Problems set

1 Basics of C Programming

1. Write a program to print "Hello, World!".
2. Write a program to take two integers as input and print their sum.
3. Write a program to find the area of a rectangle using its length and breadth.
4. Write a program to swap two numbers without using a temporary variable.
5. Write a program to check if a given number is even or odd.

2 Control Statements

1. Write a program to find the largest of three numbers using if-else.
2. Write a program to check if a number is positive, negative, or zero.
3. Write a program to calculate the factorial of a number using a for loop.
4. Write a program to print the first n natural numbers using a while loop.
5. Write a program to display the multiplication table of a given number.

3 Arrays

1. Write a program to store 5 integers in an array and print them.
2. Write a program to find the largest and smallest elements in an array.
3. Write a program to calculate the sum and average of elements in an array.
4. Write a program to reverse an array.
5. Write a program to merge two arrays into a single array.

4 Strings

1. Write a program to take a string as input and display its length.
2. Write a program to concatenate two strings without using library functions.
3. Write a program to check if a string is a palindrome.
4. Write a program to count the number of vowels and consonants in a string.
5. Write a program to find the frequency of a character in a string.

5 Functions

1. Write a function to find the GCD of two numbers.
2. Write a function to calculate the power of a number.

3. Write a recursive function to calculate the factorial of a number.
4. Write a function to check if a number is prime.
5. Write a function to find the sum of digits of a number.

6 POINTERS

1. Write a program to demonstrate pointer arithmetic.
2. Write a program to swap two numbers using pointers.
3. Write a program to find the length of a string using a pointer.
4. Write a program to reverse an array using pointers.
5. Write a program to dynamically allocate memory for an array using malloc.

7 STRUCTURES

1. Write a program to define a structure for a student (name, roll number, marks) and display the details.
2. Write a program to calculate the total and average marks of 5 students using an array of structures.
3. Write a program to store and display information of 5 employees (name, age, salary).
4. Write a program to demonstrate passing structures to functions.
5. Write a program to store a date (day, month, year) and display it in "dd-mm-yyyy" format.

8 FILE HANDLING

1. Write a program to create a file and write "Hello, World!" into it.
2. Write a program to read the contents of a file and display it on the screen.
3. Write a program to count the number of words in a file.
4. Write a program to append text to an existing file.
5. Write a program to copy the contents of one file to another.

9 SORTING AND SEARCHING

1. Write a program to implement the Bubble Sort algorithm.
2. Write a program to implement the Selection Sort algorithm.
3. Write a program to implement Linear Search.
4. Write a program to implement Binary Search.
5. Write a program to count the frequency of each element in an array.

10 ADVANCED ALGORITHMS

1. Write a program to generate the Fibonacci series up to n terms.
2. Write a program to find the sum of a geometric series.
3. Write a program to solve the Tower of Hanoi problem.
4. Write a program to find all prime numbers between two given numbers.
5. Write a program to implement the Sieve of Eratosthenes.

Solutions

1 Basics of C Programming

1.1 Hello, World! Program

```c
#include <stdio.h>
int main() {
    printf("Hello, World!\n");
    return 0;
}
```

1.2 Sum of Two Integers

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

1.3 Area of a Rectangle

```c
#include <stdio.h>
int main() {
    float length, breadth;
    printf("Enter length and breadth: ");
    scanf("%f %f", &length, &breadth);
    printf("Area: %.2f\n", length * breadth);
    return 0;
}
```

1.4 Swap Two Numbers without Temporary Variable

```c
#include <stdio.h>
int main() {
    int a, b;
    printf("Enter two numbers: ");
```

```
    scanf("%d %d", &a, &b);
    a = a + b;
    b = a - b;
    a = a - b;
    printf("After swapping: a = %d, b = %d\n", a, b);
    return 0;
}
```

1.5 Check if a Number is Even or Odd

```
#include <stdio.h>
int main() {
    int num;
    printf("Enter a number: ");
    scanf("%d", &num);
    if (num % 2 == 0)
        printf("%d is even\n", num);
    else
        printf("%d is odd\n", num);
    return 0;
}
```

2 Control Statements

2.1 Largest of Three Numbers using if-else

```
#include <stdio.h>
int main() {
    int a, b, c;
    printf("Enter three numbers: ");
    scanf("%d %d %d", &a, &b, &c);
    if (a > b && a > c)
        printf("%d is the largest\n", a);
    else if (b > c)
        printf("%d is the largest\n", b);
    else
        printf("%d is the largest\n", c);
    return 0;
}
```

