("The given angles form a valid triangle.\n");

} else {

printf("The given angles do not form a valid triangle.\n");

}

return 0;

}

93. Write a C program to input all sides of a triangle and check whether triangle is valid or not.

#include <stdio.h>

int main() { printf(“Navjot”);

double side1, side2, side3;

// Input the lengths of the three sides

printf("Enter the length of the first side: ");

scanf("%lf", &side1);

printf("Enter the length of the second side: ");

scanf("%lf", &side2);

printf("Enter the length of the third side: ");

scanf("%lf", &side3);

// Check if the triangle is valid

if (side1 + side2 > side3 && side1 + side3 > side2 && side2 + side3 > side1) {

printf("The given sides form a valid triangle.\n");

} else {

printf("The given sides do not form a valid triangle.\n");

}

return 0;

}

94. Write a C program to check whether the triangle is equilateral, isosceles or scalene triangle.

#include <stdio.h>

int main() { printf(“Navjot”);

double side1, side2, side3;

// Input the lengths of the three sides

printf("Enter the length of the first side: ");

scanf("%lf", &side1);

printf("Enter the length of the second side: ");

scanf("%lf", &side2);

printf("Enter the length of the third side: ");

scanf("%lf", &side3);

// Check if the triangle is equilateral, isosceles, or scalene

if (side1 == side2 && side2 == side3) {

printf("It's an equilateral triangle.\n");

} else if (side1 == side2 || side2 == side3 || side1 == side3) {

printf("It's an isosceles triangle.\n");

} else {

printf("It's a scalene triangle.\n");

}

return 0;

}

95. Write a C program to find all roots of a quadratic equation.

#include <stdio.h>

#include <math.h>

int main() { printf(“Navjot”);

double a, b, c;

double discriminant, root1, root2;

// Input coefficients

printf("Enter the coefficients of the quadratic equation (a, b, c): ");

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

// Calculate the discriminant

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

if (discriminant > 0) {

// Two real and distinct roots

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

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

printf("Root 1 = %.2lf and Root 2 = %.2lf\n", root1, root2);

} else if (discriminant == 0) {

// One real root (double root)

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

printf("Root 1 = Root 2 = %.2lf\n", root1);

} else {

// Complex roots

double realPart = -b / (2 \* a);

double imaginaryPart = sqrt(-discriminant) / (2 \* a);

printf("Root 1 = %.2lf + %.2lfi and Root 2 = %.2lf - %.2lfi\n", realPart, imaginaryPart, realPart, imaginaryPart);

}

return 0;

}

96. Write a C program to calculate profit or loss.

#include <stdio.h>

int main() { printf(“Navjot”);

float costPrice, sellingPrice, profitOrLoss;

// Input cost price and selling price

printf("Enter the cost price: ");

scanf("%f", &costPrice);

printf("Enter the selling price: ");

scanf("%f", &sellingPrice);

// Calculate profit or loss

profitOrLoss = sellingPrice - costPrice;

if (profitOrLoss > 0) {

printf("You made a profit of %.2f\n", profitOrLoss);

} else if (profitOrLoss < 0) {

printf("You incurred a loss of %.2f\n", -profitOrLoss);

} else {

printf("No profit, no loss.\n");

}

return 0;

}

97. Write a C program to input marks of five subjects Physics, Chemistry, Biology, Mathematics and Computer. Calculate percentage and grade.

#include <stdio.h>

int main() { printf(“Navjot”);

float physics, chemistry, biology, mathematics, computer;

float totalMarks, percentage;

char grade;

// Input marks in each subject

printf("Enter marks in Physics: ");

scanf("%f", &physics);

printf("Enter marks in Chemistry: ");

scanf("%f", &chemistry);

printf("Enter marks in Biology: ");

scanf("%f", &biology);

printf("Enter marks in Mathematics: ");

scanf("%f", &mathematics);

printf("Enter marks in Computer: ");

scanf("%f", &computer);

// Calculate total marks and percentage

totalMarks = physics + chemistry + biology + mathematics + computer;

percentage = (totalMarks / 500.0) \* 100;

// Determine the grade based on percentage

if (percentage >= 90) {

grade = 'A';

} else if (percentage >= 80) {

grade = 'B';

} else if (percentage >= 70) {

grade = 'C';

