

1. Write a Program to reverse a number.

```
// C Programs to Calculate
// reverse of a number
#include <stdio.h>

// Iterative approach
int reverse_iteration(int N)
{
    int ans = 0;
    while (N != 0) {

        ans = ans * 10 + (N % 10);
        N = N / 10;
    }

    return ans;
}

// recursive approach
int reverse(int n, int ans)
{
    if (n == 0)
        return ans;

    ans = ans * 10 + n % 10;
    return reverse(n / 10, ans);
}

int main()
{
    int N = 15942;
    printf("Initial number:%d\n", N);

    N = reverse_iteration(N);
    printf("%d after reverse using iteration\n", N);

    int ans = 0;
    ans = reverse(N, ans);
    printf("%d after again reverse using recursion", ans);

    return 0;
}
```

2. Check whether a number is a palindrome.

```
// C Program for
// Checking Palindrome
#include <stdio.h>

// Checking if the number is
// Palindrome number
void check_palindrome(int N)
{
    int T = N;
    int rev = 0; // This variable stored reversed digit

    // Execute a while loop to reverse digits of given
    // number
    while (T != 0) {
        rev = rev * 10 + T % 10;
        T = T / 10;
    }

    // Compare original_number with reversed number
    if (rev == N)
        printf("%d is palindrome\n", N);
    else
        printf("%d is not a palindrome\n", N);
}

int main()
{
    int N = 13431;
    int M = 12345;

    // Function call
    check_palindrome(N);
    check_palindrome(M);

    return 0;
}
```

3. Write a C Program to check if two numbers are equal without using the comparison operator.

```
// C Program for checking numbers
// are equal using bitwise operator
#include <stdio.h>

int main()
{
    int x = 1;
    int y = 2;

    // Using XOR
    // XOR of two equal numbers is 0
    if (!(x ^ y))
        printf(" %d is equal to %d ", x, y);
    else
        printf(" %d is not equal to %d ", x, y);

    return 0;
}
```

4. Write a C program to find the GCD of two numbers.

```
// C program to find GCD of two numbers
#include <math.h>
#include <stdio.h>

// Function to return gcd of a and b
int gcd(int a, int b)
{
    // Find Minimum of a and b
    int result = ((a < b) ? a : b);
    while (result > 0) {
        if (a % result == 0 && b % result == 0) {
            break;
        }
        result--;
    }
    return result; // return gcd of a and b
}

// Driver program to test above function
int main()
{
    int a = 98, b = 56;
```

```

printf("GCD of %d and %d is %d ", a, b, gcd(a, b));
return 0;
}

```

5. Write a C program to find the LCM of two numbers.

```

// C program to find
// LCM of two numbers
#include <stdio.h>

// minimum of two numbers
int Min(int Num1, int Num2)
{
    if (Num1 >= Num2)
        return Num2;
    else
        return Num1;
}

int LCM(int Num1, int Num2, int K)
{
    // If either of the two numbers
    // is 1, return their product
    if (Num1 == 1 || Num2 == 1)
        return Num1 * Num2;

    // If both the numbers are equal
    if (Num1 == Num2)
        return Num1;

    // If K is smaller than the
    // minimum of the two numbers
    if (K <= Min(Num1, Num2)) {

        // Checks if both numbers are
        // divisible by K or not
        if (Num1 % K == 0 && Num2 % K == 0) {

            // Recursively call LCM() function
            return K * LCM(Num1 / K, Num2 / K, 2);
        }

        // Otherwise
        else
            return LCM(Num1, Num2, K + 1);
    }
}

```

```

        // If K exceeds minimum
        else
            return Num1 * Num2;
    }

int main()
{
    // Given N & M
    int N = 12, M = 9;

    // Function Call
    int ans = LCM(N, M, 2);

    printf("%d", ans);

    return 0;
}

```

6. Write a C Program to find the Maximum and minimum of two numbers without using any loop or condition.

```

// C Program to check
// Maximum and Minimum
// Between two numbers
// Without any condition or loop
#include <stdio.h>
#include <stdlib.h>

int main()
{
    int a = 55, b = 23;

    // return maximum among the two numbers
    printf("max = %d\n", ((a + b) + abs(a - b)) / 2);

    // return minimum among the two numbers
    printf("min = %d", ((a + b) - abs(a - b)) / 2);

    return 0;
}

```

7. Write a Program in C to Print all natural numbers up to N without using a semi-colon.

```
// C program to print
// all natural numbers
// upto N without using semi-colon
#include <stdio.h>
#define N 10

int main(int val)
{
    if (val <= N && printf("%d ", val) && main(val + 1)) {
    }
}
```

8. Write a Program to find the area of a circle.

```
// C program to find area
// of circle
#include <math.h>
#include <stdio.h>
#define PI 3.142

double findArea(int r) { return PI * pow(r, 2); }

int main()
{
    printf("Area is %f", findArea(5));
    return 0;
}
```

9. Write a Program to create a pyramid pattern using C.

```
// C Program print Pyramid pattern
#include <stdio.h>

int main()
{
    int N = 5;

    // Outer Loop for number of rows
    for (int i = 1; i <= N; i++) {

        // inner Loop for space printing
```

```

        for (int j = 1; j <= N - i; j++)
            printf(" ");

        // inner Loop for star printing
        for (int j = 1; j < 2 * i; j++)
            printf("*");
        printf("\n");
    }
    return 0;
}

```

10. Write a program to form Pascal Triangle using numbers.

```

// C Program to print
// Pascal's Triangle
#include <stdio.h>

int main()
{
    int n = 5;

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

        int x = 1;

        for (int j = 1; j <= i; j++) {
            printf("%d  ", x);
            x = x * (i - j) / j;
        }
        printf("\n");
    }

    return 0;
}

```