2.2 Check if a Number is Positive, Negative, or Zero

```c
#include <stdio.h>
int main() {
    int num;
    printf("Enter a number: ");
    scanf("%d", &num);
    if (num > 0)
        printf("%d is positive\n", num);
    else if (num < 0)
        printf("%d is negative\n", num);
    else
        printf("The number is zero\n");
    return 0;
}
```

2.3 Factorial of a Number using a For Loop

```c
#include <stdio.h>
int main() {
    int num, factorial = 1;
    printf("Enter a number: ");
    scanf("%d", &num);
    for (int i = 1; i <= num; i++) {
        factorial *= i;
    }
    printf("Factorial: %d\n", factorial);
    return 0;
}
```

2.4 Print the First n Natural Numbers using a While Loop

```c
#include <stdio.h>
int main() {
    int n, i = 1;
    printf("Enter a number: ");
    scanf("%d", &n);
    while (i <= n) {
        printf("%d ", i);
        i++;
    }
    printf("\n");
    return 0;
}
```

2.5 Display Multiplication Table of a Given Number

```c
#include <stdio.h>
int main() {
    int num;
    printf("Enter a number: ");
    scanf("%d", &num);
    for (int i = 1; i <= 10; i++) {
        printf("%d x %d = %d\n", num, i, num * i);
    }
    return 0;
}
```

3 Arrays

3.1 Store 5 Integers in an Array and Print Them

```c
#include <stdio.h>
int main() {
    int arr[5];
    printf("Enter 5 integers: ");
    for (int i = 0; i < 5; i++) {
        scanf("%d", &arr[i]);
    }
    printf("Array elements: ");
    for (int i = 0; i < 5; i++) {
        printf("%d ", arr[i]);
    }
    printf("\n");
    return 0;
}
```

3.2 Find the Largest and Smallest Elements in an Array

```c
#include <stdio.h>
int main() {
    int arr[5], largest, smallest;
    printf("Enter 5 integers: ");
    for (int i = 0; i < 5; i++) {
        scanf("%d", &arr[i]);
    }
    largest = smallest = arr[0];
    for (int i = 1; i < 5; i++) {
```

```
        if (arr[i] > largest) largest = arr[i];
        if (arr[i] < smallest) smallest = arr[i];
    }
    printf("Largest: %d, Smallest: %d\n", largest, smallest);
    return 0;
}
```

3.3 Calculate the Sum and Average of Elements in an Array

```
#include <stdio.h>
int main() {
    int arr[5], sum = 0;
    float avg;
    printf("Enter 5 integers: ");
    for (int i = 0; i < 5; i++) {
        scanf("%d", &arr[i]);
    }
    for (int i = 0; i < 5; i++) {
        sum += arr[i];
    }
    avg = sum / 5.0;
    printf("Sum: %d, Average: %.2f\n", sum, avg);
    return 0;
}
```

3.4 Reverse an Array

```
#include <stdio.h>
int main() {
    int arr[5], temp;
    printf("Enter 5 integers: ");
    for (int i = 0; i < 5; i++) {
        scanf("%d", &arr[i]);
    }
    for (int i = 0; i < 5 / 2; i++) {
        temp = arr[i];
        arr[i] = arr[4 - i];
        arr[4 - i] = temp;
    }
    printf("Reversed array: ");
    for (int i = 0; i < 5; i++) {
        printf("%d ", arr[i]);
    }
    printf("\n");
```

```
    return 0;
}
```

3.5 Merge Two Arrays into a Single Array

```
#include <stdio.h>
int main() {
    int arr1[5], arr2[5], merged[10];
    printf("Enter 5 integers for array 1: ");
    for (int i = 0; i < 5; i++) {
        scanf("%d", &arr1[i]);
    }
    printf("Enter 5 integers for array 2: ");
    for (int i = 0; i < 5; i++) {
        scanf("%d", &arr2[i]);
    }
    for (int i = 0; i < 5; i++) {
        merged[i] = arr1[i];
        merged[i + 5] = arr2[i];
    }
    printf("Merged array: ");
    for (int i = 0; i < 10; i++) {
        printf("%d ", merged[i]);
    }
    printf("\n");
    return 0;
}
```