} else if (percentage >= 60) {

grade = 'D';

} else if (percentage >= 40) {

grade = 'E';

} else {

grade = 'F';

}

// Display the result

printf("Total Marks: %.2f\n", totalMarks);

printf("Percentage: %.2f%%\n", percentage);

printf("Grade: %c\n", grade);

return 0;

}

98. Write a C program to input basic salary of an employee and calculate its Gross salary according to following: Basic Salary <= 10000 : HRA = 20%, DA = 80% Basic Salary <= 20000 : HRA = 25%, DA = 90% Basic Salary > 20000 : HRA = 30%, DA = 95%

#include <stdio.h>

int main() { printf(“Navjot”);

float basicSalary, grossSalary;

float hra, da;

// Input the basic salary

printf("Enter the basic salary: ");

scanf("%f", &basicSalary);

// Calculate HRA and DA based on the basic salary

if (basicSalary <= 10000) {

hra = 0.20 \* basicSalary;

da = 0.80 \* basicSalary;

} else if (basicSalary <= 20000) {

hra = 0.25 \* basicSalary;

da = 0.90 \* basicSalary;

} else {

hra = 0.30 \* basicSalary;

da = 0.95 \* basicSalary;

}

// Calculate the gross salary

grossSalary = basicSalary + hra + da;

// Display the result

printf("Basic Salary: %.2f\n", basicSalary);

printf("HRA: %.2f\n", hra);

printf("DA: %.2f\n", da);

printf("Gross Salary: %.2f\n", grossSalary);

return 0;

}

99. 23. Write a C program to input electricity unit charges and calculate total electricity bill according to the given condition: For first 50 units Rs. 0.50/unit For next 100 units Rs. 0.75/unit For next 100 units Rs. 1.20/unit For unit above 250 Rs. 1.50/unit An additional surcharge of 20% is added to the bill.

#include <stdio.h>

int main() { printf(“Navjot”);

float units, totalBill;

// Input the electricity units consumed

printf("Enter the number of electricity units consumed: ");

scanf("%f", &units);

// Calculate the total electricity bill

if (units <= 50) {

totalBill = units \* 0.50;

} else if (units <= 150) {

totalBill = 50 \* 0.50 + (units - 50) \* 0.75;

} else if (units <= 250) {

totalBill = 50 \* 0.50 + 100 \* 0.75 + (units - 150) \* 1.20;

} else {

totalBill = 50 \* 0.50 + 100 \* 0.75 + 100 \* 1.20 + (units - 250) \* 1.50;

}

// Add a 20% surcharge to the bill

totalBill = totalBill + 0.20 \* totalBill;

// Display the total bill

printf("Total Electricity Bill: Rs. %.2f\n", totalBill);

return 0;

}

100. 24. Write a C program to convert specified days into years, weeks and days.

#include <stdio.h>

int main() { printf(“Navjot”);

int days, years, weeks;

days = 1329;

// Converts days to years, weeks, and days

years = days / 365;

weeks = (days % 365) / 7;

days = days - ((years \* 365) + (weeks \* 7));

printf("Years: %d\n", years);

printf("Weeks: %d\n", weeks);

printf("Days: %d\n", days);

return 0;

}

101. Write a program in C to read n number of values in an array and display them in reverse order.

#include <stdio.h>

int main() { printf(“Navjot”);

int n;

printf("Enter the number of values you want to store in the array: ");

scanf("%d", &n);

if (n <= 0) {

printf("Invalid input. Please enter a positive integer.\n");

return 1; // Exit with an error code

}

int arr[n];

// Input values into the array

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

printf("Enter value %d: ", i + 1);

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

}

// Display the values in reverse order

printf("Values in reverse order: ");

for (int i = n - 1; i >= 0; i--) {

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

}

printf("\n");

return 0; // Exit with success

}

102. Write a program in C to find the sum of all elements of the array

#include <stdio.h>

int main() { printf(“Navjot”);

int n, sum = 0;

printf("Enter the number of elements in the array: ");

scanf("%d", &n);

if (n <= 0) {

printf("Invalid input. Please enter a positive integer.\n");

return 1; // Exit with an error code

}

int arr[n];

// Input elements into the array

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

printf("Enter element %d: ", i + 1);

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

}

// Calculate the sum of elements

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

sum += arr[i];

