Here are the C programs for each of the tasks you've requested:

1. Check if a character is a vowel or consonant

#include <stdio.h>

int main() {

char ch;

printf("Enter a character: ");

scanf("%c", &ch);

if (ch == 'a' || ch == 'e' || ch == 'i' || ch == 'o' || ch == 'u' ||

ch == 'A' || ch == 'E' || ch == 'I' || ch == 'O' || ch == 'U') {

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

} else {

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

}

return 0;

}

```

2. Check if a character is an alphabet or not

#include <stdio.h>

#include <ctype.h>

int main() {

char ch;

printf("Enter a character: ");

scanf("%c", &ch);

if (isalpha(ch)) {

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

} else {

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

}

return 0;

}

```

3. Find out ASCII value of a character

#include <stdio.h>

int main() {

char ch;

printf("Enter a character: ");

scanf("%c", &ch);

printf("ASCII value of %c is %d.\n", ch, ch);

return 0;

}

```

4. Identify if a number is positive or negative

#include <stdio.h>

int main() {

float num;

printf("Enter a number: ");

scanf("%f", &num);

if (num > 0) {

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

} else if (num < 0) {

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

} else {

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

}

return 0;

}

```

5. Identify if a number is even or odd

#include <stdio.h>

int main() {

int num;

printf("Enter an integer: ");

scanf("%d", &num);

if (num % 2 == 0) {

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

} else {

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

}

return 0;

}

```

6. Swap two numbers without a third variable

#include <stdio.h>

int main() {

int a, b;

printf("Enter first number: ");

scanf("%d", &a);

printf("Enter second number: ");

scanf("%d", &b);

a = a + b;

b = a - b;

a = a - b;

printf("After swapping: a = %d, b = %d\n", a, b);

return 0;

}

```

7. Find out area of a circle (radius value user given)

#include <stdio.h>

#include <math.h>

int main() {

float radius, area;

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

scanf("%f", &radius);

area = M\_PI \* radius \* radius;

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

return 0;

}

```

8. Find out LCM of two numbers

#include <stdio.h>

int lcm(int a, int b) {

int max = (a > b) ? a : b;

while (1) {

if (max % a == 0 && max % b == 0) {

return max;

}

++max;

}

}

int main() {

int num1, num2;

printf("Enter first number: ");

scanf("%d", &num1);

printf("Enter second number: ");

scanf("%d", &num2);

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

return 0;

}

```

9. Find out GCD of two numbers

#include <stdio.h>

int gcd(int a, int b) {

while (b != 0) {

int temp = b;

b = a % b;

a = temp;

}

return a;

}

int main() {

int num1, num2;

printf("Enter first number: ");

scanf("%d", &num1);

printf("Enter second number: ");

scanf("%d", &num2);

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

return 0;

}

```

10. Find out greatest of two numbers

#include <stdio.h>

int main() {

int num1, num2;

printf("Enter first number: ");

scanf("%d", &num1);

printf("Enter second number: ");

scanf("%d", &num2);

if (num1 > num2) {

printf("The greatest number is %d.\n", num1);

} else {

printf("The greatest number is %d.\n", num2);

}

return 0;

}

```

11. Find out greatest of three numbers

#include <stdio.h>

int main() {

int num1, num2, num3;

printf("Enter first number: ");

scanf("%d", &num1);

printf("Enter second number: ");

scanf("%d", &num2);

printf("Enter third number: ");

scanf("%d", &num3);

int greatest = num1;

if (num2 > greatest) {

greatest = num2;

}

if (num3 > greatest) {

greatest = num3;

}

printf("The greatest number is %d.\n", greatest);

return 0;

}

```

12. Find out number of digits in an integer

#include <stdio.h>

int main() {

int num, count = 0;

printf("Enter an integer: ");

scanf("%d", &num);

do {

num /= 10;

count++;

} while (num != 0);

printf("The number of digits is %d.\n", count);

return 0;

}

```

13. Find out sum of digits of a number

#include <stdio.h>

int main() {

int num, sum = 0;

printf("Enter an integer: ");

scanf("%d", &num);

while (num != 0) {

sum += num % 10;

num /= 10;

}

printf("The sum of the digits is %d.\n", sum);

return 0;

}

```

14. Find out sum of N natural numbers

#include <stdio.h>

int main() {

int n, sum = 0;

printf("Enter a positive integer: ");

scanf("%d", &n);

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

sum += i;

}

printf("The sum of the first %d natural numbers is %d.\n", n, sum);

return 0;

}

```

15. Find out sum of numbers in a given range

#include <stdio.h>

int main() {

int start, end, sum = 0;

printf("Enter the start of the range: ");

scanf("%d", &start);

printf("Enter the end of the range: ");

scanf("%d", &end);

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

sum += i;

}

printf("The sum of the numbers in the range is %d.\n", sum);

return 0;

}

```

16. Find out reverse of a given number

#include <stdio.h>

int main() {

int num, reversed = 0;

printf("Enter an integer: ");

scanf("%d", &num);

while (num != 0) {

reversed = reversed \* 10 + num % 10;

num /= 10;

}

printf("The reversed number is %d.\n", reversed);

return 0;

}

```

17. Find out factorial of a number

#include <stdio.h>

int main() {

int n;

unsigned long long factorial = 1;

printf("Enter an integer: ");

scanf("%d", &n);

if (n < 0) {

printf("Factorial of a negative number doesn't exist.\n");

} else {

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

factorial \*= i;

}

printf("Factorial of %d is %llu.\n", n, factorial);

}

return 0;

}

```

18. Find out Fibonacci series up to n

#include <stdio.h>

int main() {

int n, t1 = 0, t2 = 1, nextTerm;

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

scanf("%d", &n);

printf("Fibonacci Series: %d, %d", t1, t

2);

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

nextTerm = t1 + t2;

printf(", %d", nextTerm);

t1 = t2;

t2 = nextTerm;

}

printf("\n");

return 0;

}

```

19. Find out if a year is a leap year or not

#include <stdio.h>

int main() {

int year;

printf("Enter a year: ");

scanf("%d", &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;

}

```

20. Identify if a user-given number is a prime number or not

#include <stdio.h>

int main() {

int num, flag = 0;

printf("Enter a positive integer: ");

scanf("%d", &num);

if (num <= 1) {

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

} else {

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

if (num % i == 0) {

flag = 1;

break;

}

}

if (flag == 0) {

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

} else {

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

}

}

return 0;

}

```

21. Identify if a user-given number is a palindrome or not

#include <stdio.h>

int main() {

int num, originalNum, reversed = 0;

printf("Enter an integer: ");

scanf("%d", &num);

originalNum = num;

while (num != 0) {

reversed = reversed \* 10 + num % 10;

num /= 10;

}

if (originalNum == reversed) {

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

} else {

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

}

return 0;

}

```

22. Identify if a user-given number is an Armstrong number or not

#include <stdio.h>

#include <math.h>

int main() {

int num, originalNum, remainder, n = 0;

float result = 0.0;

printf("Enter an integer: ");

scanf("%d", &num);

originalNum = num;

for (originalNum = num; originalNum != 0; ++n) {

originalNum /= 10;

}

for (originalNum = num; originalNum != 0; originalNum /= 10) {

remainder = originalNum % 10;

result += pow(remainder, n);

}

if ((int)result == num) {

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

} else {

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

}

return 0;

}

```

23. Identify if a user-given number is a Strong number or not

#include <stdio.h>

int factorial(int n) {

if (n == 0 || n == 1) {

return 1;

}

return n \* factorial(n - 1);