4 Strings

4.1 Take a String as Input and Display Its Length

```
#include <stdio.h>
#include <string.h>
int main() {
    char str[100];
    printf("Enter a string: ");
    fgets(str, sizeof(str), stdin);
    printf("Length of the string: %lu\n", strlen(str) - 1);  // Exclude the
newline character
    return 0;
}
```

4.2 Concatenate Two Strings Without Using Library Functions

```c
#include <stdio.h>
void concatenate(char* str1, char* str2) {
    while (*str1) str1++;  // Move to the end of the first string
    while (*str2) {
        *str1 = *str2;
        str1++;
        str2++;
    }
    *str1 = '\0';  // Null terminate the concatenated string
}
int main() {
    char str1[100], str2[50];
    printf("Enter first string: ");
    fgets(str1, sizeof(str1), stdin);
    printf("Enter second string: ");
    fgets(str2, sizeof(str2), stdin);
    concatenate(str1, str2);
    printf("Concatenated string: %s\n", str1);
    return 0;
}
```

4.3 Check if a String is a Palindrome

```c
#include <stdio.h>
#include <string.h>
int isPalindrome(char str[]) {
    int start = 0, end = strlen(str) - 1;
    while (start < end) {
        if (str[start] != str[end]) {
            return 0;
        }
        start++;
        end--;
    }
    return 1;
}
int main() {
    char str[100];
    printf("Enter a string: ");
    fgets(str, sizeof(str), stdin);
    if (isPalindrome(str)) {
        printf("The string is a palindrome\n");
    } else {
        printf("The string is not a palindrome\n");
```

```
    }
    return 0;
}
```

4.4 Count the Number of Vowels and Consonants in a String

```c
#include <stdio.h>
#include <ctype.h>
int main() {
    char str[100];
    int vowels = 0, consonants = 0;
    printf("Enter a string: ");
    fgets(str, sizeof(str), stdin);

    for (int i = 0; str[i] != '\0'; i++) {
        if (isalpha(str[i])) {
            char ch = tolower(str[i]);
            if (ch == 'a' || ch == 'e' || ch == 'i' || ch == 'o' || ch == 'u') {
                vowels++;
            } else {
                consonants++;
            }
        }
    }
    printf("Vowels: %d, Consonants: %d\n", vowels, consonants);
    return 0;
}
```

4.5 Find the Frequency of a Character in a String

```c
#include <stdio.h>
#include <string.h>
int main() {
    char str[100], ch;
    int count = 0;
    printf("Enter a string: ");
    fgets(str, sizeof(str), stdin);
    printf("Enter the character to find: ");
    scanf("%c", &ch);
    for (int i = 0; str[i] != '\0'; i++) {
        if (str[i] == ch) {
            count++;
        }
    }
    printf("The character '%c' appears %d times.\n", ch, count);
    return 0;
```

```
}
```

5 FUNCTIONS

5.1 FIND THE GCD OF TWO NUMBERS

```c
#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 a, b;
    printf("Enter two numbers: ");
    scanf("%d %d", &a, &b);
    printf("GCD: %d\n", gcd(a, b));
    return 0;
}
```

5.2 CALCULATE THE POWER OF A NUMBER

```c
#include <stdio.h>
int power(int base, int exp) {
    int result = 1;
    for (int i = 1; i <= exp; i++) {
        result *= base;
    }
    return result;
}
int main() {
    int base, exp;
    printf("Enter base and exponent: ");
    scanf("%d %d", &base, &exp);
    printf("%d^%d = %d\n", base, exp, power(base, exp));
    return 0;
}
```

5.3 RECURSIVE FUNCTION TO CALCULATE THE FACTORIAL OF A NUMBER

```c
#include <stdio.h>
int factorial(int n) {
```

```c
    if (n == 0 || n == 1) {
        return 1;
    }
    return n * factorial(n - 1);
}
int main() {
    int num;
    printf("Enter a number: ");
    scanf("%d", &num);
    printf("Factorial: %d\n", factorial(num));
    return 0;
}
```

