1+10 Beginner Level Programs

1. Even or Odd Number
2. Find Largest of Three Numbers
3. Sum of Natural Numbers (loops & recursion)
4. Reverse a Number
5. Palindrome Number/String
6. Prime Number Check
7. Factorial of a Number
8. Fibonacci Series Generation
9. Count Digits in a Number
10. Swap Two Numbers (with temp / without temp)

10+20 Intermediate Level Programs

11. Armstrong Number Check
12. Reverse an Array
13. Find the Second Largest Element in Array
14. Find GCD / LCM of Two Numbers
15. Array Rotation (Left/right)
16. Remove Duplicates from Array

17. Count Frequency of Each Element in Array

18. Check for Array Palindrome

19. Sorting an Array (Bubble Sort, Selection Sort)

20. Merge Two Arrays

21–30 Advanced Level Programs

21. Matrix Multiplication / Transpose / Diagonal Sum

22. Check Balanced Parentheses Using Stack

23. Balanced Binary Tree (Height balanced)

24. Find Cycle in Linked List

25. Lowest Common Ancestor in Binary Tree

26. Find Subarray with Given Sum (Sliding Window)

27. Regular Expression Pattern Matching

28. Huffman Coding Implementation (Character Frequency Compression)

29. N-Queens Problem (Backtracking)

30. Trapping Rainwater Problem

Program 1: Even or Odd Number

◈C Language:

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


◈Java:

```
import java.util.Scanner;
public class EvenOdd {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a number: ");
        int n = sc.nextInt();
        if (n % 2 == 0)
            System.out.println(n + " is Even");
        else
            System.out.println(n + " is Odd");
        sc.close();
    }
```

```
}
```

◈Python:

```
n = int(input("Enter a number: "))
if n % 2 == 0:
    print(f"{n} is Even")
else:
    print(f"{n} is Odd")
```

◈DSA with Java Version (Using Method & Input Validation):

```
import java.util.Scanner;

public class EvenOddDSA {
    // Method to check even or odd
    static String checkEvenOdd(int num) {
        return (num % 2 == 0) ? "Even" : "Odd";
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a number: ");
        while(!sc.hasNextInt()) {   // input validation
            System.out.println("Invalid input! Enter an integer.");
            sc.next();
        }
        int n = sc.nextInt();
```

```
        System.out.println(n + " is " + checkEvenOdd(n));
        sc.close();
    }
}
```

✅ Explanation (DSA version):

Method (checkEvenOdd) created to separate logic from main (good OOP practice).

Input validation added using Scanner.

This mirrors real DSA/Java coding style.

🟢 Program 2: Find Largest of Three Numbers

🔷C Language:


```
#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 >= a && b >= c)
        printf("%d is the largest\n", b);
    else
        printf("%d is the largest\n", c);
```

```
    return 0;
}
```

◈Java:

```
import java.util.Scanner;

public class LargestOfThree {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter three numbers: ");
        int a = sc.nextInt();
        int b = sc.nextInt();
        int c = sc.nextInt();

        if(a >= b && a >= c)
            System.out.println(a + " is the largest");
        else if(b >= a && b >= c)
            System.out.println(b + " is the largest");
        else
            System.out.println(c + " is the largest");

        sc.close();
    }
}
```

◈Python:

```
a = int(input("Enter first number: "))
```

```
b = int(input("Enter second number: "))
c = int(input("Enter third number: "))

if a >= b and a >= c:
    print(f"{a} is the largest")
elif b >= a and b >= c:
    print(f"{b} is the largest")
else:
    print(f"{c} is the largest")
```

◈DSA with Java Version (Using Method & Ternary Operator):

```
import java.util.Scanner;
```



```
public class LargestOfThreeDSA {
    // Method to find largest using ternary operator
    static int findLargest(int a, int b, int c) {
        return (a > b) ? ((a > c) ? a : c) : ((b > c) ? b : c);
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter three numbers: ");
        while(!sc.hasNextInt()) { // input validation for first number
            System.out.println("Invalid input! Enter an integer.");
            sc.next();
        }
        int a = sc.nextInt();
```

```
        int b = sc.nextInt();
        int c = sc.nextInt();

        int largest = findLargest(a, b, c);
        System.out.println(largest + " is the largest");

        sc.close();
    }
}
```

✅ Explanation (DSA version):

Method findLargest created — clean separation of logic.

Ternary operator used — concise comparison logic.

Input validation included for robustness.

Mirrors typical DSA/Java style for coding interviews.

3️⃣ Program 3: Sum of Natural Numbers (Loop + Recursion)

🔷 C Language (Loop):

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

```
        sum += i;

    printf("Sum of first %d natural numbers is %d\n", n, sum);
    return 0;
}
```

◈C Language (Recursion):

```
#include <stdio.h>

int sumNatural(int n) {
    if(n == 0) return 0;
    return n + sumNatural(n - 1);
}

int main() {
    int n;
    printf("Enter a number: ");
    scanf("%d", &n);
    printf("Sum of first %d natural numbers is %d\n", n, sumNatural(n));
    return 0;
}
```

◈Java (Loop):

```
import java.util.Scanner;

public class SumNatural {
    public static void main(String[] args) {
```

```
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a number: ");
        int n = sc.nextInt();
        int sum = 0;
        for(int i = 1; i <= n; i++)
            sum += i;
        System.out.println("Sum of first " + n + " natural numbers is " +
sum);
        sc.close();
    }
}
```

◈Java (Recursion):


```
import java.util.Scanner;

public class SumNaturalRecursion {

    static int sumNatural(int n) {
        if(n == 0) return 0;
        return n + sumNatural(n - 1);
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a number: ");
        int n = sc.nextInt();
        System.out.println("Sum of first " + n + " natural numbers is " +
sumNatural(n));
        sc.close();
```

```
    }
}
```

◈Python (Loop + Recursion):

```
n = int(input("Enter a number: "))

# Loop
sum_loop = sum(range(1, n+1))
print("Sum (Loop):", sum_loop)

# Recursion
def sum_recursive(n):
    if n == 0:
        return 0
    return n + sum_recursive(n-1)

print("Sum (Recursion):", sum_recursive(n))
```


◈DSA Java (Using Method + Recursion + Validation):

```
import java.util.Scanner;

public class SumNaturalDSA {

    static int sumRecursive(int n) {
        if(n == 0) return 0;
        return n + sumRecursive(n-1);
    }
```

```
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a number: ");
        while(!sc.hasNextInt()) {
            System.out.println("Enter a valid integer!");
            sc.next();
        }
        int n = sc.nextInt();
        System.out.println("Sum of first " + n + " natural numbers is " +
sumRecursive(n));
        sc.close();
    }
}
```

4⃣ Program 4: Reverse a Number

🔷 C Language:

```
#include <stdio.h>
int main() {
    int n, rev = 0;
    printf("Enter a number: ");
    scanf("%d", &n);

    while(n != 0) {
```

```
        rev = rev * 10 + n % 10;
        n /= 10;
    }
    printf("Reversed number is %d\n", rev);
    return 0;
}
```

◈Java:

```
import java.util.Scanner;
public class ReverseNumber {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a number: ");
        int n = sc.nextInt();
        int rev = 0;
        while(n != 0) {
            rev = rev * 10 + n % 10;
            n /= 10;
        }
        System.out.println("Reversed number is " + rev);
        sc.close();
    }
}
```


◈Python:

```
n = int(input("Enter a number: "))
rev = 0
```

```
temp = n
while temp != 0:
    rev = rev * 10 + temp % 10
    temp //= 10
print("Reversed number is", rev)
```

◈DSA Java (Using Method):

```
import java.util.Scanner;

public class ReverseNumberDSA {

    static int reverseNumber(int n) {
        int rev = 0;
        while(n != 0) {
            rev = rev * 10 + n % 10;
            n /= 10;
        }
        return rev;
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a number: ");
        int n = sc.nextInt();
        System.out.println("Reversed number is " + reverseNumber(n));
        sc.close();
    }
}
```

5️⃣ Program 5: Palindrome Number/String

🔷 C Language (Number):

```
#include <stdio.h>
int main() {
    int n, rev = 0, temp;
    printf("Enter a number: ");
    scanf("%d", &n);
    temp = n;

    while(temp != 0) {
        rev = rev * 10 + temp % 10;
        temp /= 10;
    }

    if(n == rev)
        printf("%d is a palindrome\n", n);
    else
        printf("%d is not a palindrome\n", n);

    return 0;
}
```