}

int main() {

int num, originalNum, remainder, sum = 0;

printf("Enter an integer: ");

scanf("%d", &num);

originalNum = num;

while (num != 0) {

remainder = num % 10;

sum += factorial(remainder);

num /= 10;

}

if (sum == originalNum) {

printf("%d is a Strong number.\n", originalNum);

} else {

printf("%d is not a Strong number.\n", originalNum);

}

return 0;

}

```

24. Identify if a user-given number is a Perfect number or not

#include <stdio.h>

int main() {

int num, sum = 0;

printf("Enter an integer: ");

scanf("%d", &num);

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

if (num % i == 0) {

sum += i;

}

}

if (sum == num) {

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

} else {

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

}

return 0;

}

```

25. Identify if two user-given numbers are a friendly pair (amicable numbers) or not

#include <stdio.h>

int sumOfDivisors(int num) {

int sum = 0;

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

if (num % i == 0) {

sum += i;

}

}

return sum;

}

int main() {

int num1, num2;

printf("Enter the first number: ");

scanf("%d", &num1);

printf("Enter the second number: ");

scanf("%d", &num2);

if (sumOfDivisors(num1) == num2 && sumOfDivisors(num2) == num1) {

printf("%d and %d are amicable numbers.\n", num1, num2);

} else {

printf("%d and %d are not amicable numbers.\n", num1, num2);

}

return 0;

}

```

These programs cover a wide range of basic tasks and should help you get started with C programming.

Here are the C programs for each of the tasks you've requested:

26. Identify Automorphic number or not

#include <stdio.h>

#include <math.h>

int main() {

int num, sq, lastDigits;

printf("Enter a number: ");

scanf("%d", &num);

sq = num \* num;

int digits = log10(num) + 1;

lastDigits = sq % (int)pow(10, digits);

if (lastDigits == num) {

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

} else {

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

}

return 0;

}

```

27. Identify Harshad number or not

#include <stdio.h>

int main() {

int num, sum = 0, originalNum;

printf("Enter a number: ");

scanf("%d", &num);

originalNum = num;

while (num != 0) {

sum += num % 10;

num /= 10;

}

if (originalNum % sum == 0) {

printf("%d is a Harshad number.\n", originalNum);

} else {

printf("%d is not a Harshad number.\n", originalNum);

}

return 0;

}

```

28. Identify Abundant number or not

#include <stdio.h>

int main() {

int num, sum = 0;

printf("Enter a number: ");

scanf("%d", &num);

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

if (num % i == 0) {

sum += i;

}

}

if (sum > num) {

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

} else {

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

}

return 0;

}

```

29. Find out power of a number

#include <stdio.h>

#include <math.h>

int main() {

double base, exponent, result;

printf("Enter base: ");

scanf("%lf", &base);

printf("Enter exponent: ");

scanf("%lf", &exponent);

result = pow(base, exponent);

printf("%.2lf ^ %.2lf = %.2lf\n", base, exponent, result);

return 0;

}

```

30. Find out factors of a number

#include <stdio.h>

int main() {

int num;

printf("Enter a number: ");

scanf("%d", &num);

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

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

if (num % i == 0) {

printf("%d ", i);

}

}

printf("\n");

return 0;

}

```

31. Print prime numbers in a given range

#include <stdio.h>

#include <stdbool.h>

bool isPrime(int num) {

if (num <= 1) return false;

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

if (num % i == 0) return false;

}

return true;

}

int main() {

int start, end;

printf("Enter the start of the range: ");

scanf("%d", &start);

printf("Enter the end of the range: ");

scanf("%d", &end);

printf("Prime numbers between %d and %d are: ", start, end);

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

if (isPrime(i)) {

printf("%d ", i);

}

}

printf("\n");

return 0;

}

```

32. Print Armstrong numbers between two intervals

#include <stdio.h>

#include <math.h>

int isArmstrong(int num) {

int originalNum = num, sum = 0, n = 0, remainder;

while (originalNum != 0) {

originalNum /= 10;

++n;

}

originalNum = num;

while (originalNum != 0) {

remainder = originalNum % 10;

sum += pow(remainder, n);

originalNum /= 10;

}

return (sum == num);

}

int main() {

int start, end;

printf("Enter the start of the range: ");

scanf("%d", &start);

printf("Enter the end of the range: ");

scanf("%d", &end);

printf("Armstrong numbers between %d and %d are: ", start, end);

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

if (isArmstrong(i)) {

printf("%d ", i);

}

}

printf("\n");

return 0;

}

```

33. Can a number be expressed as a sum of two prime numbers?

#include <stdio.h>

#include <stdbool.h>

bool isPrime(int num) {

if (num <= 1) return false;

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

if (num % i == 0) return false;

}

return true;

}

void sumOfTwoPrimes(int num) {

bool found = false;

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

if (isPrime(i) && isPrime(num - i)) {

printf("%d can be expressed as the sum of %d and %d.\n", num, i, num - i);

found = true;

}

}

if (!found) {

printf("%d cannot be expressed as the sum of two prime numbers.\n", num);

}

}

int main() {

int num;

printf("Enter a number: ");

scanf("%d", &num);

sumOfTwoPrimes(num);

return 0;

}

```

34. Replace all 0’s with 1 in a given integer

#include <stdio.h>

int replaceZerosWithOnes(int num) {

int result = 0, place = 1;

while (num > 0) {

int digit = num % 10;

if (digit == 0) {

digit = 1;

}

result += digit \* place;

place \*= 10;

num /= 10;

}

return result;

}

int main() {

int num;

printf("Enter an integer: ");

scanf("%d", &num);

if (num == 0) {

num = 1;

} else {

num = replaceZerosWithOnes(num);

}

printf("Modified number: %d\n", num);

return 0;

}

```

35. Binary to decimal conversion

#include <stdio.h>

#include <math.h>

int binaryToDecimal(int binary) {

int decimal = 0, i = 0, remainder;

while (binary != 0) {

remainder = binary % 10;

binary /= 10;

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

++i;

}

return decimal;

}

int main() {

int binary;

printf("Enter a binary number: ");

scanf("%d", &binary);

printf("Decimal representation: %d\n", binaryToDecimal(binary));

return 0;

}

```

36. Decimal to binary conversion

#include <stdio.h>

void decimalToBinary(int num) {

if (num == 0) {

printf("0");

return;

}

int binary[32];

int i = 0;

while (num > 0) {

binary[i] = num % 2;

num /= 2;

i++;

}

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

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

}

}

int main() {

int num;

printf("Enter a decimal number: ");

scanf("%d", &num);

printf("Binary representation: ");

decimalToBinary(num);

printf("\n");

return 0;

}

```

37. Decimal to octal conversion

#include <stdio.h>

void decimalToOctal(int num) {

if (num == 0) {

printf("0");

return;

}

int octal[32];

int i = 0;

while (num > 0) {

octal[i] = num % 8;

num /= 8;

i++;

}

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

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

}

}

int main() {

int num;

printf("Enter a decimal number: ");

scanf("%d", &num);

printf("Octal representation: ");

decimalToOctal(num);

printf("\n");

return 0;

}

```

38. Octal to decimal conversion

#include <stdio.h>

#include <math.h>

int octalToDecimal(int octal) {

int decimal = 0, i = 0, remainder;

while (octal != 0) {

remainder

= octal % 10;

octal /= 10;

decimal += remainder \* pow(8, i);

++i;

}

return decimal;

}

int main() {

int octal;

printf("Enter an octal number: ");

scanf("%d", &octal);

printf("Decimal representation: %d\n", octalToDecimal(octal));

return 0;

}