5.4 Check if a Number is Prime

```c
#include <stdio.h>
int isPrime(int num) {
    if (num <= 1) return 0;
    for (int i = 2; i * i <= num; i++) {
        if (num % i == 0) return 0;
    }
    return 1;
}
int main() {
    int num;
    printf("Enter a number: ");
    scanf("%d", &num);
    if (isPrime(num)) {
        printf("%d is prime\n", num);
    } else {
        printf("%d is not prime\n", num);
    }
    return 0;
}
```

5.5 Find the Sum of Digits of a Number

```c
#include <stdio.h>
int sumOfDigits(int num) {
    int sum = 0;
    while (num != 0) {
        sum += num % 10;
        num /= 10;
    }
    return sum;
}
```

```c
int main() {
    int num;
    printf("Enter a number: ");
    scanf("%d", &num);
    printf("Sum of digits: %d\n", sumOfDigits(num));
    return 0;
}
```

6 POINTERS

6.1 POINTER ARITHMETIC

```c
#include <stdio.h>
int main() {
    int arr[] = {1, 2, 3, 4, 5};
    int* ptr = arr;

    printf("First element: %d\n", *ptr);
    ptr++;
    printf("Second element: %d\n", *ptr);
    ptr += 2;
    printf("Fourth element: %d\n", *ptr);

    return 0;
}
```

6.2 SWAP TWO NUMBERS USING POINTERS

```c
#include <stdio.h>
void swap(int* a, int* b) {
    *a = *a + *b;
    *b = *a - *b;
    *a = *a - *b;
}
int main() {
    int a = 5, b = 10;
    printf("Before swapping: a = %d, b = %d\n", a, b);
    swap(&a, &b);
    printf("After swapping: a = %d, b = %d\n", a, b);
    return 0;
}
```

6.3 FIND THE LENGTH OF A STRING USING A POINTER

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

```c
int stringLength(char* str) {
    int length = 0;
    while (*str != '\0') {
        length++;
        str++;
    }
    return length;
}
int main() {
    char str[100];
    printf("Enter a string: ");
    fgets(str, sizeof(str), stdin);
    printf("Length of the string: %d\n", stringLength(str));
    return 0;
}
```

6.4 Reverse an Array Using Pointers

```c
#include <stdio.h>
void reverseArray(int* arr, int size) {
    int* start = arr;
    int* end = arr + size - 1;
    while (start < end) {
        int temp = *start;
        *start = *end;
        *end = temp;
        start++;
        end--;
    }
}
int main() {
    int arr[5] = {1, 2, 3, 4, 5};
    printf("Before reversing: ");
    for (int i = 0; i < 5; i++) {
        printf("%d ", arr[i]);
    }
    printf("\n");
    reverseArray(arr, 5);
    printf("After reversing: ");
    for (int i = 0; i < 5; i++) {
        printf("%d ", arr[i]);
    }
    printf("\n");
    return 0;
}
```

6.5 Dynamically Allocate Memory for an Array Using malloc

```c
#include <stdio.h>
#include <stdlib.h>
int main() {
    int size;
    printf("Enter the size of the array: ");
    scanf("%d", &size);
    int* arr = (int*)malloc(size * sizeof(int));

    if (arr == NULL) {
        printf("Memory allocation failed.\n");
        return 1;
    }

    printf("Enter elements: ");
    for (int i = 0; i < size; i++) {
        scanf("%d", &arr[i]);
    }

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

    free(arr);  // Don't forget to free the allocated memory
    return 0;
}
```

7 Structures

7.1 Store and Display Student Details Using Structure

```c
#include <stdio.h>
struct Student {
    char name[50];
    int rollNumber;
    float marks;
};
int main() {
    struct Student student;
    printf("Enter student name: ");
    fgets(student.name, sizeof(student.name), stdin);
    printf("Enter roll number: ");
```

```
    scanf("%d", &student.rollNumber);
    printf("Enter marks: ");
    scanf("%f", &student.marks);

    printf("\nStudent Details:\n");
    printf("Name: %s", student.name);
    printf("Roll Number: %d\n", student.rollNumber);
    printf("Marks: %.2f\n", student.marks);
    return 0;
}
```