}

printf("Sum of elements in the array: %d\n", sum);

return 0; // Exit with success

}

103. Write a program in C to copy the elements of one array into another array.

#include <stdio.h>

int main() { printf(“Navjot”);

int n;

printf("Enter the number of elements in the source array: ");

scanf("%d", &n);

if (n <= 0) {

printf("Invalid input. Please enter a positive integer.\n");

return 1; // Exit with an error code

}

int sourceArray[n];

int destinationArray[n];

// Input elements into the source array

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

printf("Enter element %d of the source array: ", i + 1);

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

}

// Copy elements from sourceArray to destinationArray

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

destinationArray[i] = sourceArray[i];

}

// Display the elements in the destination array

printf("Elements in the destination array: ");

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

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

}

printf("\n");

return 0; // Exit with success

}

104. Write a program in C to count the total number of duplicate elements in an array.

#include <stdio.h>

int main() { printf(“Navjot”);

int n;

printf("Enter the number of elements in the array: ");

scanf("%d", &n);

if (n <= 0) {

printf("Invalid input. Please enter a positive integer.\n");

return 1; // Exit with an error code

}

int arr[n];

int count = 0;

// Input elements into the array

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

printf("Enter element %d: ", i + 1);

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

}

// Count the duplicate elements

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

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

if (arr[i] == arr[j]) {

count++;

break; // To avoid counting the same element multiple times

}

}

}

printf("Total number of duplicate elements in the array: %d\n", count);

return 0; // Exit with success

}

105. Write a program in C to find the maximum and minimum elements in an array.

#include <stdio.h>

int main() { printf(“Navjot”);

int n;

printf("Enter the number of elements in the array: ");

scanf("%d", &n);

if (n <= 0) {

printf("Invalid input. Please enter a positive integer.\n");

return 1; // Exit with an error code

}

int arr[n];

// Input elements into the array

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

printf("Enter element %d: ", i + 1);

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

}

int max = arr[0]; // Initialize max with the first element

int min = arr[0]; // Initialize min with the first element

// Find the maximum and minimum elements

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

if (arr[i] > max) {

max = arr[i];

}

if (arr[i] < min) {

min = arr[i];

}

}

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

printf("Minimum element in the array: %d\n", min);

return 0; // Exit with success

}

106. Write a C program to sort the elements of an array in descending order.

#include <stdio.h>

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

int temp;

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

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

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

// Swap the elements if they are in the wrong order

temp = arr[j];

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

arr[j + 1] = temp;

}

}

}

}

int main() { printf(“Navjot”);

int n;

printf("Enter the number of elements in the array: ");

scanf("%d", &n);

if (n <= 0) {

printf("Invalid input. Please enter a positive integer.\n");

return 1; // Exit with an error code

}

int arr[n];

// Input elements into the array

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

printf("Enter element %d: ", i + 1);

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

}

// Sort the array in descending order using bubble sort

bubbleSort(arr, n);

// Display the sorted array in descending order

printf("Sorted array in descending order: ");

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

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

}

printf("\n");

return 0; // Exit with success

}

107. Write a program in C to separate odd and even integers into separate arrays.

#include <stdio.h>

int main() { printf(“Navjot”);

int n;

printf("Enter the number of elements in the array: ");

scanf("%d", &n);

if (n <= 0) {

printf("Invalid input. Please enter a positive integer.\n");

return 1; // Exit with an error code

}

int originalArray[n];

int evenArray[n]; // To store even integers

int oddArray[n]; // To store odd integers

int evenCount = 0; // Count of even integers

int oddCount = 0; // Count of odd integers

// Input elements into the original array

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

printf("Enter element %d: ", i + 1);

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

}

// Separate even and odd integers into separate arrays

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

if (originalArray[i] % 2 == 0) {

evenArray[evenCount] = originalArray[i];

evenCount++;

} else {

oddArray[oddCount] = originalArray[i];

oddCount++;

}

}

// Display even integers

printf("Even integers: ");

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

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

}

printf("\n");

// Display odd integers

printf("Odd integers: ");