```

39. Binary to octal conversion

#include <stdio.h>

#include <math.h>

int binaryToDecimal(int binary) {

int decimal = 0, i = 0, remainder;

while (binary != 0) {

remainder = binary % 10;

binary /= 10;

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

++i;

}

return decimal;

}

void decimalToOctal(int num) {

if (num == 0) {

printf("0");

return;

}

int octal[32];

int i = 0;

while (num > 0) {

octal[i] = num % 8;

num /= 8;

i++;

}

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

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

}

}

int main() {

int binary;

printf("Enter a binary number: ");

scanf("%d", &binary);

int decimal = binaryToDecimal(binary);

printf("Octal representation: ");

decimalToOctal(decimal);

printf("\n");

return 0;

}

```

40. Octal to binary conversion

#include <stdio.h>

#include <math.h>

int octalToDecimal(int octal) {

int decimal = 0, i = 0, remainder;

while (octal != 0) {

remainder = octal % 10;

octal /= 10;

decimal += remainder \* pow(8, i);

++i;

}

return decimal;

}

void decimalToBinary(int num) {

if (num == 0) {

printf("0");

return;

}

int binary[32];

int i = 0;

while (num > 0) {

binary[i] = num % 2;

num /= 2;

i++;

}

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

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

}

}

int main() {

int octal;

printf("Enter an octal number: ");

scanf("%d", &octal);

int decimal = octalToDecimal(octal);

printf("Binary representation: ");

decimalToBinary(decimal);

printf("\n");

return 0;

}

```

41. Convert digit/number to words

#include <stdio.h>

void numberToWords(int num) {

char \*single\_digits[] = { "zero", "one", "two", "three", "four", "five", "six", "seven", "eight", "nine" };

if (num == 0) {

printf("%s", single\_digits[0]);

return;

}

int reversedNum = 0;

while (num != 0) {

reversedNum = reversedNum \* 10 + num % 10;

num /= 10;

}

while (reversedNum != 0) {

int digit = reversedNum % 10;

printf("%s ", single\_digits[digit]);

reversedNum /= 10;

}

}

int main() {

int num;

printf("Enter a number: ");

scanf("%d", &num);

printf("Number in words: ");

numberToWords(num);

printf("\n");

return 0;

}

```

42. Find out number of days in a given month of a given year

#include <stdio.h>

#include <stdbool.h>

bool isLeapYear(int year) {

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

}

int main() {

int month, year;

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

scanf("%d", &month);

printf("Enter year: ");

scanf("%d", &year);

int days;

switch (month) {

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

days = 31;

break;

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

days = 30;

break;

case 2:

days = isLeapYear(year) ? 29 : 28;

break;

default:

printf("Invalid month.\n");

return 1;

}

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

return 0;

}

```

43. Find out permutations in which n people can occupy r seats in a theatre

#include <stdio.h>

int factorial(int n) {

int fact = 1;

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

fact \*= i;

}

return fact;

}

int main() {

int n, r;

printf("Enter total number of people (n): ");

scanf("%d", &n);

printf("Enter number of seats (r): ");

scanf("%d", &r);

if (n < r) {

printf("n should be greater than or equal to r.\n");

} else {

int permutations = factorial(n) / factorial(n - r);

printf("Number of permutations: %d\n", permutations);

}

return 0;

}

```

44. Find out number of integers which has exactly 9 divisors

#include <stdio.h>

#include <math.h>

int countDivisors(int num) {

int count = 0;

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

if (num % i == 0) {

if (num / i == i)

count++;

else

count += 2;

}

}

return count;

}

int main() {

int start, end, count = 0;

printf("Enter the start of the range: ");

scanf("%d", &start);

printf("Enter the end of the range: ");

scanf("%d", &end);

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

if (countDivisors(i) == 9) {

count++;

}

}

printf("Number of integers with exactly 9 divisors: %d\n", count);

return 0;

}

```

45. Find out roots of a quadratic equation

#include <stdio.h>

#include <math.h>

int main() {

double a, b, c, discriminant, root1, root2;

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

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

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

if (discriminant > 0) {

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

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

printf("Roots are real and different: %.2lf, %.2lf\n", root1, root2);

} else if (discriminant == 0) {

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

printf("Roots are real and same: %.2lf, %.2lf\n", root1, root2);

} else {

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

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

printf("Roots are complex and different: %.2lf + %.2lfi, %.2lf - %.2lfi\n",

realPart, imaginaryPart, realPart, imaginaryPart);

}

return 0;

}

```

46. Print solid rectangle star pattern

#include <stdio.h>

int main() {

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("\* ");

}

printf("\n");

}

return 0;

}

```

47. Print hollow rectangle star pattern

#include <stdio.h>

int main() {

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) {

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

printf("\* ");

} else {

printf(" ");

}

}

printf("\n");

}

return 0;

}

```

48. Print pyramid pattern using stars

#include <stdio.h>

int main() {

int rows;

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

scanf("%d", &rows);

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

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

printf(" ");

}

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

printf("\* ");

}

printf("\n");

}

return 0;

}

```

49. Print pyramid pattern using numbers

#include <stdio.h>

int main() {

int rows;

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

scanf("%d", &rows);

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

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

printf(" ");

}

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

printf("%d ", i);

}

printf("\n");

}

return 0;

}

```

50. Print diamond pattern using stars

#include <stdio.h>

int main() {

int rows, n;

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

scanf("%d", &rows);

n = rows;

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

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

printf(" ");

}

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

printf("\* ");

}

printf("\n");

}

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

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

printf(" ");

}

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

printf("\* ");

}

printf("\n");

}

return 0;

}

```

These programs cover a wide variety of tasks, from basic arithmetic and logic checks to more complex algorithms and pattern printing. Each one is designed to be clear and concise, making it easy to understand the logic and modify if needed.

Here are the C programs for the tasks specified:

51. Print diamond pattern using numbers

#include <stdio.h>

int main() {

int rows, n;

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

scanf("%d", &rows);

n = rows;

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

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

printf(" ");

}

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

printf("%d ", i);

}

printf("\n");

}

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

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

printf(" ");

}

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

printf("%d ", i);

}

printf("\n");

}

return 0;

}

```

52. Print Floyd’s triangle

#include <stdio.h>

int main() {

int rows, number = 1;

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

scanf("%d", &rows);

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

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

printf("%d ", number++);

}

printf("\n");

}

return 0;

}

```

53. Print Pascal's triangle

#include <stdio.h>

int factorial(int n) {

int fact = 1;

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

fact \*= i;

}

return fact;

}

int combination(int n, int k) {

return factorial(n) / (factorial(k) \* factorial(n - k));

}

int main() {

int rows;

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

scanf("%d", &rows);

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

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

printf("%d ", combination(i, j));

}

printf("\n");

}

return 0;

}

```

54. Find out smallest and largest element in an array

#include <stdio.h>

int main() {

int n;

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

scanf("%d", &n);

int arr[n];

printf("Enter the elements: ");

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

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

}

int smallest = arr[0], largest = arr[0];

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

if (arr[i] < smallest) {

smallest = arr[i];

}

if (arr[i] > largest) {

largest = arr[i];

}

}

printf("Smallest element: %d\n", smallest);

printf("Largest element: %d\n", largest);

return 0;

}

```

55. Find out sum of elements in an array

#include <stdio.h>

int main() {

int n, sum = 0;

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

scanf("%d", &n);

int arr[n];

printf("Enter the elements: ");

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

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

}

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

sum += arr[i];