7.2 Calculate Total and Average Marks of 5 Students Using an Array of Structures

```
#include <stdio.h>
struct Student {
    char name[50];
    int marks[5];
};
int main() {
    struct Student students[5];
    int totalMarks, sum;
    float average;

    for (int i = 0; i < 5; i++) {
        printf("Enter details for student %d\n", i+1);
        printf("Enter name: ");
        fgets(students[i].name, sizeof(students[i].name), stdin);
        printf("Enter marks for 5 subjects: ");
        sum = 0;
        for (int j = 0; j < 5; j++) {
            scanf("%d", &students[i].marks[j]);
            sum += students[i].marks[j];
        }
        average = sum / 5.0;
        totalMarks = sum;
        printf("Total Marks: %d, Average Marks: %.2f\n", totalMarks, average);
    }
    return 0;
}
```

7.3 Store and Display Information of 5 Employees (Name, Age, Salary)

```
#include <stdio.h>
struct Employee {
    char name[50];
    int age;
```

```
    float salary;
};
int main() {
    struct Employee employees[5];

    for (int i = 0; i < 5; i++) {
        printf("Enter details for employee %d\n", i+1);
        printf("Enter name: ");
        fgets(employees[i].name, sizeof(employees[i].name), stdin);
        printf("Enter age: ");
        scanf("%d", &employees[i].age);
        printf("Enter salary: ");
        scanf("%f", &employees[i].salary);
        getchar();  // To capture newline character after entering salary
    }

    printf("\nEmployee Details:\n");
    for (int i = 0; i < 5; i++) {
        printf("\nEmployee %d:\n", i+1);
        printf("Name: %s", employees[i].name);
        printf("Age: %d\n", employees[i].age);
        printf("Salary: %.2f\n", employees[i].salary);
    }
    return 0;
}
```

7.4 Passing Structures to Functions

```
#include <stdio.h>
struct Rectangle {
    int length;
    int breadth;
};
int calculateArea(struct Rectangle rect) {
    return rect.length * rect.breadth;
}
int main() {
    struct Rectangle rect = {10, 5};
    printf("Area of rectangle: %d\n", calculateArea(rect));
    return 0;
}
```

7.5 Store and Display a Date in "DD-MM-YYYY" Format

```
#include <stdio.h>
struct Date {
```

```
    int day;
    int month;
    int year;
};
int main() {
    struct Date date;
    printf("Enter day: ");
    scanf("%d", &date.day);
    printf("Enter month: ");
    scanf("%d", &date.month);
    printf("Enter year: ");
    scanf("%d", &date.year);

    printf("The date is: %02d-%02d-%04d\n", date.day, date.month, date.year);
    return 0;
}
```

8 File Handling

8.1 Create a File and Write "Hello, World!"

```
#include <stdio.h>
int main() {
    FILE* file = fopen("hello.txt", "w");
    if (file == NULL) {
        printf("File could not be created.\n");
        return 1;
    }
    fprintf(file, "Hello, World!\n");
    fclose(file);
    printf("File created and text written successfully.\n");
    return 0;
}
```

8.2 Read Contents of a File and Display on Screen

```
#include <stdio.h>
int main() {
    FILE* file = fopen("hello.txt", "r");
    if (file == NULL) {
        printf("File not found.\n");
        return 1;
    }
    char ch;
    while ((ch = fgetc(file)) != EOF) {
```

```
        putchar(ch);
    }
    fclose(file);
    return 0;
}
```

8.3 Count the Number of Words in a File

```
#include <stdio.h>
#include <ctype.h>
int countWords(FILE* file) {
    int count = 0;
    char ch;
    int inWord = 0;

    while ((ch = fgetc(file)) != EOF) {
        if (isalpha(ch)) {
            if (!inWord) {
                inWord = 1;
                count++;
            }
        } else {
            inWord = 0;
        }
    }
    return count;
}
int main() {
    FILE* file = fopen("sample.txt", "r");
    if (file == NULL) {
        printf("File not found.\n");
        return 1;
    }
    int wordCount = countWords(file);
    fclose(file);
    printf("Number of words in file: %d\n", wordCount);
    return 0;
}
```

8.4 Append Text to an Existing File

```
#include <stdio.h>
int main() {
    FILE* file = fopen("hello.txt", "a");
    if (file == NULL) {
        printf("File not found.\n");
```

```
        return 1;
    }
    fprintf(file, "Appending new content!\n");
    fclose(file);
    printf("Text appended successfully.\n");
    return 0;
}
```