🔷 Java:

```
import java.util.Scanner;
public class Palindrome {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a number: ");
        int n = sc.nextInt();
        int temp = n, rev = 0;

        while(temp != 0) {
            rev = rev * 10 + temp % 10;
            temp /= 10;
        }

        if(n == rev)
            System.out.println(n + " is a palindrome");
        else
            System.out.println(n + " is not a palindrome");

        sc.close();
    }
}
```


🔷Python:

```
n = int(input("Enter a number: "))
rev = int(str(n)[::-1])
if n == rev:
    print(f"{n} is a palindrome")
```

```
else:
    print(f"{n} is not a palindrome")
```

◈DSA Java (Method + String Support):

```
import java.util.Scanner;

public class PalindromeDSA {

    static boolean isPalindrome(int n) {
        int rev = 0, temp = n;
        while(temp != 0) {
            rev = rev * 10 + temp % 10;
            temp /= 10;
        }
        return n == rev;
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a number: ");
        int n = sc.nextInt();
        System.out.println(n + (isPalindrome(n) ? " is a palindrome" : " is
not a palindrome"));
        sc.close();
    }
}
```

6⃣ Program 6: Prime Number Check

🔷 C Language:

```
#include <stdio.h>
int main() {
    int n, i, flag = 0;
    printf("Enter a number: ");
    scanf("%d", &n);

    if(n <= 1) flag = 1; // not prime

    for(i = 2; i <= n/2; i++) {
        if(n % i == 0) {
            flag = 1;
            break;
        }
    }

    if(flag == 0)
        printf("%d is prime\n", n);
    else
        printf("%d is not prime\n", n);

    return 0;
}
```


🔷 Java:

```
import java.util.Scanner;
public class PrimeCheck {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a number: ");
        int n = sc.nextInt();
        boolean isPrime = true;

        if(n <= 1) isPrime = false;
        for(int i = 2; i <= n/2; i++) {
            if(n % i == 0) {
                isPrime = false;
                break;
            }
        }

        System.out.println(n + (isPrime ? " is prime" : " is not prime"));
        sc.close();
    }
}
```


◈Python:

```
n = int(input("Enter a number: "))
is_prime = True
if n <= 1:
    is_prime = False
for i in range(2, n//2 + 1):
```

```
    if n % i == 0:
        is_prime = False
        break
print(f"{n} is {'prime' if is_prime else 'not prime'}")
```

◈DSA Java (Method + Optimized up to √n):

```
import java.util.Scanner;

public class PrimeCheckDSA {

    static boolean isPrime(int n) {
        if(n <= 1) return false;
        for(int i = 2; i*i <= n; i++) {
            if(n % i == 0) return false;
        }
        return true;
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a number: ");
        int n = sc.nextInt();
        System.out.println(n + (isPrime(n) ? " is prime" : " is not prime"));
        sc.close();
    }
}
```

7️⃣ Program 7: Factorial of a Number

🔷 C Language (Loop):

```
#include <stdio.h>
int main() {
    int n;
    unsigned long long fact = 1;
    printf("Enter a number: ");
    scanf("%d", &n);

    for(int i = 1; i <= n; i++)
        fact *= i;

    printf("Factorial of %d is %llu\n", n, fact);
    return 0;
}
```


🔷 C Language (Recursion):

```
#include <stdio.h>
unsigned long long factorial(int n) {
    if(n == 0) return 1;
    return n * factorial(n-1);
}
int main() {
    int n;
```

```
    printf("Enter a number: ");
    scanf("%d", &n);
    printf("Factorial of %d is %llu\n", n, factorial(n));
    return 0;
}
```

◈Java (Loop):

```
import java.util.Scanner;
public class Factorial {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a number: ");
        int n = sc.nextInt();
        long fact = 1;
        for(int i = 1; i <= n; i++)
            fact *= i;
        System.out.println("Factorial of " + n + " is " + fact);
        sc.close();
    }
}
```


◈Java (Recursion):

```
import java.util.Scanner;
public class FactorialRecursion {

    static long factorial(int n) {
        if(n == 0) return 1;
```

```
        return n * factorial(n-1);
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a number: ");
        int n = sc.nextInt();
        System.out.println("Factorial of " + n + " is " + factorial(n));
        sc.close();
    }
}
```

◈Python (Loop + Recursion):


```
n = int(input("Enter a number: "))

# Loop
fact = 1
for i in range(1, n+1):
    fact *= i
print("Factorial (Loop):", fact)

# Recursion
def factorial(n):
    if n == 0:
        return 1
    return n * factorial(n-1)

print("Factorial (Recursion):", factorial(n))
```

◈DSA Java (Method + Recursion + Validation):

```
import java.util.Scanner;

public class FactorialDSA {

    static long factorial(int n) {
        if(n == 0) return 1;
        return n * factorial(n-1);
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a number: ");
        while(!sc.hasNextInt()) {
            System.out.println("Enter a valid integer!");
            sc.next();
        }
        int n = sc.nextInt();
        System.out.println("Factorial of " + n + " is " + factorial(n));
        sc.close();
    }
}
```

8⃣Program 8: Fibonacci Series Generation

◈C Language (Loop):

```
#include <stdio.h>
int main() {
    int n, t1 = 0, t2 = 1, next;
    printf("Enter number of terms: ");
    scanf("%d", &n);

    printf("Fibonacci Series: ");
    for(int i = 1; i <= n; i++) {
        printf("%d ", t1);
        next = t1 + t2;
        t1 = t2;
        t2 = next;
    }
    printf("\n");
    return 0;
}
```


◈C Language (Recursion):

```
#include <stdio.h>
int fibonacci(int n) {
    if(n == 0) return 0;
    if(n == 1) return 1;
    return fibonacci(n-1) + fibonacci(n-2);
}
int main() {
```

```
    int n;
    printf("Enter number of terms: ");
    scanf("%d", &n);
    printf("Fibonacci Series: ");
    for(int i = 0; i < n; i++)
        printf("%d ", fibonacci(i));
    printf("\n");
    return 0;
}
```

◈Java (Loop):

```
import java.util.Scanner;
public class Fibonacci {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter number of terms: ");
        int n = sc.nextInt();
        int t1 = 0, t2 = 1;
        System.out.print("Fibonacci Series: ");
        for(int i=1; i<=n; i++) {
            System.out.print(t1 + " ");
            int next = t1 + t2;
            t1 = t2;
            t2 = next;
        }
        System.out.println();
        sc.close();
    }
```

```
}
```

◈Java (Recursion):

```
import java.util.Scanner;
public class FibonacciRecursion {

    static int fibonacci(int n) {
        if(n == 0) return 0;
        if(n == 1) return 1;
        return fibonacci(n-1) + fibonacci(n-2);
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter number of terms: ");
        int n = sc.nextInt();
        System.out.print("Fibonacci Series: ");
        for(int i = 0; i < n; i++)
            System.out.print(fibonacci(i) + " ");
        System.out.println();
        sc.close();
    }
}
```


◈Python (Loop + Recursion):

```
n = int(input("Enter number of terms: "))
```

```
# Loop
a, b = 0, 1
print("Fibonacci Series (Loop):", end=" ")
for _ in range(n):
    print(a, end=" ")
    a, b = b, a + b
print()

# Recursion
def fib(n):
    if n == 0: return 0
    if n == 1: return 1
    return fib(n-1) + fib(n-2)

print("Fibonacci Series (Recursion):", end=" ")
for i in range(n):
    print(fib(i), end=" ")
print()
```


◈DSA Java (Method + Recursion + Loop Option):

```
import java.util.Scanner;

public class FibonacciDSA {

    static int fibonacciRec(int n) {
        if(n == 0) return 0;
        if(n == 1) return 1;
        return fibonacciRec(n-1) + fibonacciRec(n-2);
```

```
    }

    static void fibonacciLoop(int n) {
        int a=0, b=1;
        for(int i=0;i<n;i++) {
            System.out.print(a + " ");
            int next = a+b;
            a = b;
            b = next;
        }
        System.out.println();
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter number of terms: ");
        int n = sc.nextInt();

        System.out.print("Fibonacci Series (Loop): ");
        fibonacciLoop(n);

        System.out.print("Fibonacci Series (Recursion): ");
        for(int i=0;i<n;i++) {
            System.out.print(fibonacciRec(i) + " ");
        }
        System.out.println();
        sc.close();
    }
}
```