}

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

return 0;

}

```

56. Check if two arrays are the same or not

#include <stdio.h>

#include <stdbool.h>

bool areArraysSame(int arr1[], int arr2[], int n) {

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

if (arr1[i] != arr2[i]) {

return false;

}

}

return true;

}

int main() {

int n;

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

scanf("%d", &n);

int arr1[n], arr2[n];

printf("Enter elements of first array: ");

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

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

}

printf("Enter elements of second array: ");

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

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

}

if (areArraysSame(arr1, arr2, n)) {

printf("The arrays are the same.\n");

} else {

printf("The arrays are not the same.\n");

}

return 0;

}

```

57. Find out sum of positive square elements in an array

#include <stdio.h>

int main() {

int n, sum = 0;

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

scanf("%d", &n);

int arr[n];

printf("Enter the elements: ");

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

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

}

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

if (arr[i] > 0) {

sum += arr[i] \* arr[i];

}

}

printf("Sum of positive square elements: %d\n", sum);

return 0;

}

```

58. Identify second smallest element in an array

#include <stdio.h>

#include <limits.h>

int main() {

int n;

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

scanf("%d", &n);

int arr[n];

printf("Enter the elements: ");

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

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

}

int first = INT\_MAX, second = INT\_MAX;

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

if (arr[i] < first) {

second = first;

first = arr[i];

} else if (arr[i] < second && arr[i] != first) {

second = arr[i];

}

}

if (second == INT\_MAX) {

printf("No second smallest element.\n");

} else {

printf("Second smallest element: %d\n", second);

}

return 0;

}

```

59. Sort elements of an array

#include <stdio.h>

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

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

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

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

int temp = arr[j];

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

arr[j + 1] = temp;

}

}

}

}

int main() {

int n;

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

scanf("%d", &n);

int arr[n];

printf("Enter the elements: ");

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

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

}

sortArray(arr, n);

printf("Sorted elements: ");

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

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

}

printf("\n");

return 0;

}

```

60. Reverse an array

#include <stdio.h>

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

int start = 0, end = n - 1;

while (start < end) {

int temp = arr[start];

arr[start] = arr[end];

arr[end] = temp;

start++;

end++;

}

}

int main() {

int n;

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

scanf("%d", &n);

int arr[n];

printf("Enter the elements: ");

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

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

}

reverseArray(arr, n);

printf("Reversed array: ");

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

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

}

printf("\n");

return 0;

}

```

61. Find out longest palindrome in an array

#include <stdio.h>

#include <stdbool.h>

bool isPalindrome(int num) {

int reversed = 0, original = num;

while (num != 0) {

reversed = reversed \* 10 + num % 10;

num /= 10;

}

return original == reversed;

}

int main() {

int n;

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

scanf("%d", &n);

int arr[n];

printf("Enter the elements: ");

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

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

}

int maxPalindrome =

-1;

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

if (isPalindrome(arr[i]) && arr[i] > maxPalindrome) {

maxPalindrome = arr[i];

}

}

if (maxPalindrome == -1) {

printf("No palindrome found.\n");

} else {

printf("Longest palindrome: %d\n", maxPalindrome);

}

return 0;

}

```

62. Count distinct elements of an array

#include <stdio.h>

#include <stdbool.h>

int main() {

int n;

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

scanf("%d", &n);

int arr[n];

printf("Enter the elements: ");

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

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

}

int count = 0;

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

bool isDistinct = true;

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

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

isDistinct = false;

break;

}

}

if (isDistinct) {

count++;

}

}

printf("Number of distinct elements: %d\n", count);

return 0;

}

```

63. Print and count non-repeating elements of an array

#include <stdio.h>

#include <stdbool.h>

int main() {

int n;

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

scanf("%d", &n);

int arr[n];

printf("Enter the elements: ");

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

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

}

int nonRepeatingCount = 0;

printf("Non-repeating elements: ");

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

bool isNonRepeating = true;

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

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

isNonRepeating = false;

break;

}

}

if (isNonRepeating) {

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

nonRepeatingCount++;

}

}

printf("\nCount of non-repeating elements: %d\n", nonRepeatingCount);

return 0;

}

```

64. Print and count repeating elements in an array

#include <stdio.h>

#include <stdbool.h>

int main() {

int n;

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

scanf("%d", &n);

int arr[n];

printf("Enter the elements: ");

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

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

}

bool counted[n];

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

counted[i] = false;

}

int repeatingCount = 0;

printf("Repeating elements: ");

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

if (!counted[i]) {

int count = 0;

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

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

count++;

counted[j] = true;

}

}

if (count > 1) {

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

repeatingCount++;

}

}

}

printf("\nCount of repeating elements: %d\n", repeatingCount);

return 0;

}

```

65. Remove duplicate elements in an array

#include <stdio.h>

int main() {

int n;

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

scanf("%d", &n);

int arr[n];

printf("Enter the elements: ");

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

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

}

int index = 0;

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

int j;

for (j = 0; j < index; j++) {

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

break;

}

}

if (j == index) {

arr[index++] = arr[i];

}

}

printf("Array after removing duplicates: ");

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

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

}

printf("\n");

return 0;

}

```

66. Find out minimum scalar product of two vectors

#include <stdio.h>

#include <stdlib.h>

int compare(const void \*a, const void \*b) {

return (\*(int \*)a - \*(int \*)b);

}

int main() {

int n;

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

scanf("%d", &n);

int vector1[n], vector2[n];

printf("Enter elements of first vector: ");

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

scanf("%d", &vector1[i]);

}

printf("Enter elements of second vector: ");

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

scanf("%d", &vector2[i]);

}

qsort(vector1, n, sizeof(int), compare);

qsort(vector2, n, sizeof(int), compare);

int scalarProduct = 0;

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

scalarProduct += vector1[i] \* vector2[n - 1 - i];

}

printf("Minimum scalar product: %d\n", scalarProduct);

return 0;

}

```

67. Find out maximum scalar product of two vectors

#include <stdio.h>

#include <stdlib.h>

int compare(const void \*a, const void \*b) {

return (\*(int \*)a - \*(int \*)b);

}

int main() {

int n;

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

scanf("%d", &n);

int vector1[n], vector2[n];

printf("Enter elements of first vector: ");

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

scanf("%d", &vector1[i]);

}

printf("Enter elements of second vector: ");

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

scanf("%d", &vector2[i]);

}

qsort(vector1, n, sizeof(int), compare);

qsort(vector2, n, sizeof(int), compare);

int scalarProduct = 0;

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

scalarProduct += vector1[i] \* vector2[i];

}

printf("Maximum scalar product: %d\n", scalarProduct);

return 0;

}

```

68. Find out triplets with a given sum

#include <stdio.h>

#include <stdbool.h>

void findTriplets(int arr[], int n, int sum) {

bool found = false;

for (int i = 0; i < n - 2; i++) {

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

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

if (arr[i] + arr[j] + arr[k] == sum) {

printf("Triplet: %d, %d, %d\n", arr[i], arr[j], arr[k]);

found = true;

}

}

}

}

if (!found) {

printf("No triplets found.\n");

}

}

int main() {

int n, sum;

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

scanf("%d", &n);

int arr[n];

printf("Enter the elements: ");

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

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

}

printf("Enter the sum value: ");

scanf("%d", &sum);

findTriplets(arr, n, sum);

return 0;

}

```

69. Find out number of even and odd elements in an array

#include <stdio.h>

int main() {

int n, evenCount = 0, oddCount = 0;

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

scanf("%d", &n);

int arr[n];

printf("Enter the elements: ");