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

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

}

printf("\n");

return 0; // Exit with success

}

108. Write a program in C to merge two arrays of the same size sorted in descending/ascending order.

#include <stdio.h>

int main() { printf(“Navjot”);

int n;

printf("Enter the number of elements in each array: ");

scanf("%d", &n);

if (n <= 0) {

printf("Invalid input. Please enter a positive integer.\n");

return 1; // Exit with an error code

}

int firstArray[n];

int secondArray[n];

int mergedArray[2 \* n];

int i, j, k;

// Input elements into the first array

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

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

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

}

// Input elements into the second array

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

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

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

}

// Merge the two arrays in descending order

i = 0; // Index for the first array

j = 0; // Index for the second array

k = 0; // Index for the merged array

while (i < n && j < n) {

if (firstArray[i] >= secondArray[j]) {

mergedArray[k] = firstArray[i];

i++;

} else {

mergedArray[k] = secondArray[j];

j++;

}

k++;

}

// Copy the remaining elements from the first array, if any

while (i < n) {

mergedArray[k] = firstArray[i];

i++;

k++;

}

// Copy the remaining elements from the second array, if any

while (j < n) {

mergedArray[k] = secondArray[j];

j++;

k++;

}

// Display the merged array in descending order

printf("Merged array in descending order: ");

for (i = 0; i < 2 \* n; i++) {

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

}

printf("\n");

return 0; // Exit with success

}

109. Write a program in C to merge two arrays of the same size sorted in descending order.

#include <stdio.h>

int main() { printf(“Navjot”);

int n;

printf("Enter the number of elements in each array: ");

scanf("%d", &n);

if (n <= 0) {

printf("Invalid input. Please enter a positive integer.\n");

return 1; // Exit with an error code

}

int firstArray[n];

int secondArray[n];

int mergedArray[2 \* n];

int i, j, k;

// Input elements into the first array

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

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

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

}

// Input elements into the second array

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

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

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

}

// Merge the two arrays in descending order

i = 0; // Index for the first array

j = 0; // Index for the second array

k = 0; // Index for the merged array

while (i < n && j < n) {

if (firstArray[i] >= secondArray[j]) {

mergedArray[k] = firstArray[i];

i++;

} else {

mergedArray[k] = secondArray[j];

j++;

}

k++;

}

// Copy the remaining elements from the first array, if any

while (i < n) {

mergedArray[k] = firstArray[i];

i++;

k++;

}

// Copy the remaining elements from the second array, if any

while (j < n) {

mergedArray[k] = secondArray[j];

j++;

k++;

}

// Display the merged array in descending order

printf("Merged array in descending order: ");

for (i = 0; i < 2 \* n; i++) {

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

}

printf("\n");

return 0; // Exit with success

}

110. WAP using Switch case: Consider two matrices of the size m and n. Implement matrix operation and display. Show these things in program 1) Read matrix elements and display 2) Matrix Multiplication and display 3) addition of matrix and display 4)Subtraction of Matrix and display 5)Transpose of Matrix and display.

#include <stdio.h>

void readMatrix(int matrix[][10], int m, int n) {

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

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

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

printf("Enter element at row %d, column %d: ", i + 1, j + 1);

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

}

}

}

void displayMatrix(int matrix[][10], int m, int n) {

printf("Matrix:\n");

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

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

printf("%d\t", matrix[i][j]);

}

printf("\n");

}

}

void matrixAddition(int matrix1[][10], int matrix2[][10], int result[][10], int m, int n) {

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

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

result[i][j] = matrix1[i][j] + matrix2[i][j];

}

}

}

void matrixSubtraction(int matrix1[][10], int matrix2[][10], int result[][10], int m, int n) {

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

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

result[i][j] = matrix1[i][j] - matrix2[i][j];

}

}

}

void matrixMultiplication(int matrix1[][10], int m1, int n1, int matrix2[][10], int m2, int n2, int result[][10]) {

if (n1 != m2) {

printf("Matrix multiplication is not possible. Column count of the first matrix must be equal to the row count of the second matrix.\n");

return;

}

for (int i = 0; i < m1; i++) {

for (int j = 0; j < n2; j++) {

result[i][j] = 0;

for (int k = 0; k < n1; k++) {

result[i][j] += matrix1[i][k] \* matrix2[k][j];

}

}

}

}

void matrixTranspose(int matrix[][10], int m, int n, int result[][10]) {

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

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

result[j][i] = matrix[i][j];

}

}

}

int main() { printf(“Navjot”);

int choice, m, n;

int matrix1[10][10], matrix2[10][10], result[10][10];