9️⃣ Program 9: Count Digits in a Number

🔷 C Language:

```
#include <stdio.h>
int main() {
    int n, count=0;
    printf("Enter a number: ");
    scanf("%d", &n);
    int temp = n;
    if(temp == 0) count = 1;
    while(temp != 0) {
        count++;
        temp /= 10;
    }
    printf("Number of digits in %d is %d\n", n, count);
    return 0;
}
```


🔷 Java:

```
import java.util.Scanner;
public class CountDigits {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
```

```
        System.out.print("Enter a number: ");
        int n = sc.nextInt();
        int count = (n==0)?1:0;
        int temp = n;
        while(temp != 0) {
            count++;
            temp /= 10;
        }
        System.out.println("Number of digits in " + n + " is " + count);
        sc.close();
    }
}
```

Intermediate Level Programs (10 Programs)

1. Armstrong Number Check – A number is Armstrong if sum of its digits powered by number of digits equals the number itself.

2. Reverse an Array – Reverse elements of an array.

3. Find the Second Largest Element in Array – Identify 2nd maximum number in an array.

4. Find GCD / LCM of Two Numbers – Using loops or Euclidean algorithm.

5. Array Rotation (Left / Right) – Rotate array elements by given positions.

6. Remove Duplicates from Array – Eliminate repeated elements.

7. Count Frequency of Each Element in Array – Count how many times each element occurs.

8. Check for Array Palindrome – Determine if array reads same forwards and backwards.

9. Sorting an Array – Implement Bubble Sort and Selection Sort.

10. Merge Two Arrays – Combine two arrays into one (sorted/unsorted).

1️⃣ Program 1: Armstrong Number Check

🔷 C Language:


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

int main() {
    int n, sum = 0, temp, remainder, digits = 0;
    printf("Enter a number: ");
    scanf("%d", &n);

    temp = n;
    while(temp != 0) {
        digits++;
        temp /= 10;
    }

    temp = n;
```

```
    while(temp != 0) {
        remainder = temp % 10;
        sum += pow(remainder, digits);
        temp /= 10;
    }

    if(sum == n)
        printf("%d is an Armstrong number\n", n);
    else
        printf("%d is not an Armstrong number\n", n);

    return 0;
}
```


◈Java:

```
import java.util.Scanner;

public class Armstrong {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a number: ");
        int n = sc.nextInt();
        int sum = 0, temp = n, digits = 0;

        while(temp != 0) {
            digits++;
            temp /= 10;
        }
```

```
        temp = n;
        while(temp != 0) {
            int remainder = temp % 10;
            sum += Math.pow(remainder, digits);
            temp /= 10;
        }

        if(sum == n)
            System.out.println(n + " is an Armstrong number");
        else
            System.out.println(n + " is not an Armstrong number");

        sc.close();
    }
}
```


◈Python:

```
n = int(input("Enter a number: "))
digits = len(str(n))
sum = 0
temp = n
while temp != 0:
    remainder = temp % 10
    sum += remainder ** digits
    temp //= 10

if sum == n:
```

```
    print(f"{n} is an Armstrong number")
else:
    print(f"{n} is not an Armstrong number")
```

◈DSA Java (Method + Validation):

```
import java.util.Scanner;

public class ArmstrongDSA {

    static boolean isArmstrong(int n) {
        int sum = 0, temp = n, digits = String.valueOf(n).length();
        while(temp != 0) {
            int remainder = temp % 10;
            sum += Math.pow(remainder, digits);
            temp /= 10;
        }
        return sum == n;
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a number: ");
        while(!sc.hasNextInt()) {
            System.out.println("Enter a valid integer!");
            sc.next();
        }
        int n = sc.nextInt();
        System.out.println(n + (isArmstrong(n) ? " is an Armstrong
number" : " is not an Armstrong number"));
```

```
        sc.close();
    }
}
```

2️⃣ Program 2: Reverse an Array

🔷 C Language:

```
#include <stdio.h>
int main() {
    int n;
    printf("Enter array size: ");
    scanf("%d", &n);
    int arr[n];
    printf("Enter elements: ");
    for(int i = 0; i < n; i++)
        scanf("%d", &arr[i]);

    printf("Reversed array: ");
    for(int i = n-1; i >= 0; i--)
        printf("%d ", arr[i]);
    printf("\n");
    return 0;
}
```


🔷 Java:

```
import java.util.Scanner;
public class ReverseArray {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter array size: ");
        int n = sc.nextInt();
        int[] arr = new int[n];
        System.out.println("Enter elements:");
        for(int i=0; i<n; i++)
            arr[i] = sc.nextInt();

        System.out.print("Reversed array: ");
        for(int i=n-1; i>=0; i--)
            System.out.print(arr[i] + " ");
        System.out.println();
        sc.close();
    }
}
```


◈Python:

```
arr = list(map(int, input("Enter array elements separated by space: ").split()))
print("Reversed array:", arr[::-1])
```

◈DSA Java (Method + Array Reverse):

```
import java.util.Scanner;
```

```
import java.util.Arrays;

public class ReverseArrayDSA {

    static void reverseArray(int[] arr) {
        int n = arr.length;
        for(int i=0; i<n/2; i++) {
            int temp = arr[i];
            arr[i] = arr[n-1-i];
            arr[n-1-i] = temp;
        }
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter array size: ");
        int n = sc.nextInt();
        int[] arr = new int[n];
        System.out.println("Enter elements:");
        for(int i=0;i<n;i++) arr[i] = sc.nextInt();

        reverseArray(arr);
        System.out.println("Reversed array: " + Arrays.toString(arr));
        sc.close();
    }
}
```

3️⃣ Program 3: Find the Second Largest Element in Array

🔷 C Language:

```
#include <stdio.h>
#include <limits.h>
int main() {
    int n;
    printf("Enter array size: ");
    scanf("%d", &n);
    int arr[n];
    printf("Enter elements: ");
    for(int i=0;i<n;i++) scanf("%d", &arr[i]);

    int largest = INT_MIN, second = INT_MIN;
    for(int i=0;i<n;i++){
        if(arr[i] > largest){
            second = largest;
            largest = arr[i];
        } else if(arr[i] > second && arr[i] != largest){
            second = arr[i];
        }
    }
    if(second == INT_MIN)
        printf("No second largest element\n");
    else
        printf("Second largest element is %d\n", second);
    return 0;
```

}

◈Java:

```
import java.util.Scanner;
public class SecondLargest {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter array size: ");
        int n = sc.nextInt();
        int[] arr = new int[n];
        System.out.println("Enter elements:");
        for(int i=0;i<n;i++) arr[i] = sc.nextInt();

        int largest = Integer.MIN_VALUE, second = Integer.MIN_VALUE;
        for(int num : arr){
            if(num > largest){
                second = largest;
                largest = num;
            } else if(num > second && num != largest){
                second = num;
            }
        }

        if(second == Integer.MIN_VALUE)
            System.out.println("No second largest element");
        else
            System.out.println("Second largest element is " + second);
        sc.close();
```

```
    }
}
```

◈Python:

```
arr = list(map(int, input("Enter array elements: ").split()))
largest = second = float('-inf')
for num in arr:
    if num > largest:
        second = largest
        largest = num
    elif num > second and num != largest:
        second = num
if second == float('-inf'):
    print("No second largest element")
else:
    print("Second largest element is", second)
```


◈DSA Java (Method + Arrays):

```
import java.util.Scanner;

public class SecondLargestDSA {

    static int secondLargest(int[] arr) {
        int largest = Integer.MIN_VALUE, second = Integer.MIN_VALUE;
        for(int num : arr) {
            if(num > largest) {
                second = largest;
```