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

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

}

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

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

evenCount++;

} else {

oddCount++;

}

}

printf("Number of even elements: %d\n", evenCount);

printf("Number of odd elements: %d\n", oddCount);

return 0;

}

```

70. Find out frequency of each element of an array

#include <stdio.h>

int main() {

int n;

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

scanf("%d", &n);

int arr[n], frequency[n];

printf("Enter the elements: ");

for (int i =

0; i < n; i++) {

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

frequency[i] = -1;

}

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

int count = 1;

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

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

count++;

frequency[j] = 0;

}

}

if (frequency[i] != 0) {

frequency[i] = count;

}

}

printf("Frequency of each element:\n");

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

if (frequency[i] != 0) {

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

}

}

return 0;

}

```

71. Find out symmetric pairs in an array

#include <stdio.h>

#include <stdbool.h>

typedef struct {

int first;

int second;

} Pair;

int main() {

int n;

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

scanf("%d", &n);

Pair pairs[n];

printf("Enter the pairs:\n");

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

scanf("%d %d", &pairs[i].first, &pairs[i].second);

}

printf("Symmetric pairs:\n");

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

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

if (pairs[i].first == pairs[j].second && pairs[i].second == pairs[j].first) {

printf("(%d, %d) and (%d, %d)\n", pairs[i].first, pairs[i].second, pairs[j].first, pairs[j].second);

}

}

}

return 0;

}

```

72. Identify arrays are disjoint or not

#include <stdio.h>

#include <stdbool.h>

bool areDisjoint(int arr1[], int n1, int arr2[], int n2) {

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

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

if (arr1[i] == arr2[j]) {

return false;

}

}

}

return true;

}

int main() {

int n1, n2;

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

scanf("%d", &n1);

int arr1[n1];

printf("Enter the elements of first array: ");

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

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

}

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

scanf("%d", &n2);

int arr2[n2];

printf("Enter the elements of second array: ");

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

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

}

if (areDisjoint(arr1, n1, arr2, n2)) {

printf("The arrays are disjoint.\n");

} else {

printf("The arrays are not disjoint.\n");

}

return 0;

}

```

73. Identify array is a subset of another array or not

#include <stdio.h>

#include <stdbool.h>

bool isSubset(int arr1[], int n1, int arr2[], int n2) {

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

bool found = false;

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

if (arr2[i] == arr1[j]) {

found = true;

break;

}

}

if (!found) {

return false;

}

}

return true;

}

int main() {

int n1, n2;

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

scanf("%d", &n1);

int arr1[n1];

printf("Enter the elements of first array: ");

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

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

}

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

scanf("%d", &n2);

int arr2[n2];

printf("Enter the elements of second array: ");

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

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

}

if (isSubset(arr1, n1, arr2, n2)) {

printf("The second array is a subset of the first array.\n");

} else {

printf("The second array is not a subset of the first array.\n");

}

return 0;

}

```

74. Sort elements of an array by frequency

#include <stdio.h>

#include <stdlib.h>

typedef struct {

int value;

int frequency;

} Element;

int compare(const void \*a, const void \*b) {

Element \*e1 = (Element \*)a;

Element \*e2 = (Element \*)b;

if (e1->frequency == e2->frequency) {

return e1->value - e2->value;

}

return e2->frequency - e1->frequency;

}

int main() {

int n;

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

scanf("%d", &n);

int arr[n], freq[n];

Element elements[n];

printf("Enter the elements: ");

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

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

freq[i] = -1;

}

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

int count = 1;

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

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

count++;

freq[j] = 0;

}

}

if (freq[i] != 0) {

freq[i] = count;

elements[i].value = arr[i];

elements[i].frequency = count;

}

}

qsort(elements, n, sizeof(Element), compare);

printf("Elements sorted by frequency: ");

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

if (freq[i] != 0) {

for (int j = 0; j < elements[i].frequency; j++) {

printf("%d ", elements[i].value);

}

}

}

printf("\n");

return 0;

}

```

75. Circular rotation of an array by K positions

#include <stdio.h>

void rotateArray(int arr[], int n, int k) {

int rotated[n];

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

rotated[(i + k) % n] = arr[i];

}

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

arr[i] = rotated[i];

}

}

int main() {

int n, k;

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

scanf("%d", &n);

int arr[n];

printf("Enter the elements: ");

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

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

}

printf("Enter the number of positions to rotate: ");

scanf("%d", &k);

rotateArray(arr, n, k);

printf("Array after rotation: ");

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

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

}

printf("\n");

return 0;

}

```

These programs cover a wide range of tasks involving arrays and numerical operations in C.

76. Merge two sorted arrays

#include <stdio.h>

void mergeArrays(int arr1[], int n1, int arr2[], int n2, int merged[]) {

int i = 0, j = 0, k = 0;

while (i < n1 && j < n2) {

if (arr1[i] < arr2[j]) {

merged[k++] = arr1[i++];

} else {

merged[k++] = arr2[j++];

}

}

while (i < n1) {

merged[k++] = arr1[i++];

}

while (j < n2) {

merged[k++] = arr2[j++];

}

}

int main() {

int n1, n2;

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

scanf("%d", &n1);

int arr1[n1];

printf("Enter the elements of first array: ");

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

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

}

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

scanf("%d", &n2);

int arr2[n2];

printf("Enter the elements of second array: ");

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

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

}

int merged[n1 + n2];

mergeArrays(arr1, n1, arr2, n2, merged);

printf("Merged array: ");

for (int i = 0; i < n1 + n2; i++) {

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

}

printf("\n");

return 0;

}

```

77. Find out sum of all odd frequency elements in an array

#include <stdio.h>

int main() {

int n;

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

scanf("%d", &n);

int arr[n], frequency[n];

printf("Enter the elements: ");

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

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

frequency[i] = -1;

}

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

int count = 1;

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

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

count++;

frequency[j] = 0;

}

}

if (frequency[i] != 0) {

frequency[i] = count;

}

}

int sum = 0;

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

if (frequency[i] % 2 != 0 && frequency[i] != 0) {

sum += arr[i] \* frequency[i];

}

}

printf("Sum of all odd frequency elements: %d\n", sum);

return 0;

}

```

78. Find out the median of two sorted arrays

#include <stdio.h>

int findMedianSortedArrays(int arr1[], int n1, int arr2[], int n2) {

int merged[n1 + n2];

int i = 0, j = 0, k = 0;

while (i < n1 && j < n2) {

if (arr1[i] < arr2[j]) {

merged[k++] = arr1[i++];

} else {

merged[k++] = arr2[j++];

}

}

while (i < n1) {

merged[k++] = arr1[i++];

}

while (j < n2) {

merged[k++] = arr2[j++];

}

if ((n1 + n2) % 2 == 0) {

return (merged[(n1 + n2) / 2 - 1] + merged[(n1 + n2) / 2]) / 2;

} else {

return merged[(n1 + n2) / 2];

}

}

int main() {

int n1, n2;

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

scanf("%d", &n1);

int arr1[n1];

printf("Enter the elements of first array: ");

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

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

}

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

scanf("%d", &n2);

int arr2[n2];

printf("Enter the elements of second array: ");

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

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

}

int median = findMedianSortedArrays(arr1, n1, arr2, n2);

printf("Median: %d\n", median);

return 0;

}

```

79. Solve 0-1 Knapsack problem

#include <stdio.h>

int max(int a, int b) {

return (a > b) ? a : b;

}

int knapSack(int W, int wt[], int val[], int n) {

int i, w;

int K[n + 1][W + 1];

