1) FIBONACCI (RECURSIVE) - compact, well-commented

```c
#include <stdio.h>

// Return nth Fibonacci number (recursive)
int fibonacci(int n) {
    if (n == 0) return 0;     // base case
    if (n == 1) return 1;     // base case
    return fibonacci(n - 1) + fibonacci(n - 2);
}

int main() {
    int n, i;
    printf("Enter number of terms: ");
    if (scanf("%d", &n) != 1) return 0;
    printf("Fibonacci Series: ");
    for (i = 0; i < n; i++) {
        printf("%d ", fibonacci(i));
    }
    printf("\n");
    return 0;
}
```

2) FACTORIAL (ITERATIVE)

```c
#include <stdio.h>

int main() {
    int num, i;
    unsigned long long fact = 1ULL; // large enough for moderate inputs
    printf("Enter a non-negative integer: ");
    if (scanf("%d", &num) != 1) return 0;
    if (num < 0) {
        printf("Factorial not defined for negative numbers.\n");
        return 0;
    }
    for (i = 1; i <= num; ++i) fact *= i;
    printf("Factorial of %d = %llu\n", num, fact);
    return 0;
}
```

3) ODD OR EVEN

```c
#include <stdio.h>

int main() {
    int num;
    printf("Enter an integer: ");
    if (scanf("%d", &num) != 1) return 0;
    if (num % 2 == 0)
        printf("%d is Even\n", num);
    else
        printf("%d is Odd\n", num);
    return 0;
}
```

4) MATRIX MULTIPLICATION (cleaned, with dynamic limits)

```c
#include <stdio.h>

int main() {
    int r1, c1, r2, c2, i, j, k;
    printf("Enter rows and columns of first matrix: ");
    if (scanf("%d %d", &r1, &c1) != 2) return 0;
    printf("Enter rows and columns of second matrix: ");
    if (scanf("%d %d", &r2, &c2) != 2) return 0;
    if (c1 != r2) {
        printf("Matrix multiplication not possible! (columns of A must equal rows of B)\n");
        return 0;
    }

    // Use fixed-size arrays but only up to the entered sizes
    int A[10][10], B[10][10], C[10][10];
    printf("Enter elements of first matrix (%d x %d):\n", r1, c1);
    for (i = 0; i < r1; ++i)
        for (j = 0; j < c1; ++j)
            scanf("%d", &A[i][j]);

    printf("Enter elements of second matrix (%d x %d):\n", r2, c2);
    for (i = 0; i < r2; ++i)
        for (j = 0; j < c2; ++j)
            scanf("%d", &B[i][j]);

    // Initialize result
    for (i = 0; i < r1; ++i)
        for (j = 0; j < c2; ++j)
            C[i][j] = 0;

    // Multiply
    for (i = 0; i < r1; ++i)
        for (j = 0; j < c2; ++j)
            for (k = 0; k < c1; ++k)
                C[i][j] += A[i][k] * B[k][j];

    printf("Resultant Matrix:\n");
    for (i = 0; i < r1; ++i) {
        for (j = 0; j < c2; ++j) printf("%d ", C[i][j]);
        printf("\n");
    }
    return 0;
}
```

5) FACTORIAL (RECURSIVE)

```
#include <stdio.h>

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

int main() {
    int num;
    printf("Enter a non-negative integer: ");
    if (scanf("%d", &num) != 1) return 0;
    if (num < 0) {
        printf("Factorial not defined for negative numbers.\n");
        return 0;
    }
    printf("Factorial of %d = %llu\n", num, factorial((unsigned int)num));
    return 0;
}
```

*** End of programs ***