```
            largest = num;
        } else if(num > second && num != largest) {
            second = num;
        }
    }
    return second == Integer.MIN_VALUE ? -1 : second;
}

public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);
    System.out.print("Enter array size: ");
    int n = sc.nextInt();
    int[] arr = new int[n];
    System.out.println("Enter elements:");
    for(int i=0;i<n;i++) arr[i] = sc.nextInt();
    int second = secondLargest(arr);
    if(second == -1) System.out.println("No second largest element");
    else System.out.println("Second largest element is " + second);
    sc.close();
}
}
```


4⃣ Program 4: Find GCD / LCM of Two Numbers

🔷 C Language:

```
#include <stdio.h>
int main() {
    int a, b, gcd, lcm;
```

```
    printf("Enter two numbers: ");
    scanf("%d %d", &a, &b);

    int x = a, y = b;
    while(y != 0){
        int temp = y;
        y = x % y;
        x = temp;
    }
    gcd = x;
    lcm = (a * b) / gcd;

    printf("GCD: %d\nLCM: %d\n", gcd, lcm);
    return 0;
}
```


◈Java:

```
import java.util.Scanner;
public class GcdLcm {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter two numbers: ");
        int a = sc.nextInt();
        int b = sc.nextInt();
        int x = a, y = b;

        while(y != 0){
            int temp = y;
```

```
            y = x % y;
            x = temp;
        }
        int gcd = x;
        int lcm = (a*b)/gcd;

        System.out.println("GCD: " + gcd);
        System.out.println("LCM: " + lcm);
        sc.close();
    }
}
```

◈Python:


```
a, b = map(int, input("Enter two numbers: ").split())
x, y = a, b
while y != 0:
    x, y = y, x % y
gcd = x
lcm = (a*b)//gcd
print("GCD:", gcd)
print("LCM:", lcm)
```

◈DSA Java (Method + Euclidean):

```
import java.util.Scanner;

public class GcdLcmDSA {
```

```
    static int gcd(int a, int b){
        while(b != 0){
            int temp = b;
            b = a % b;
            a = temp;
        }
        return a;
    }

    static int lcm(int a, int b){
        return (a*b)/gcd(a,b);
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter two numbers: ");
        int a = sc.nextInt();
        int b = sc.nextInt();

        System.out.println("GCD: " + gcd(a,b));
        System.out.println("LCM: " + lcm(a,b));
        sc.close();
    }
}
```

5️⃣ Program 5: Array Rotation (Left / Right)

◈C Language (Left Rotate by 1):

```
#include <stdio.h>
int main() {
    int n;
    printf("Enter array size: ");
    scanf("%d", &n);
    int arr[n];
    printf("Enter elements: ");
    for(int i=0;i<n;i++) scanf("%d",&arr[i]);

    int temp = arr[0];
    for(int i=0;i<n-1;i++) arr[i]=arr[i+1];
    arr[n-1] = temp;

    printf("Left rotated array: ");
    for(int i=0;i<n;i++) printf("%d ",arr[i]);
    printf("\n");
    return 0;
}
```


◈Java:

```
import java.util.Scanner;
public class ArrayRotation {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter array size: ");
```

```
        int n = sc.nextInt();
        int[] arr = new int[n];
        System.out.println("Enter elements:");
        for(int i=0;i<n;i++) arr[i]=sc.nextInt();

        int temp = arr[0];
        for(int i=0;i<n-1;i++) arr[i]=arr[i+1];
        arr[n-1]=temp;

        System.out.print("Left rotated array: ");
        for(int num: arr) System.out.print(num+" ");
        System.out.println();
        sc.close();
    }
}
```


◈Python:

```
arr = list(map(int, input("Enter array elements: ").split()))
arr = arr[1:] + arr[:1]
print("Left rotated array:", arr)
```

◈DSA Java (Method + Rotate by k positions):

```
import java.util.Scanner;
import java.util.Arrays;

public class ArrayRotationDSA {
```

```
    static void leftRotate(int[] arr, int k){
        int n = arr.length;
        k = k % n;
        int[] temp = new int[k];
        for(int i=0;i<k;i++) temp[i]=arr[i];
        for(int i=0;i<n-k;i++) arr[i]=arr[i+k];
        for(int i=0;i<k;i++) arr[n-k+i]=temp[i];
    }

    public static void main(String[] args){
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter array size: ");
        int n = sc.nextInt();
        int[] arr = new int[n];
        System.out.println("Enter elements:");
        for(int i=0;i<n;i++) arr[i]=sc.nextInt();

        System.out.print("Enter number of positions to rotate: ");
        int k = sc.nextInt();
        leftRotate(arr, k);
        System.out.println("Left rotated array: " + Arrays.toString(arr));
        sc.close();
    }
}
```

6️⃣ Program 6: Remove Duplicates from Array

◈C Language:

```
#include <stdio.h>
int main() {
    int n;
    printf("Enter array size: ");
    scanf("%d",&n);
    int arr[n];
    printf("Enter elements: ");
    for(int i=0;i<n;i++) scanf("%d",&arr[i]);

    int newArr[n], k=0;
    for(int i=0;i<n;i++){
        int j;
        for(j=0;j<k;j++){
            if(arr[i]==newArr[j]) break;
        }
        if(j==k) newArr[k++]=arr[i];
    }

    printf("Array after removing duplicates: ");
    for(int i=0;i<k;i++) printf("%d ",newArr[i]);
    printf("\n");
    return 0;
}
```


◈Java:

```
import java.util.*;
public class RemoveDuplicates {
    public static void main(String[] args){
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter array size: ");
        int n = sc.nextInt();
        int[] arr = new int[n];
        System.out.println("Enter elements:");
        for(int i=0;i<n;i++) arr[i]=sc.nextInt();

        Set<Integer> set = new LinkedHashSet<>();
        for(int num: arr) set.add(num);

        System.out.println("Array after removing duplicates: " + set);
        sc.close();
    }
}
```


◈Python:

```
arr = list(map(int, input("Enter array elements: ").split()))
arr_no_dup = list(dict.fromkeys(arr))
print("Array after removing duplicates:", arr_no_dup)
```

◈DSA Java (Method + LinkedHashSet):

```
import java.util.*;

public class RemoveDuplicatesDSA {
```

```
    static List<Integer> removeDuplicates(int[] arr){
        Set<Integer> set = new LinkedHashSet<>();
        for(int num: arr) set.add(num);
        return new ArrayList<>(set);
    }

    public static void main(String[] args){
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter array size: ");
        int n = sc.nextInt();
        int[] arr = new int[n];
        System.out.println("Enter elements:");
        for(int i=0;i<n;i++) arr[i]=sc.nextInt();

        System.out.println("Array after removing duplicates: " +
removeDuplicates(arr));
        sc.close();
    }
}
```


7⃣ Program 7: Count Frequency of Each Element in Array

🔷 C Language:

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

```
int main() {
    int n;
    printf("Enter array size: ");
    scanf("%d",&n);
    int arr[n], freq[n];
    printf("Enter 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;
    }

    printf("Element frequencies:\n");
    for(int i=0;i<n;i++){
        if(freq[i]!=0)
            printf("%d occurs %d times\n", arr[i], freq[i]);
    }
    return 0;
}
```

◈Java:

```
import java.util.*;
public class FrequencyArray {
    public static void main(String[] args){
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter array size: ");
        int n = sc.nextInt();
        int[] arr = new int[n];
        System.out.println("Enter elements:");
        for(int i=0;i<n;i++) arr[i]=sc.nextInt();

        Map<Integer,Integer> freq = new LinkedHashMap<>();
        for(int num: arr){
          freq.put(num, freq.getOrDefault(num,0)+1);
        }

        System.out.println("Element frequencies:");
        for(Map.Entry<Integer,Integer> entry: freq.entrySet()){
            System.out.println(entry.getKey() + " occurs " + entry.getValue()
+ " times");
        }
        sc.close();
    }
}
```


◈Python:

```
arr = list(map(int, input("Enter array elements: ").split()))
freq = {}
for num in arr:
    freq[num] = freq.get(num,0)+1
print("Element frequencies:")
for key, val in freq.items():
    print(f"{key} occurs {val} times")
```

🔷DSA Java (Method + HashMap):