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

for (w = 0; w <= W; w++) {

if (i == 0 || w == 0) {

K[i][w] = 0;

} else if (wt[i - 1] <= w) {

K[i][w] = max(val[i - 1] + K[i - 1][w - wt[i - 1]], K[i - 1][w]);

} else {

K[i][w] = K[i - 1][w];

}

}

}

return K[n][W];

}

int main() {

int n, W;

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

scanf("%d", &n);

int val[n], wt[n];

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

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

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

}

printf("Enter the weights of the items: ");

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

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

}

printf("Enter the maximum weight: ");

scanf("%d", &W);

printf("Maximum value in Knapsack: %d\n", knapSack(W, wt, val, n));

return 0;

}

```

80. Matrix operations (addition, subtraction, multiplication)

#include <stdio.h>

void inputMatrix(int matrix[10][10], int row, int col) {

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

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

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

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

}

}

}

void printMatrix(int matrix[10][10], int row, int col) {

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

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

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

}

printf("\n");

}

}

void addMatrix(int matrix1[10][10], int matrix2[10][10], int result[10][10], int row, int col) {

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

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

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

}

}

}

void subtractMatrix(int matrix1[10][10], int matrix2[10][10], int result[10][10], int row, int col) {

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

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

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

}

}

}

void multiplyMatrix(int matrix1[10][10], int matrix2[10][10], int result[10][10], int row1, int col1, int col2) {

for (int i = 0; i < row1; i++) {

for (int j = 0; j < col2; j++) {

result[i][j] = 0;

for (int k = 0; k < col1; k++) {

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

}

}

}

}

int main() {

int row1, col1, row2, col2;

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

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

scanf("%d %d", &row1, &col1);

inputMatrix(matrix1, row1, col1);

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

scanf("%d %d", &row2, &col2);

inputMatrix(matrix2, row2, col2);

printf

("Matrix addition:\n");

if (row1 == row2 && col1 == col2) {

addMatrix(matrix1, matrix2, result, row1, col1);

printMatrix(result, row1, col1);

} else {

printf("Matrices cannot be added.\n");

}

printf("Matrix subtraction:\n");

if (row1 == row2 && col1 == col2) {

subtractMatrix(matrix1, matrix2, result, row1, col1);

printMatrix(result, row1, col1);

} else {

printf("Matrices cannot be subtracted.\n");

}

printf("Matrix multiplication:\n");

if (col1 == row2) {

multiplyMatrix(matrix1, matrix2, result, row1, col1, col2);

printMatrix(result, row1, col2);

} else {

printf("Matrices cannot be multiplied.\n");

}

return 0;

}

```

81. Find out the transpose of a matrix

#include <stdio.h>

void transposeMatrix(int matrix[10][10], int transpose[10][10], int row, int col) {

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

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

transpose[j][i] = matrix[i][j];

}

}

}

int main() {

int row, col;

int matrix[10][10], transpose[10][10];

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

scanf("%d %d", &row, &col);

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

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

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

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

}

}

transposeMatrix(matrix, transpose, row, col);

printf("Transpose of the matrix:\n");

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

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

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

}

printf("\n");

}

return 0;

}

```

82. Identify upper triangular matrix or not

#include <stdio.h>

#include <stdbool.h>

bool isUpperTriangular(int matrix[10][10], int row, int col) {

if (row != col) {

return false;

}

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

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

if (matrix[i][j] != 0) {

return false;

}

}

}

return true;

}

int main() {

int row, col;

int matrix[10][10];

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

scanf("%d %d", &row, &col);

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

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

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

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

}

}

if (isUpperTriangular(matrix, row, col)) {

printf("The matrix is an upper triangular matrix.\n");

} else {

printf("The matrix is not an upper triangular matrix.\n");

}

return 0;

}

```

83. Identify lower triangular matrix or not

#include <stdio.h>

#include <stdbool.h>

bool isLowerTriangular(int matrix[10][10], int row, int col) {

if (row != col) {

return false;

}

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

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

if (matrix[i][j] != 0) {

return false;

}

}

}

return true;

}

int main() {

int row, col;

int matrix[10][10];

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

scanf("%d %d", &row, &col);

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

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

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

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

}

}

if (isLowerTriangular(matrix, row, col)) {

printf("The matrix is a lower triangular matrix.\n");

} else {

printf("The matrix is not a lower triangular matrix.\n");

}

return 0;

}

```

84. Find out the maximum element in a row of a matrix

#include <stdio.h>

int main() {

int row, col;

int matrix[10][10];

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

scanf("%d %d", &row, &col);

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

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

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

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

}

}

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

int max = matrix[i][0];

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

if (matrix[i][j] > max) {

max = matrix[i][j];

}

}

printf("Maximum element in row %d: %d\n", i + 1, max);

}

return 0;

}

```

85. Find out the maximum element in a column of a matrix

#include <stdio.h>

int main() {

int row, col;

int matrix[10][10];

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

scanf("%d %d", &row, &col);

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

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

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

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

}

}

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

int max = matrix[0][j];

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

if (matrix[i][j] > max) {

max = matrix[i][j];

}

}

printf("Maximum element in column %d: %d\n", j + 1, max);

}

return 0;

}

```

86. Find out sum of each row and column of a matrix

#include <stdio.h>

int main() {

int row, col;

int matrix[10][10];

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

scanf("%d %d", &row, &col);

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

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

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

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

}

}

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

int sum = 0;

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

sum += matrix[i][j];

}

printf("Sum of elements in row %d: %d\n", i + 1, sum);

}

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

int sum = 0;

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

sum += matrix[i][j];

}

printf("Sum of elements in column %d: %d\n", j + 1, sum);

}

return 0;

}

```

87. Find out sum of boundary elements of a matrix

#include <stdio.h>

int main() {

int row, col;

int matrix[10][10];

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

scanf("%d %d", &row, &col);

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

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

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

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

}

}

int sum = 0;

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

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

if (i == 0 || i == row - 1 || j == 0 || j == col - 1) {

sum += matrix[i][j];

}

}

}

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

return 0;

}

```

88. Print matrix in a spiral form

#include <stdio.h>

void spiralPrint(int row, int col, int matrix[10][10]) {

int i, row\_start = 0, col\_start = 0;

while (row\_start < row && col\_start < col

) {

for (i = col\_start; i < col; ++i) {

printf("%d ", matrix[row\_start][i]);

}

row\_start++;

for (i = row\_start; i < row; ++i) {

printf("%d ", matrix[i][col - 1]);

}

col--;

if (row\_start < row) {

for (i = col - 1; i >= col\_start; --i) {

printf("%d ", matrix[row - 1][i]);

}

row--;

}

if (col\_start < col) {

for (i = row - 1; i >= row\_start; --i) {

printf("%d ", matrix[i][col\_start]);

}

col\_start++;

}

}

}

int main() {

int row, col;

int matrix[10][10];

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

scanf("%d %d", &row, &col);

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

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

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

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

}

}

printf("Spiral form of the matrix: ");

spiralPrint(row, col, matrix);

printf("\n");

return 0;

}

```

89. Rotate the matrix by K times

#include <stdio.h>

void rotateMatrix(int matrix[10][10], int row, int col) {

int temp[row][col];

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

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

temp[i][j] = matrix[row - j - 1][i];

}

}

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

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

matrix[i][j] = temp[i][j];

}

}

}

int main() {

int row, col, k;

int matrix[10][10];

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

scanf("%d %d", &row, &col);

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

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

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

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

}

}

printf("Enter the number of times to rotate: ");

scanf("%d", &k);