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

scanf("%d", &m);

printf("Enter the number of columns for the matrices: ");

scanf("%d");

printf("Matrix Operations:\n");

printf("1. Read and Display Matrix\n");

printf("2. Matrix Multiplication\n");

printf("3. Matrix Addition\n");

printf("4. Matrix Subtraction\n");

printf("5. Transpose of Matrix\n");

printf("Enter your choice (1-5): ");

scanf("%d", &choice);

switch (choice) {

case 1:

printf("For Matrix 1:\n");

readMatrix(matrix1, m, n);

printf("Matrix 1:\n");

displayMatrix(matrix1, m, n);

printf("For Matrix 2:\n");

readMatrix(matrix2, m, n);

printf("Matrix 2:\n");

displayMatrix(matrix2, m, n);

break;

case 2:

readMatrix(matrix1, m, n);

readMatrix(matrix2, m, n);

matrixMultiplication(matrix1, m, n, matrix2, m, n, result);

printf("Matrix 1:\n");

displayMatrix(matrix1, m, n);

printf("Matrix 2:\n");

displayMatrix(matrix2, m, n);

printf("Matrix Multiplication Result:\n");

displayMatrix(result, m, n);

break;

case 3:

readMatrix(matrix1, m, n);

readMatrix(matrix2, m, n);

matrixAddition(matrix1, matrix2, result, m, n);

printf("Matrix 1:\n");

displayMatrix(matrix1, m, n);

printf("Matrix 2:\n");

displayMatrix(matrix2, m, n);

printf("Matrix Addition Result:\n");

displayMatrix(result, m, n);

break;

case 4:

readMatrix(matrix1, m, n);

readMatrix(matrix2, m, n);

matrixSubtraction(matrix1, matrix2, result, m, n);

printf("Matrix 1:\n");

displayMatrix(matrix1, m, n);

printf("Matrix 2:\n");

displayMatrix(matrix2, m, n);

printf("Matrix Subtraction Result:\n");

displayMatrix(result, m, n);

break;

case 5:

readMatrix(matrix1, m, n);

printf("Matrix 1:\n");

displayMatrix(matrix1, m, n);

matrixTranspose(matrix1, m, n, result);

printf("Transpose of Matrix 1:\n");

displayMatrix(result, n, m);

break;

default:

printf("Invalid choice. Please enter a valid option.\n");

}

return 0;

}

111. Write a program in C to read n number of values in an array and display them in reverse order:

#include <stdio.h>

int main() { printf(“Navjot”);

int n;

printf("Navjot Jangra\n");

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

scanf("%d", &n);

if (n <= 0) {

printf("Invalid input for the number of values.\n");

return 1;

}

int arr[n];

printf("Enter %d values, one at a time:\n", n);

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

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

}

printf("Values in reverse order:\n");

for (int i = n - 1; i >= 0; i--) {

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

}

return 0;

}

112. Write a program in C to find the sum of all elements of the array:

#include <stdio.h>

int main() { printf(“Navjot”);

int n, sum = 0;

printf("Navjot Jangra\n");

printf("Enter the number of elements in the array: ");

scanf("%d", &n);

if (n <= 0) {

printf("Invalid input for the number of elements.\n");

return 1;

}

int arr[n];

printf("Enter %d elements, one at a time:\n", n);

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

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

sum += arr[i];

}

printf("Sum of all elements in the array: %d\n", sum);

return 0;

}

113. Write a program in C to copy the elements of one array into another array:

#include <stdio.h>

int main() { printf(“Navjot”);

int n;

printf("Navjot Jangra\n");

printf("Enter the number of elements in the array: ");

scanf("%d", &n);

if (n <= 0) {

printf("Invalid input for the number of elements.\n");

return 1;

}

int Arrayone[n];

int Arraytwo[n];

printf("Enter %d elements for the arrayone, one at a time:\n", n);

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

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

}

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

Arraytwo[i] = Arrayone[i];

}

printf("Elements copied from Arrayone to Arraytwo:\n");

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

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

}

printf("\n");

return 0;

}

114. Write a program in C to count the total number of duplicate elements in an array:

#include <stdio.h>

int main() { printf(“Navjot”);

int n, count = 0;

printf("Navjot Jangra\n");

printf("Enter the number of elements in the array: ");

scanf("%d", &n);

if (n <= 0) {

printf("Invalid input for the number of elements.\n");

return 1;