```
import java.util.*;

public class FrequencyArrayDSA {

    static void countFrequency(int[] arr){
        Map<Integer,Integer> freq = new LinkedHashMap<>();
        for(int num: arr) freq.put(num,freq.getOrDefault(num,0)+1);

        for(Map.Entry<Integer,Integer> entry: freq.entrySet()){
            System.out.println(entry.getKey() + " occurs " + entry.getValue()
+ " times");
        }
    }

    public static void main(String[] args){
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter array size: ");
        int n = sc.nextInt();
        int[] arr = new int[n];
        System.out.println("Enter elements:");
```

```
        for(int i=0;i<n;i++) arr[i]=sc.nextInt();

        countFrequency(arr);
        sc.close();
    }
}
```

8️⃣ Program 8: Check for Array Palindrome

◈ C Language:


```
#include <stdio.h>
int main(){
    int n, flag=1;
    printf("Enter array size: ");
    scanf("%d",&n);
    int arr[n];
    printf("Enter elements: ");
    for(int i=0;i<n;i++) scanf("%d",&arr[i]);

    for(int i=0;i<n/2;i++){
        if(arr[i]!=arr[n-1-i]){
            flag=0;
            break;
        }
    }
```

```
    if(flag) printf("Array is palindrome\n");
    else printf("Array is not palindrome\n");
    return 0;
}
```

◈Java:

```
import java.util.Scanner;
public class ArrayPalindrome {
    public static void main(String[] args){
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter array size: ");
        int n = sc.nextInt();
        int[] arr = new int[n];
        System.out.println("Enter elements:");
        for(int i=0;i<n;i++) arr[i]=sc.nextInt();

        boolean isPalindrome = true;
        for(int i=0;i<n/2;i++){
            if(arr[i]!=arr[n-1-i]){
                isPalindrome=false;
                break;
            }
        }

        System.out.println(isPalindrome ? "Array is palindrome" : "Array is
not palindrome");
        sc.close();
    }
```

```
}
```

◈Python:

```
arr = list(map(int, input("Enter array elements: ").split()))
if arr == arr[::-1]:
    print("Array is palindrome")
else:
    print("Array is not palindrome")
```

◈DSA Java (Method + Arrays):

```
import java.util.*;
public class ArrayPalindromeDSA {
    static boolean isPalindrome(int[] arr){
        int n = arr.length;
        for(int i=0;i<n/2;i++){
            if(arr[i]!=arr[n-1-i]) return false;
        }
        return true;
    }

    public static void main(String[] args){
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter array size: ");
        int n = sc.nextInt();
        int[] arr = new int[n];
        System.out.println("Enter elements:");
```

```
        for(int i=0;i<n;i++) arr[i]=sc.nextInt();


        System.out.println(isPalindrome(arr) ? "Array is palindrome" :
"Array is not palindrome");
        sc.close();
    }
}
```

9️⃣ Program 9: Sorting an Array (Bubble Sort & Selection Sort)

🔷 C Language (Bubble Sort):


```
#include <stdio.h>
int main(){
    int n;
    printf("Enter array size: ");
    scanf("%d",&n);
    int arr[n];
    printf("Enter elements: ");
    for(int i=0;i<n;i++) scanf("%d",&arr[i]);

    for(int i=0;i<n-1;i++){
        for(int j=0;j<n-1-i;j++){
            if(arr[j]>arr[j+1]){
                int temp = arr[j];
                arr[j]=arr[j+1];
```

```
            arr[j+1]=temp;
        }
    }
}

printf("Sorted array (Bubble Sort): ");
for(int i=0;i<n;i++) printf("%d ",arr[i]);
printf("\n");
return 0;
}
```

◈Java (Bubble Sort):

```
import java.util.Scanner;
public class BubbleSort {
    public static void main(String[] args){
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter array size: ");
        int n = sc.nextInt();
        int[] arr = new int[n];
        System.out.println("Enter elements:");
        for(int i=0;i<n;i++) arr[i]=sc.nextInt();

        for(int i=0;i<n-1;i++){
            for(int j=0;j<n-1-i;j++){
                if(arr[j]>arr[j+1]){
                    int temp = arr[j];
                    arr[j]=arr[j+1];
                    arr[j+1]=temp;
```

```
            }
        }
    }

    System.out.print("Sorted array (Bubble Sort): ");
    for(int num: arr) System.out.print(num+" ");
    System.out.println();
    sc.close();
  }
}
```

◈Python (Bubble Sort):

```
arr = list(map(int, input("Enter array elements: ").split()))
n = len(arr)
for i in range(n-1):
    for j in range(n-1-i):
        if arr[j]>arr[j+1]:
            arr[j], arr[j+1] = arr[j+1], arr[j]
print("Sorted array (Bubble Sort):", arr)
```


◈DSA Java (Method + Bubble Sort & Selection Sort):

```
import java.util.*;

public class SortArrayDSA {

    static void bubbleSort(int[] arr){
        int n = arr.length;
```

```
        for(int i=0;i<n-1;i++){
            for(int j=0;j<n-1-i;j++){
                if(arr[j]>arr[j+1]){
                    int temp = arr[j];
                    arr[j]=arr[j+1];
                    arr[j+1]=temp;
                }
            }
        }
    }

    static void selectionSort(int[] arr){
        int n=arr.length;
        for(int i=0;i<n-1;i++){
            int minIdx=i;
            for(int j=i+1;j<n;j++){
                if(arr[j]<arr[minIdx]) minIdx=j;
            }
            int temp=arr[i];
            arr[i]=arr[minIdx];
            arr[minIdx]=temp;
        }
    }

    public static void main(String[] args){
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter array size: ");
        int n = sc.nextInt();
        int[] arr = new int[n];
```

```
        System.out.println("Enter elements:");
        for(int i=0;i<n;i++) arr[i]=sc.nextInt();

        bubbleSort(arr);
        System.out.println("Sorted array (Bubble Sort): " +
Arrays.toString(arr));

        // Resetting for Selection Sort demo
        System.out.println("Enter elements again for Selection Sort:");
        for(int i=0;i<n;i++) arr[i]=sc.nextInt();
        selectionSort(arr);
        System.out.println("Sorted array (Selection Sort): " +
Arrays.toString(arr));
        sc.close();
    }
}
```


🔟 Program 10: Merge Two Arrays

🔷C Language:

```
#include <stdio.h>
int main() {
    int n1, n2;
    printf("Enter size of first array: ");
    scanf("%d",&n1);
    int arr1[n1];
    printf("Enter elements of first array: ");
    for(int i=0;i<n1;i++) scanf("%d",&arr1[i]);
```

```
    printf("Enter size of second array: ");
    scanf("%d",&n2);
    int arr2[n2];
    printf("Enter elements of second array: ");
    for(int i=0;i<n2;i++) scanf("%d",&arr2[i]);

    int merged[n1+n2];
    for(int i=0;i<n1;i++) merged[i]=arr1[i];
    for(int i=0;i<n2;i++) merged[n1+i]=arr2[i];

    printf("Merged array: ");
    for(int i=0;i<n1+n2;i++) printf("%d ",merged[i]);
    printf("\n");
    return 0;
}
```


◈Java:

```
import java.util.Scanner;
import java.util.Arrays;
public class MergeArrays {
    public static void main(String[] args){
        Scanner sc = new Scanner(System.in);

        System.out.print("Enter size of first array: ");
        int n1 = sc.nextInt();
        int[] arr1 = new int[n1];
        System.out.println("Enter elements of first array:");
        for(int i=0;i<n1;i++) arr1[i]=sc.nextInt();
```

```
        System.out.print("Enter size of second array: ");
        int n2 = sc.nextInt();
        int[] arr2 = new int[n2];
        System.out.println("Enter elements of second array:");
        for(int i=0;i<n2;i++) arr2[i]=sc.nextInt();

        int[] merged = new int[n1+n2];
        for(int i=0;i<n1;i++) merged[i]=arr1[i];
        for(int i=0;i<n2;i++) merged[n1+i]=arr2[i];

        System.out.println("Merged array: " + Arrays.toString(merged));
        sc.close();
    }
}
```


◈Python:

```
arr1 = list(map(int, input("Enter first array elements: ").split()))
arr2 = list(map(int, input("Enter second array elements: ").split()))
merged = arr1 + arr2
print("Merged array:", merged)
```

◈DSA Java (Method + Merge Two Arrays):

```
import java.util.*;
public class MergeArraysDSA {

    static int[] mergeArrays(int[] arr1, int[] arr2){
        int n1 = arr1.length, n2 = arr2.length;
```