8.5 Copy Contents of One File to Another

```
#include <stdio.h>
int main() {
    FILE *source, *destination;
    char ch;
    source = fopen("source.txt", "r");
    if (source == NULL) {
        printf("Source file not found.\n");
        return 1;
    }
    destination = fopen("destination.txt", "w");
    if (destination == NULL) {
        printf("Unable to open destination file.\n");
        fclose(source);
        return 1;
    }

    while ((ch = fgetc(source)) != EOF) {
        fputc(ch, destination);
    }

    fclose(source);
    fclose(destination);
    printf("File copied successfully.\n");
    return 0;
}
```

9 Sorting and Searching

9.1 Implement Bubble Sort

```
#include <stdio.h>
void bubbleSort(int arr[], int size) {
    for (int i = 0; i < size - 1; i++) {
        for (int j = 0; j < size - i - 1; j++) {
            if (arr[j] > arr[j + 1]) {
```

```
                int temp = arr[j];
                arr[j] = arr[j + 1];
                arr[j + 1] = temp;
            }
        }
    }
}
int main() {
    int arr[] = {5, 1, 4, 2, 8};
    int size = sizeof(arr) / sizeof(arr[0]);
    bubbleSort(arr, size);
    printf("Sorted array: ");
    for (int i = 0; i < size; i++) {
        printf("%d ", arr[i]);
    }
    printf("\n");
    return 0;
}
```

9.2 Implement Selection Sort

```
#include <stdio.h>
void selectionSort(int arr[], int size) {
    for (int i = 0; i < size - 1; i++) {
        int minIdx = i;
        for (int j = i + 1; j < size; j++) {
            if (arr[j] < arr[minIdx]) {
                minIdx = j;
            }
        }
        if (minIdx != i) {
            int temp = arr[i];
            arr[i] = arr[minIdx];
            arr[minIdx] = temp;
        }
    }
}
int main() {
    int arr[] = {5, 1, 4, 2, 8};
    int size = sizeof(arr) / sizeof(arr[0]);
    selectionSort(arr, size);
    printf("Sorted array: ");
    for (int i = 0; i < size; i++) {
        printf("%d ", arr[i]);
    }
```

```
    printf("\n");
    return 0;
}
```

9.3 Implement Linear Search

```
#include <stdio.h>
int linearSearch(int arr[], int size, int target) {
    for (int i = 0; i < size; i++) {
        if (arr[i] == target) {
            return i;
        }
    }
    return -1;  // Not found
}
int main() {
    int arr[] = {5, 1, 4, 2, 8};
    int target = 4;
    int size = sizeof(arr) / sizeof(arr[0]);
    int index = linearSearch(arr, size, target);
    if (index != -1) {
        printf("Element found at index: %d\n", index);
    } else {
        printf("Element not found\n");
    }
    return 0;
}
```

9.4 Implement Binary Search

```
#include <stdio.h>
int binarySearch(int arr[], int size, int target) {
    int left = 0, right = size - 1;
    while (left <= right) {
        int mid = left + (right - left) / 2;
        if (arr[mid] == target) {
            return mid;
        }
        if (arr[mid] < target) {
            left = mid + 1;
        } else {
            right = mid - 1;
        }
    }
    return -1;  // Not found
```

```
}
int main() {
    int arr[] = {1, 2, 4, 5, 8};
    int target = 4;
    int size = sizeof(arr) / sizeof(arr[0]);
    int index = binarySearch(arr, size, target);
    if (index != -1) {
        printf("Element found at index: %d\n", index);
    } else {
        printf("Element not found\n");
    }
    return 0;
}
```

9.5 Find the Frequency of Each Element in an Array

```
#include <stdio.h>
int main() {
    int arr[] = {5, 1, 4, 2, 8, 1, 4};
    int size = sizeof(arr) / sizeof(arr[0]);
    int freq[size];

    for (int i = 0; i < size; i++) {
        freq[i] = -1;
    }

    for (int i = 0; i < size; i++) {
        if (freq[i] == -1) {
            int count = 1;
            for (int j = i + 1; j < size; j++) {
                if (arr[i] == arr[j]) {
                    count++;
                    freq[j] = 0;  // Mark duplicate elements
                }
            }
            freq[i] = count;
            printf("Element %d occurs %d times\n", arr[i], count);
        }
    }
    return 0;
}
```