}

int arr[n];

printf("Enter %d elements, one at a time:\n", n);

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

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

}

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

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

if (arr[i] == arr[j]) {

count++;

break;

}

}

}

if (count > 0) {

printf("Total number of duplicate elements in the array: %d\n", count);

} else {

printf("No duplicate elements found in the array.\n");

}

return 0;

}

115. Write a program in C to find the maximum and minimum elements in an array:

int main() { printf(“Navjot”);

#include <stdio.h>

int n;

printf("Navjot Jangra\n");

printf("Enter the number of elements in the array: ");

scanf("%d", &n);

if (n <= 0) {

printf("Invalid input for the number of elements.\n");

return 1;

}

int arr[n];

printf("Enter %d elements, one at a time:\n", n);

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

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

}

int max = arr[0];

int min = arr[0];

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

if (arr[i] > max) {

max = arr[i];

}

if (arr[i] < min) {

min = arr[i];

}

}

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

printf("Minimum element in the array: %d\n", min);

return 0;

}

116. Write a C program to sort the elements of an array in descending order:

#include <stdio.h>

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

int temp;

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

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

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

temp = arr[j];

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

arr[j + 1] = temp;

}

}

}

}

int main() { printf(“Navjot”);

int n;

printf("Navjot Jangra\n");

printf("Enter the number of elements in the array: ");

scanf("%d", &n);

if (n <= 0) {

printf("Invalid input for the number of elements.\n");

return 1;

}

int arr[n];

printf("Enter %d elements, one at a time:\n", n);

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

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

}

bubbleSort(arr, n);

printf("Elements in descending order: ");

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

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

}

printf("\n");

return 0;

}

117. Write a program in C to separate odd and even integers into separate arrays:

#include <stdio.h>

int main() { printf(“Navjot”);

int n;

printf("Navjot Jangra\n");

printf("Enter the number of elements in the array: ");

scanf("%d", &n);

if (n <= 0) {

printf("Invalid input for the number of elements.\n");

return 1;

}

int arr[n];

int evenArr[n];

int oddArr[n];

int evenCount = 0;

int oddCount = 0;

printf("Enter %d elements, one at a time:\n", n);

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

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

if (arr[i] % 2 == 0) {

evenArr[evenCount] = arr[i];

evenCount++;

} else {

oddArr[oddCount] = arr[i];

oddCount++;

}

}

printf("Even integers: ");

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

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

}

printf("\nOdd integers: ");

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

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

}

printf("\n");

return 0;

}

118. Write a program in C to merge two arrays of the same size sorted in descending/ascending order:

#include <stdio.h>

void mergeArrays(int arr1[], int arr2[], int n, int mergedArr[], int ascending) {

int i = 0, j = 0, k = 0;

while (i < n && j < n) {

if (ascending ? (arr1[i] < arr2[j]) : (arr1[i] > arr2[j])) {

mergedArr[k] = arr1[i];

i++;

} else {

mergedArr[k] = arr2[j];

j++;

}

k++;

}

while (i < n) {

mergedArr[k] = arr1[i];

i++;

k++;

}

while (j < n) {

mergedArr[k] = arr2[j];

j++;

k++;

}

}

int main() { printf(“Navjot”);

int n;

printf("Navjot Jangra\n");

printf("Enter the number of elements in each array: ");

scanf("%d", &n);

if (n <= 0) {

printf("Invalid input for the number of elements.\n");

return 1;

}

int arr1[n];

int arr2[n];

int mergedArr[2 \* n];

printf("Enter %d elements for the first array, one at a time:\n", n);

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

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

}

printf("Enter %d elements for the second array, one at a time:\n", n);

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

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

}

int choice;

printf("Enter 1 to merge in ascending order, or 0 to merge in descending order: ");

scanf("%d", &choice);

if (choice != 0 && choice != 1) {

printf("Invalid choice. Please enter 1 for ascending or 0 for descending order.\n");

return 1;

}

mergeArrays(arr1, arr2, n, mergedArr, choice);

printf("Merged array in %s order: ", (choice ? "ascending" : "descending"));

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

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

}

printf("\n");

return 0;

}