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

rotateMatrix(matrix, row, col);

}

printf("Matrix after rotation:\n");

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

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

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

}

printf("\n");

}

return 0;

}

```

90. Rotate matrix by 90 degrees clockwise and anticlockwise

#include <stdio.h>

void rotate90Clockwise(int matrix[10][10], int row, int col) {

int temp[row][col];

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

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

temp[i][j] = matrix[row - j - 1][i];

}

}

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

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

matrix[i][j] = temp[i][j];

}

}

}

void rotate90AntiClockwise(int matrix[10][10], int row, int col) {

int temp[row][col];

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

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

temp[i][j] = matrix[j][col - i - 1];

}

}

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

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

matrix[i][j] = temp[i][j];

}

}

}

int main() {

int row, col;

int matrix[10][10];

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

scanf("%d %d", &row, &col);

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

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

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

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

}

}

printf("Matrix rotated by 90 degrees clockwise:\n");

rotate90Clockwise(matrix, row, col);

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

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

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

}

printf("\n");

}

printf("Matrix rotated by 90 degrees anticlockwise:\n");

rotate90AntiClockwise(matrix, row, col);

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

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

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

}

printf("\n");

}

return 0;

}

```

91. Basic string operations (copy, concat, compare)

#include <stdio.h>

#include <string.h>

void stringCopy(char destination[], char source[]) {

int i = 0;

while (source[i] != '\0') {

destination[i] = source[i];

i++;

}

destination[i] = '\0';

}

void stringConcat(char destination[], char source[]) {

int i = 0, j = 0;

while (destination[i] != '\0') {

i++;

}

while (source[j] != '\0') {

destination[i] = source[j];

i++;

j++;

}

destination[i] = '\0';

}

int stringCompare(char str1[], char str2[]) {

int i = 0;

while (str1[i] == str2[i]) {

if (str1[i] == '\0' || str2[i] == '\0') {

break;

}

i++;

}

if (str1[i] == '\0' && str2[i] == '\0') {

return 0;

} else {

return str1[i] - str2[i];

}

}

int main() {

char str1[100], str2[100], result[200];

printf("Enter first string: ");

scanf("%s", str1);

printf("Enter second string: ");

scanf("%s", str2);

// Copy

stringCopy(result, str1);

printf("Copy of first string: %s\n", result);

// Concat

stringConcat(result, str2);

printf("Concatenation of both strings: %s\n", result);

// Compare

int cmp = stringCompare(str1, str2);

if (cmp == 0) {

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

} else if (cmp > 0) {

printf("First string is greater.\n");

} else {

printf("Second string is greater.\n");

}

return 0;

}

```

92. Find length of the string without using strlen() function

#include <stdio.h>

int stringLength(char str[]) {

int length = 0;

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

length++;

}

return length;

}

int main() {

char str[100];

printf("Enter a string: ");

scanf("%s", str);

int length = stringLength(str);

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

return 0;

}

```

93. Toggle each character in a string

#include <stdio.h>

void toggleCase(char str[]) {

for (int i = 0; str[i] != '\0'; i++) {

if (str[i] >= 'a' && str[i] <= 'z') {

str[i] = str[i] - 'a' + 'A';

} else if (str[i] >= 'A' && str[i] <= 'Z') {

str[i] = str[i] - 'A' + 'a';

}

}

}

int main() {

char str[100];

printf("Enter a string: ");

scanf("%s", str);

toggleCase(str);

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

return 0;

}

```

94. here's a C program to count the number of vowels in a string:

#include <stdio.h>

int countVowels(char str[]) {

int count = 0;

for (int i = 0; str[i] != '\0'; i++) {

if (str[i] == 'a' || str[i] == 'e' || str[i] == 'i' || str[i] == 'o' || str[i] == 'u' ||

str[i] == 'A' || str[i] == 'E' || str[i] == 'I' || str[i] == 'O' || str[i] == 'U') {

count++;

}

}

return count;

}

int main() {

char str[100];

printf("Enter a string: ");

scanf("%s", str);

int vowels = countVowels(str);

printf("Number of vowels in the string: %d\n", vowels);

return 0;

}

```

This program prompts the user to input a string, then counts and prints the number of vowels in the entered string.

95. Remove vowels from a string

#include <stdio.h>

void removeVowels(char str[]) {

int i, j;

for (i = 0; str[i] != '\0'; ++i) {

while (str[i] == 'a' || str[i] == 'e' || str[i] == 'i' || str[i] == 'o' || str[i] == 'u' ||

str[i] == 'A' || str[i] == 'E' || str[i] == 'I' || str[i] == 'O' || str[i] == 'U') {

for (j = i; str[j] != '\0'; ++j) {

str[j] = str[j + 1];

}

str[j] = '\0';

}

}

}

int main() {

char str[100];

printf("Enter a string: ");

scanf("%s", str);

removeVowels(str);

printf("String after removing vowels: %s\n", str);

return 0;

}

```

96. Identify if a string is a palindrome or not

#include <stdio.h>

#include <stdbool.h>

#include <string.h>

bool isPalindrome(char str[]) {

int i = 0, j = strlen(str) - 1;

while (i < j) {

if (str[i] != str[j]) {

return false;

}

i++;

j--;

}

return true;

}

int main() {

char str[100];

printf("Enter a string: ");

scanf("%s", str);

if (isPalindrome(str)) {

printf("The string is a palindrome.\n");

} else {

printf("The string is not a palindrome.\n");

}

return 0;

}

```

97. Sort a string in alphabetical order

#include <stdio.h>

#include <string.h>

void sortString(char str[]) {

int i, j, len = strlen(str);

char temp;

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

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

if (str[i] > str[j]) {

temp = str[i];

str[i] = str[j];

str[j] = temp;

}

}

}

}

int main() {

char str[100];

printf("Enter a string: ");

scanf("%s", str);

sortString(str);

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

return 0;

}

```

98. Remove brackets from an algebraic expression

#include <stdio.h>

void removeBrackets(char str[]) {

int i, j;

for (i = 0; str[i] != '\0'; ++i) {

if (str[i] == '(' || str[i] == ')') {

for (j = i; str[j] != '\0'; ++j) {

str[j] = str[j + 1];

}

str[j] = '\0';

--i; // Decrement i to revisit the same index for potential next bracket removal

}

}

}

int main() {

char str[100];

printf("Enter an algebraic expression: ");

scanf("%s", str);

removeBrackets(str);

printf("Expression after removing brackets: %s\n", str);

return 0;

}

```

99. Remove spaces from a string

#include <stdio.h>

void removeSpaces(char str[]) {

int i, j;

for (i = 0; str[i] != '\0'; ++i) {

if (str[i] == ' ') {

for (j = i; str[j] != '\0'; ++j) {

str[j] = str[j + 1];

}

str[j] = '\0';

--i; // Decrement i to revisit the same index for potential next space removal

}

}

}

int main() {

char str[100];

printf("Enter a string: ");

scanf("%[^\n]s", str);

removeSpaces(str);

printf("String after removing spaces: %s\n", str);

return 0;

}

```

100. Count the sum of numbers in a string

#include <stdio.h>

int sumNumbers(char str[]) {

int i = 0, sum = 0;

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

if (str[i] >= '0' && str[i] <= '9') {

int num = 0;

while (str[i] >= '0' && str[i] <= '9') {

num = num \* 10 + (str[i] - '0');

i++;

}

sum += num;

} else {

i++;

}

}

return sum;

}

int main() {

char str[100];

printf("Enter a string: ");

scanf("%[^\n]s", str);

int sum = sumNumbers(str);

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

return 0;

}