10 ADVANCED ALGORITHMS

10.1 SOLVE THE N-QUEENS PROBLEM (BACKTRACKING)

```c
#include <stdio.h>
#define N 4
int board[N][N];

int isSafe(int row, int col) {
    for (int i = 0; i < col; i++) {
        if (board[row][i] == 1) {
            return 0;  // Check if the queen is in the same row
        }
    }
    for (int i = row, j = col; i >= 0 && j >= 0; i--, j--) {
        if (board[i][j] == 1) {
            return 0;  // Check upper diagonal
        }
    }
    for (int i = row, j = col; j >= 0 && i < N; i++, j--) {
        if (board[i][j] == 1) {
            return 0;  // Check lower diagonal
        }
    }
    return 1;  // Safe to place the queen
}

int solveNQueens(int col) {
    if (col >= N) {
        return 1;  // All queens are placed
    }

    for (int i = 0; i < N; i++) {
        if (isSafe(i, col)) {
            board[i][col] = 1;
            if (solveNQueens(col + 1)) {
                return 1;
            }
            board[i][col] = 0;  // Backtrack
        }
    }
    return 0;  // No safe position found
}
```

```
void printBoard() {
    for (int i = 0; i < N; i++) {
        for (int j = 0; j < N; j++) {
            printf("%d ", board[i][j]);
        }
        printf("\n");
    }
}

int main() {
    for (int i = 0; i < N; i++) {
        for (int j = 0; j < N; j++) {
            board[i][j] = 0;  // Initialize board to 0
        }
    }

    if (solveNQueens(0)) {
        printf("Solution to N-Queens Problem:\n");
        printBoard();
    } else {
        printf("Solution does not exist\n");
    }
    return 0;
}
```

10.2 Generate the Fibonacci Series up to n Terms

```
#include <stdio.h>
void generateFibonacci(int n) {
    int a = 0, b = 1, next;
    printf("Fibonacci Series: ");
    for (int i = 0; i < n; i++) {
        printf("%d ", a);
        next = a + b;
        a = b;
        b = next;
    }
    printf("\n");
}
int main() {
    int n;
    printf("Enter the number of terms: ");
    scanf("%d", &n);
    generateFibonacci(n);
    return 0;
```

```
}
```

10.3 Solve the Tower of Hanoi Problem

```c
#include <stdio.h>
void towerOfHanoi(int n, char from, char to, char aux) {
    if (n == 1) {
        printf("Move disk 1 from %c to %c\n", from, to);
        return;
    }
    towerOfHanoi(n - 1, from, aux, to);
    printf("Move disk %d from %c to %c\n", n, from, to);
    towerOfHanoi(n - 1, aux, to, from);
}
int main() {
    int n;
    printf("Enter number of disks: ");
    scanf("%d", &n);
    towerOfHanoi(n, 'A', 'C', 'B');
    return 0;
}
```

10.4 Find All Prime Numbers Between Two Given Numbers

```c
#include <stdio.h>
int isPrime(int n) {
    if (n <= 1) return 0;
    for (int i = 2; i * i <= n; i++) {
        if (n % i == 0) return 0;
    }
    return 1;
}
void printPrimes(int start, int end) {
    printf("Prime numbers between %d and %d: ", start, end);
    for (int i = start; i <= end; i++) {
        if (isPrime(i)) {
            printf("%d ", i);
        }
    }
    printf("\n");
}
int main() {
    int start, end;
    printf("Enter range (start and end): ");
    scanf("%d %d", &start, &end);
    printPrimes(start, end);
```

```
    return 0;
}
```

10.5 Implement the Sieve of Eratosthenes

```
#include <stdio.h>
#define MAX 100
void sieveOfEratosthenes(int n) {
    int prime[MAX];
    for (int i = 0; i <= n; i++) {
        prime[i] = 1;
    }
    prime[0] = prime[1] = 0;

    for (int i = 2; i * i <= n; i++) {
        if (prime[i]) {
            for (int j = i * i; j <= n; j += i) {
                prime[j] = 0;
            }
        }
    }

    printf("Prime numbers up to %d: ", n);
    for (int i = 2; i <= n; i++) {
        if (prime[i]) {
            printf("%d ", i);
        }
    }
    printf("\n");
}
int main() {
    int n;
    printf("Enter a number: ");
    scanf("%d", &n);
    sieveOfEratosthenes(n);
    return 0;
}
```