```
        int[] merged = new int[n1+n2];
        for(int i=0;i<n1;i++) merged[i]=arr1[i];
        for(int i=0;i<n2;i++) merged[n1+i]=arr2[i];
        return merged;
    }

    public static void main(String[] args){
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter size of first array: ");
        int n1 = sc.nextInt();
        int[] arr1 = new int[n1];
        System.out.println("Enter elements of first array:");
        for(int i=0;i<n1;i++) arr1[i]=sc.nextInt();

        System.out.print("Enter size of second array: ");
        int n2 = sc.nextInt();
        int[] arr2 = new int[n2];
        System.out.println("Enter elements of second array:");
        for(int i=0;i<n2;i++) arr2[i]=sc.nextInt();

        int[] merged = mergeArrays(arr1, arr2);
        System.out.println("Merged array: " + Arrays.toString(merged));
        sc.close();
    }
}
```


Advanced Level Programs 1–10 (21–30 overall)

1Program 1: Matrix Multiplication / Transpose / Diagonal Sum

C Language (Matrix Multiplication, Transpose, Diagonal Sum)

```
#include <stdio.h>
int main(){
    int r1,c1,r2,c2;
    printf("Enter rows and cols of first matrix: ");
    scanf("%d %d",&r1,&c1);
    int m1[r1][c1];
    printf("Enter first matrix:\n");
    for(int i=0;i<r1;i++)
        for(int j=0;j<c1;j++)
            scanf("%d",&m1[i][j]);
    printf("Enter rows and cols of second matrix: ");
    scanf("%d %d",&r2,&c2);
    if(c1!=r2){ printf("Cannot multiply\n"); return 0; }
    int m2[r2][c2], mult[r1][c2];
    printf("Enter second matrix:\n");
    for(int i=0;i<r2;i++)
        for(int j=0;j<c2;j++)
            scanf("%d",&m2[i][j]);

    // Multiplication
    for(int i=0;i<r1;i++){
        for(int j=0;j<c2;j++){
            mult[i][j]=0;
```

```
            for(int k=0;k<c1;k++)
                mult[i][j]+=m1[i][k]*m2[k][j];
        }
    }

    printf("Multiplication Result:\n");
    for(int i=0;i<r1;i++){
        for(int j=0;j<c2;j++) printf("%d ",mult[i][j]);
        printf("\n");
    }

    // Transpose of first matrix
    printf("Transpose of first matrix:\n");
    for(int i=0;i<c1;i++){
        for(int j=0;j<r1;j++)
            printf("%d ",m1[j][i]);
        printf("\n");
    }

    // Diagonal sum of first matrix (if square)
    if(r1==c1){
        int sum=0;
        for(int i=0;i<r1;i++) sum+=m1[i][i];
        printf("Diagonal sum of first matrix: %d\n", sum);
    }
    return 0;
}
```


Java:

```
import java.util.*;
public class MatrixOps {
    public static void main(String[] args){
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter rows and cols of first matrix: ");
        int r1 = sc.nextInt(), c1 = sc.nextInt();
        int[][] m1 = new int[r1][c1];
        System.out.println("Enter first matrix:");
        for(int i=0;i<r1;i++)
            for(int j=0;j<c1;j++) m1[i][j]=sc.nextInt();

        System.out.print("Enter rows and cols of second matrix: ");
        int r2 = sc.nextInt(), c2 = sc.nextInt();
        if(c1!=r2){ System.out.println("Cannot multiply"); return; }
        int[][] m2 = new int[r2][c2];
        System.out.println("Enter second matrix:");
        for(int i=0;i<r2;i++)
            for(int j=0;j<c2;j++) m2[i][j]=sc.nextInt();

        int[][] mult = new int[r1][c2];
        for(int i=0;i<r1;i++)
            for(int j=0;j<c2;j++)
                for(int k=0;k<c1;k++)
                    mult[i][j]+=m1[i][k]*m2[k][j];

        System.out.println("Multiplication Result:");
        for(int i=0;i<r1;i++){
            for(int j=0;j<c2;j++) System.out.print(mult[i][j]+" ");
```

```
            System.out.println();
        }

        System.out.println("Transpose of first matrix:");
        for(int i=0;i<c1;i++){
            for(int j=0;j<r1;j++) System.out.print(m1[j][i]+" ");
            System.out.println();
        }

        if(r1==c1){
            int sum=0;
            for(int i=0;i<r1;i++) sum+=m1[i][i];
            System.out.println("Diagonal sum of first matrix: "+sum);
        }
        sc.close();
    }
}
```


Python:

```
r1,c1 = map(int,input("Enter rows and cols of first matrix: ").split())
m1 = [list(map(int,input().split())) for _ in range(r1)]

r2,c2 = map(int,input("Enter rows and cols of second matrix: ").split())
if c1!=r2:
    print("Cannot multiply")
else:
    m2 = [list(map(int,input().split())) for _ in range(r2)]
    mult = [[sum(m1[i][k]*m2[k][j] for k in range(c1)) for j in range(c2)]
for i in range(r1)]
```

```
print("Multiplication Result:")
for row in mult: print(*row)


print("Transpose of first matrix:")
for i in range(c1): print(*[m1[j][i] for j in range(r1)])


if r1==c1:
    print("Diagonal sum of first matrix:", sum(m1[i][i] for i in
range(r1)))
```

DSA Java:

```
import java.util.*;
public class MatrixDSA {
   static int[][] multiply(int[][] a, int[][] b){
      int r1 = a.length, c1 = a[0].length, c2 = b[0].length;
      int[][] res = new int[r1][c2];
      for(int i=0;i<r1;i++)
         for(int j=0;j<c2;j++)
            for(int k=0;k<c1;k++)
               res[i][j]+=a[i][k]*b[k][j];
      return res;
   }
   static int[][] transpose(int[][] a){
      int r=a.length,c=a[0].length;
      int[][] t=new int[c][r];
      for(int i=0;i<r;i++)
         for(int j=0;j<c;j++) t[j][i]=a[i][j];
      return t;
   }
```

```
    static int diagonalSum(int[][] a){
        int sum=0;
        for(int i=0;i<a.length;i++) sum+=a[i][i];
        return sum;
    }
    public static void main(String[] args){
        Scanner sc=new Scanner(System.in);
        System.out.print("Enter rows and cols of first matrix: ");
        int r1=sc.nextInt(), c1=sc.nextInt();
        int[][] m1=new int[r1][c1];
        System.out.println("Enter first matrix:");
        for(int i=0;i<r1;i++)
            for(int j=0;j<c1;j++) m1[i][j]=sc.nextInt();

        System.out.print("Enter rows and cols of second matrix: ");
        int r2=sc.nextInt(), c2=sc.nextInt();
        if(c1!=r2){ System.out.println("Cannot multiply"); return; }
        int[][] m2=new int[r2][c2];
        System.out.println("Enter second matrix:");
        for(int i=0;i<r2;i++)
            for(int j=0;j<c2;j++) m2[i][j]=sc.nextInt();

        int[][] res=multiply(m1,m2);
        System.out.println("Multiplication Result:");
        for(int i=0;i<res.length;i++)
System.out.println(Arrays.toString(res[i]));

        System.out.println("Transpose of first matrix:");
        int[][] t=transpose(m1);
        for(int i=0;i<t.length;i++) System.out.println(Arrays.toString(t[i]));
```

```
        if(r1==c1) System.out.println("Diagonal sum: "+diagonalSum(m1));
        sc.close();
    }
}
```

2️⃣ Program 2: Check Balanced Parentheses Using Stack

C Language:

```
#include <stdio.h>
#include <string.h>
char stack[100];
int top=-1;
void push(char c){ stack[++top]=c; }
char pop(){ return stack[top--]; }
int isBalanced(char *exp){
    for(int i=0;i<strlen(exp);i++){
        if(exp[i]=='('||exp[i]=='{'||exp[i]=='[') push(exp[i]);
        else if(exp[i]==')'){
            if(top==-1||pop()!='(') return 0;
        }
        else if(exp[i]=='}'){
            if(top==-1||pop()!='{') return 0;
        }
        else if(exp[i]==']'){
```