```

Certainly! Here are solutions for the next set of problems:

101. Frequency of characters in a string

#include <stdio.h>

void countFrequency(char str[]) {

int frequency[256] = {0}; // Assuming ASCII characters

for (int i = 0; str[i] != '\0'; i++) {

frequency[(int)str[i]]++;

}

for (int i = 0; i < 256; i++) {

if (frequency[i] != 0) {

printf("'%c' occurs %d times\n", i, frequency[i]);

}

}

}

int main() {

char str[100];

printf("Enter a string: ");

scanf("%[^\n]s", str);

printf("Frequency of characters in the string:\n");

countFrequency(str);

return 0;

}

```

102. Replace substring in a string

#include <stdio.h>

#include <string.h>

void replaceSubstring(char str[], char oldSubstr[], char newSubstr[]) {

int i, j, k, oldLen = strlen(oldSubstr), newLen = strlen(newSubstr), found;

int strLen = strlen(str);

char temp[1000];

for (i = 0, k = 0; i < strLen; ) {

found = 1;

for (j = 0; j < oldLen; j++) {

if (str[i + j] != oldSubstr[j]) {

found = 0;

break;

}

}

if (found) {

for (j = 0; j < newLen; j++, k++) {

temp[k] = newSubstr[j];

}

i += oldLen;

} else {

temp[k++] = str[i++];

}

}

temp[k] = '\0';

strcpy(str, temp);

}

int main() {

char str[1000], oldSubstr[100], newSubstr[100];

printf("Enter a string: ");

scanf("%[^\n]s", str);

printf("Enter the substring to replace: ");

scanf("%s", oldSubstr);

printf("Enter the new substring: ");

scanf("%s", newSubstr);

replaceSubstring(str, oldSubstr, newSubstr);

printf("String after replacement: %s\n", str);

return 0;

}

```

103. Reverse a string

#include <stdio.h>

#include <string.h>

void reverseString(char str[]) {

int len = strlen(str);

for (int i = 0; i < len / 2; i++) {

char temp = str[i];

str[i] = str[len - i - 1];

str[len - i - 1] = temp;

}

}

int main() {

char str[100];

printf("Enter a string: ");

scanf("%[^\n]s", str);

reverseString(str);

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

return 0;

}

```

104. Add two complex numbers by passing structure to a function

#include <stdio.h>

typedef struct {

float real;

float imag;

} Complex;

Complex addComplex(Complex num1, Complex num2) {

Complex sum;

sum.real = num1.real + num2.real;

sum.imag = num1.imag + num2.imag;

return sum;

}

int main() {

Complex num1, num2, sum;

printf("For 1st complex number:\n");

printf("Enter real and imaginary parts respectively: ");

scanf("%f %f", &num1.real, &num1.imag);

printf("For 2nd complex number:\n");

printf("Enter real and imaginary parts respectively: ");

scanf("%f %f", &num2.real, &num2.imag);

sum = addComplex(num1, num2);

printf("Sum = %.2f + %.2fi\n", sum.real, sum.imag);

return 0;

}

```

105. Generate multiplication table

#include <stdio.h>

int main() {

int num;

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

scanf("%d", &num);

printf("Multiplication Table of %d:\n", num);

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

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

}

return 0;

}

```

106. Find the sum of natural numbers using recursion

#include <stdio.h>

int sumOfNaturalNumbers(int n) {

if (n == 0) {

return 0;

} else {

return n + sumOfNaturalNumbers(n - 1);

}

}

int main() {

int n;

printf("Enter a positive integer: ");

scanf("%d", &n);

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

return 0;

}

```

107. Reverse a sentence using recursion

#include <stdio.h>

#include <string.h>

void reverseSentence() {

char c;

scanf("%c", &c);

if (c != '\n') {

reverseSentence();

printf("%c", c);

}

}

int main() {

printf("Enter a sentence: ");

reverseSentence();

printf("\n");

return 0;

}

```

108. Calculate power using recursion

#include <stdio.h>

int power(int base, int exponent) {

if (exponent == 0) {

return 1;

} else {

return base \* power(base, exponent - 1);

}

}

int main() {

int base, exponent;

printf("Enter base: ");

scanf("%d", &base);

printf("Enter exponent: ");

scanf("%d", &exponent);

printf("%d^%d = %d\n", base, exponent, power(base, exponent));

return 0;

}

```

109. Access array elements using pointers

#include <stdio.h>

int main() {

int arr[] = {1, 2, 3, 4, 5};

int \*ptr = arr;

printf("Elements of the array:\n");

for (int i = 0; i < 5; i++) {

printf("%d ", \*(ptr + i));

}

printf("\n");

return 0;

}

```

Sure, here are solutions for problems 110, 111, and 112:

110. Add two distances (in inch-feet system) using structures

#include <stdio.h>

typedef struct {

int feet;

float inch;

} Distance;

Distance addDistances(Distance d1, Distance d2) {

Distance sum;

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

sum.inch = d1.inch + d2.inch;

if (sum.inch >= 12.0) {

sum.inch -= 12.0;

sum.feet++;

}

return sum;

}

int main() {

Distance d1, d2, sum;

printf("Enter first distance (feet inch): ");

scanf("%d %f", &d1.feet, &d1.inch);

printf("Enter second distance (feet inch): ");

scanf("%d %f", &d2.feet, &d2.inch);

sum = addDistances(d1, d2);

printf("Sum of distances: %d feet %.2f inches\n", sum.feet, sum.inch);

return 0;

}

```

111. Store information using structures with dynamically memory allocation

#include <stdio.h>

#include <stdlib.h>

typedef struct {

char name[50];

int age;

float salary;

} Employee;

int main() {

int n;

printf("Enter number of employees: ");

scanf("%d", &n);

Employee \*employees = (Employee \*)malloc(n \* sizeof(Employee));

if (employees == NULL) {

printf("Memory allocation failed!\n");

return 1;

}

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

printf("Enter name, age, and salary of employee %d: ", i + 1);

scanf("%s %d %f", employees[i].name, &employees[i].age, &employees[i].salary);

}

printf("\nEmployee Information:\n");

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

printf("Name: %s, Age: %d, Salary: %.2f\n", employees[i].name, employees[i].age, employees[i].salary);

}

free(employees);

return 0;

}

```

112. Calculate difference between two time periods using structure

#include <stdio.h>

typedef struct {

int hours;

int minutes;

int seconds;

} Time;

Time difference(Time t1, Time t2) {

Time diff;

int totalSeconds1 = t1.hours \* 3600 + t1.minutes \* 60 + t1.seconds;

int totalSeconds2 = t2.hours \* 3600 + t2.minutes \* 60 + t2.seconds;

int diffSeconds = totalSeconds1 - totalSeconds2;

diff.hours = diffSeconds / 3600;

diffSeconds %= 3600;

diff.minutes = diffSeconds / 60;

diff.seconds = diffSeconds % 60;

return diff;

}

int main() {

Time t1, t2, diff;

printf("Enter time 1 (hours minutes seconds): ");

scanf("%d %d %d", &t1.hours, &t1.minutes, &t1.seconds);

printf("Enter time 2 (hours minutes seconds): ");

scanf("%d %d %d", &t2.hours, &t2.minutes, &t2.seconds);

diff = difference(t1, t2);

printf("Difference: %d hours %d minutes %d seconds\n", diff.hours, diff.minutes, diff.seconds);

return 0;

}

```

These programs implement the functionalities as described for each problem.