```
            if(top==-1||pop()!='[') return 0;
        }
    }
    return top==-1;
}
int main(){
    char exp[100];
    printf("Enter expression: ");
    scanf("%s",exp);
    if(isBalanced(exp)) printf("Balanced\n");
    else printf("Not Balanced\n");
    return 0;
}
```


Java:

```
import java.util.*;
public class BalancedParentheses {
    public static boolean isBalanced(String exp){
        Stack<Character> stack=new Stack<>();
        for(char c: exp.toCharArray()){
            if(c=='('||c=='{'||c=='[') stack.push(c);
            else if(c==')'){
                if(stack.isEmpty()||stack.pop()!='(') return false;
            } else if(c=='}'){
                if(stack.isEmpty()||stack.pop()!='{') return false;
            } else if(c==']'){
                if(stack.isEmpty()||stack.pop()!='[') return false;
            }
```

```
        }
        return stack.isEmpty();
    }
    public static void main(String[] args){
        Scanner sc=new Scanner(System.in);
        System.out.print("Enter expression: ");
        String exp=sc.next();
        System.out.println(isBalanced(exp) ? "Balanced" : "Not Balanced");
        sc.close();
    }
}
```

Python:


```
exp=input("Enter expression: ")
stack=[]
pairs={'(':')','{':'}','[':']'}
balanced=True
for c in exp:
    if c in '({[': stack.append(c)
    elif c in ')}]':
        if not stack or pairs[stack.pop()]!=c:
            balanced=False
            break
print("Balanced" if balanced and not stack else "Not Balanced")
```

DSA Java:

```
import java.util.*;
```

```
public class BalancedParenthesesDSA {
    static boolean isBalanced(String exp){
        Stack<Character> s=new Stack<>();
        Map<Character,Character> map=new HashMap<>();
        map.put('(','('); map.put('{','{'); map.put('[','[');
        for(char c: exp.toCharArray()){
            if(c=='('||c=='{'||c=='[') s.push(c);
            else if(c==')'){ if(s.isEmpty()||s.pop()!='(') return false; }
            else if(c=='}'){ if(s.isEmpty()||s.pop()!='{') return false; }
            else if(c==']'){ if(s.isEmpty()||s.pop()!='[') return false; }
        }
        return s.isEmpty();
    }
    public static void main(String[] args){
        Scanner sc=new Scanner(System.in);
        System.out.print("Enter expression: ");
        String exp=sc.next();
        System.out.println(isBalanced(exp) ? "Balanced" : "Not Balanced");
        sc.close();
    }
}
```


3️⃣ Program 3: Balanced Binary Tree (Height balanced)

C Language (Check Height Balanced Binary Tree)

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

typedef struct Node{
    int data;
```

```
    struct Node *left,*right;
} Node;

Node* newNode(int data){
    Node* node=(Node*)malloc(sizeof(Node));
    node->data=data;
    node->left=node->right=NULL;
    return node;
}

int height(Node* root){
    if(!root) return 0;
    int l=height(root->left);
    if(l==-1) return -1;
    int r=height(root->right);
    if(r==-1) return -1;
    if(abs(l-r)>1) return -1;
    return (l>r?l:r)+1;
}

int isBalanced(Node* root){
    return height(root)!=-1;
}

int main(){
    Node* root=newNode(1);
    root->left=newNode(2);
    root->right=newNode(3);
    root->left->left=newNode(4);
```

```
    root->left->right=newNode(5);

    printf(isBalanced(root) ? "Balanced\n" : "Not Balanced\n");
    return 0;
}
```

Java:

```
class Node{
    int data;
    Node left,right;
    Node(int d){ data=d; left=right=null; }
}
public class BalancedBinaryTree {
    static int height(Node root){
        if(root==null) return 0;
        int l=height(root.left);
        if(l==-1) return -1;
        int r=height(root.right);
        if(r==-1) return -1;
        if(Math.abs(l-r)>1) return -1;
        return Math.max(l,r)+1;
    }
    static boolean isBalanced(Node root){ return height(root)!=-1; }
    public static void main(String[] args){
        Node root=new Node(1);
        root.left=new Node(2);
        root.right=new Node(3);
        root.left.left=new Node(4);
```

```
        root.left.right=new Node(5);
        System.out.println(isBalanced(root) ? "Balanced" : "Not Balanced");
    }
}
```

Python:

```
class Node:
    def __init__(self,data): self.data=data; self.left=self.right=None
def height(root):
    if not root: return 0
    l=height(root.left)
    if l==-1: return -1
    r=height(root.right)
    if r==-1: return -1
    if abs(l-r)>1: return -1
    return max(l,r)+1
def isBalanced(root): return height(root)!=-1

root=Node(1)
root.left=Node(2)
root.right=Node(3)
root.left.left=Node(4)
root.left.right=Node(5)

print("Balanced" if isBalanced(root) else "Not Balanced")
```

DSA Java:

(Same as above with method modularization)

4⃣ Program 4: Find Cycle in Linked List (Floyd’s Algorithm)

C Language:

```
#include <stdio.h>
#include <stdlib.h>
typedef struct Node{
    int data;
    struct Node *next;
} Node;
```



```
Node* newNode(int d){ Node* n=(Node*)malloc(sizeof(Node)); n->data=d; n->next=NULL; return n; }

int hasCycle(Node* head){
    Node *slow=head,*fast=head;
    while(fast && fast->next){
        slow=slow->next;
        fast=fast->next->next;
        if(slow==fast) return 1;
    }
    return 0;
}

int main(){
    Node* head=newNode(1);
```

```
    head->next=newNode(2);
    head->next->next=newNode(3);
    head->next->next->next=head; // create cycle
    printf(hasCycle(head) ? "Cycle detected\n" : "No cycle\n");
    return 0;
}
```

Java:

```
class Node{ int data; Node next; Node(int d){ data=d; next=null; } }
public class LinkedListCycle{
    static boolean hasCycle(Node head){
        Node slow=head, fast=head;
        while(fast!=null && fast.next!=null){
            slow=slow.next;
            fast=fast.next.next;
            if(slow==fast) return true;
        }
        return false;
    }
    public static void main(String[] args){
        Node head=new Node(1);
        head.next=new Node(2);
        head.next.next=new Node(3);
        head.next.next.next=head; // cycle
        System.out.println(hasCycle(head) ? "Cycle detected" : "No cycle");
    }
}
```

Python:

```
class Node:
    def __init__(self,d): self.data=d; self.next=None
def hasCycle(head):
    slow=fast=head
    while fast and fast.next:
        slow=slow.next
        fast=fast.next.next
        if slow==fast: return True
    return False

head=Node(1)
head.next=Node(2)
head.next.next=Node(3)
head.next.next.next=head
print("Cycle detected" if hasCycle(head) else "No cycle")
```


DSA Java:

(Same logic as above with method modularization)

5️⃣ Program 5: Lowest Common Ancestor in Binary Tree

C Language:

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

```
#include <stdlib.h>
typedef struct Node{ int data; struct Node *left,*right; } Node;
Node* newNode(int d){ Node* n=(Node*)malloc(sizeof(Node)); n->data=d; n->left=n->right=NULL; return n; }

Node* LCA(Node* root,int n1,int n2){
    if(!root) return NULL;
    if(root->data==n1||root->data==n2) return root;
    Node* l=LCA(root->left,n1,n2);
    Node* r=LCA(root->right,n1,n2);
    if(l && r) return root;
    return l?l:r;
}

int main(){
    Node* root=newNode(1);
    root->left=newNode(2); root->right=newNode(3);
    root->left->left=newNode(4); root->left->right=newNode(5);
    Node* lca=LCA(root,4,5);
    printf("LCA: %d\n", lca->data);
    return 0;
}
```


Java, Python, DSA Java:

(Logic is same as above, use Node class and recursive LCA method)

---

6 Program 6: Find Subarray with Given Sum (Sliding Window)

C Language (for positive numbers):

```
#include <stdio.h>
int main(){
    int n,sum;
    printf("Enter array size: "); scanf("%d",&n);
    int arr[n]; printf("Enter elements: ");
    for(int i=0;i<n;i++) scanf("%d",&arr[i]);
    printf("Enter sum to find: "); scanf("%d",&sum);

    int curr_sum=arr[0],start=0;
    for(int i=1;i<=n;i++){
        while(curr_sum>sum && start<i-1){
            curr_sum-=arr[start];
            start++;
        }
        if(curr_sum==sum){
            printf("Sum found between indexes %d and %d\n",start,i-1);
            return 0;
        }
        if(i<n) curr_sum+=arr[i];
    }
    printf("No subarray found\n");
    return 0;
}
```


Java, Python, DSA Java:

(Sliding window implementation for positive numbers, same logic)

7⃣ Program 7: Regular Expression Pattern Matching

Python:

```
import re
pattern = input("Enter regex pattern: ")
string = input("Enter string to match: ")
if re.fullmatch(pattern,string):
    print("Matched")
else:
    print("Not Matched")
```


Java:

```
import java.util.regex.*;
import java.util.Scanner;
public class RegexMatch{
    public static void main(String[] args){
        Scanner sc=new Scanner(System.in);
        System.out.print("Enter regex: "); String pattern=sc.next();
        System.out.print("Enter string: "); String str=sc.next();
        if(Pattern.matches(pattern,str)) System.out.println("Matched");
        else System.out.println("Not Matched");
        sc.close();
    }
```

}

C:

(C does not have native regex in standard library; POSIX regex via <regex.h> can be used.)

DSA Java:

(Use Java Pattern.matches as above in a method)

8️⃣ Program 8: Huffman Coding Implementation (Character Frequency Compression)

Java (DSA style with PriorityQueue)


```
import java.util.*;
class Node implements Comparable<Node>{
    char ch; int freq; Node left,right;
    Node(char ch,int freq){ this.ch=ch; this.freq=freq; }
    Node(int freq,Node l,Node r){ this.freq=freq; left=l; right=r; }
    public int compareTo(Node o){ return this.freq-o.freq; }
}

public class HuffmanCoding{
    static void printCodes(Node root,String s){
        if(root.left==null && root.right==null){
            System.out.println(root.ch + ": " + s);
            return;
        }
        if(root.left!=null) printCodes(root.left,s+"0");
```

```
        if(root.right!=null) printCodes(root.right,s+"1");
    }

    public static void main(String[] args){
        String text="huffman";
        Map<Character,Integer> freqMap=new HashMap<>();
        for(char c:text.toCharArray())
            freqMap.put(c,freqMap.getOrDefault(c,0)+1);

        PriorityQueue<Node> pq=new PriorityQueue<>();
        for(Map.Entry<Character,Integer> e: freqMap.entrySet())
            pq.add(new Node(e.getKey(),e.getValue()));

        while(pq.size()>1){
            Node left=pq.poll();
            Node right=pq.poll();
            Node merged=new Node(left.freq+right.freq,left,right);
            pq.add(merged);
        }

        Node root=pq.peek();
        System.out.println("Huffman Codes:");
        printCodes(root,"");
    }
}
```


Python (Huffman using heapq):

```
import heapq,collections
```

```
text="huffman"
freq=collections.Counter(text)
heap=[[wt,[ch,""]] for ch,wt in freq.items()]
heapq.heapify(heap)

while len(heap)>1:
    lo=heapq.heappop(heap)
    hi=heapq.heappop(heap)
    for pair in lo[1:]: pair[1]='0'+pair[1]
    for pair in hi[1:]: pair[1]='1'+pair[1]
    heapq.heappush(heap,[lo[0]+hi[0]]+lo[1:]+hi[1:])

huff=heapq.heappop(heap)
print("Huffman Codes:")
for p in huff[1:]: print(f"{p[0]}: {p[1]}")
```

9️⃣ Program 9: N-Queens Problem (Backtracking)

C Language:

```
#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]) return 0;
    for(int i=row,j=col;i>=0 && j>=0;i--,j--) if(board[i][j]) return 0;
    for(int i=row,j=col;i<N && j>=0;i++,j--) if(board[i][j]) return 0;
    return 1;
}

int solve(int col){
    if(col>=N) return 1;
    for(int i=0;i<N;i++){
        if(isSafe(i,col)){
            board[i][col]=1;
            if(solve(col+1)) return 1;
            board[i][col]=0;
        }
    }
    return 0;
}

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(){
    if(solve(0)) printBoard();
    else printf("No solution\n");
    return 0;
```

```
}
```

Java:

```
public class NQueens{
    static int N=4;
    static int[][] board=new int[N][N];

    static boolean isSafe(int row,int col){
        for(int i=0;i<col;i++) if(board[row][i]==1) return false;
        for(int i=row,j=col;i>=0 && j>=0;i--,j--) if(board[i][j]==1) return
false;
        for(int i=row,j=col;i<N && j>=0;i++,j--) if(board[i][j]==1) return
false;
        return true;
    }
```



```
    static boolean solve(int col){
        if(col>=N) return true;
        for(int i=0;i<N;i++){
            if(isSafe(i,col)){
                board[i][col]=1;
                if(solve(col+1)) return true;
                board[i][col]=0;
            }
        }
        return false;
    }

    static void printBoard(){
```

```
        for(int i=0;i<N;i++){
            for(int j=0;j<N;j++) System.out.print(board[i][j]+" ");
            System.out.println();
        }
    }

    public static void main(String[] args){
        if(solve(0)) printBoard();
        else System.out.println("No solution");
    }
}
```

Python:


```
N=4
board=[[0]*N for _ in range(N)]

def isSafe(row,col):
    for i in range(col):
        if board[row][i]==1: return False
    for i,j in zip(range(row,-1,-1),range(col,-1,-1)):
        if board[i][j]==1: return False
    for i,j in zip(range(row,N),range(col,-1,-1)):
        if board[i][j]==1: return False
    return True

def solve(col):
    if col>=N: return True
    for i in range(N):
```

```
        if isSafe(i,col):
            board[i][col]=1
            if solve(col+1): return True
            board[i][col]=0
    return False

if solve(0):
    for row in board: print(row)
else: print("No solution")
```

DSA Java:

(Same as Java above, can wrap solve() and isSafe() in separate class methods.)

---

[10] Program 10: Trapping Rainwater Problem

C Language:

```
#include <stdio.h>
int min(int a,int b){ return a<b?a:b; }
int main(){
    int n;
    printf("Enter number of bars: "); scanf("%d",&n);
    int arr[n]; printf("Enter heights: ");
    for(int i=0;i<n;i++) scanf("%d",&arr[i]);

    int left[n], right[n];
```

```
    left[0]=arr[0];
    for(int i=1;i<n;i++) left[i]=(arr[i]>left[i-1]?arr[i]:left[i-1]);
    right[n-1]=arr[n-1];
    for(int i=n-2;i>=0;i--) right[i]=(arr[i]>right[i+1]?arr[i]:right[i+1]);

    int water=0;
    for(int i=0;i<n;i++) water+=min(left[i],right[i])-arr[i];
    printf("Trapped water: %d\n", water);
    return 0;
}
```

Java:

```
import java.util.*;
public class TrappingRainwater{
    public static void main(String[] args){
        Scanner sc=new Scanner(System.in);
        System.out.print("Enter number of bars: ");
        int n=sc.nextInt();
        int[] arr=new int[n];
        System.out.println("Enter heights:");
        for(int i=0;i<n;i++) arr[i]=sc.nextInt();
        int[] left=new int[n];
        int[] right=new int[n];
        left[0]=arr[0];
        for(int i=1;i<n;i++) left[i]=Math.max(left[i-1],arr[i]);
        right[n-1]=arr[n-1];
        for(int i=n-2;i>=0;i--) right[i]=Math.max(right[i+1],arr[i]);
        int water=0;
```

```
        for(int i=0;i<n;i++) water+=Math.min(left[i],right[i])-arr[i];
        System.out.println("Trapped water: "+water);
        sc.close();
    }
}
```

Python:

```
arr=list(map(int,input("Enter heights: ").split()))
n=len(arr)
left=[0]*n
right=[0]*n
left[0]=arr[0]
for i in range(1,n): left[i]=max(left[i-1],arr[i])
right[n-1]=arr[n-1]
for i in range(n-2,-1,-1): right[i]=max(right[i+1],arr[i])
water=sum(min(left[i],right[i])-arr[i] for i in range(n))
print("Trapped water:",water)
```


DSA Java:

(Same logic as above, can be wrapped in calculateTrappedWater(int[] arr) method)
