MCA Lab Assignment - C Solutions (Q16 - Q39)

The following C file is menu-driven: functions Q16() .. Q39() implement each question.
Save as: assignment_q16_q39.c
Compile: gcc assignment_q16_q39.c -o assign16
Run: ./assign16

----- CODE START -----

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <math.h>

unsigned long long fact(unsigned int n) {
    unsigned long long f = 1;
    for (unsigned int i = 2; i <= n; ++i) f *= i;
    return f;
}

/* Q16: Even numbers from 50 to 100 and their sum */
void Q16() {
    int sum = 0;
    printf("Even numbers from 50 to 100:\n");
    for (int i = 50; i <= 100; ++i) {
        if (i % 2 == 0) {
            printf("%d ", i);
            sum += i;
        }
    }
    printf("\nSum = %d\n", sum);
}

/* Q17: Odd numbers from 50 to 100 and their count */
void Q17() {
    int count = 0;
    printf("Odd numbers from 50 to 100:\n");
    for (int i = 50; i <= 100; ++i) {
        if (i % 2 != 0) {
            printf("%d ", i);
            count++;
        }
    }
    printf("\nCount = %d\n", count);
}

/* Q18: Numbers divisible by 7 or 11 (from 50..100) and their sum */
void Q18() {
    int sum = 0;
    printf("Numbers from 50 to 100 divisible by 7 or 11:\n");
    for (int i = 50; i <= 100; ++i) {
        if (i % 7 == 0 || i % 11 == 0) {
            printf("%d ", i);
            sum += i;
        }
    }
    printf("\nSum = %d\n", sum);
}

/* Q19: Prime numbers from 5 to 50 and their count */
int is_prime(int x) {
    if (x < 2) return 0;
    if (x == 2) return 1;
    if (x % 2 == 0) return 0;
    for (int i = 3; i * i <= x; i += 2)
        if (x % i == 0) return 0;
    return 1;
}
void Q19() {
    int count = 0;
    printf("Primes from 5 to 50:\n");
    for (int i = 5; i <= 50; ++i) {
        if (is_prime(i)) {
            printf("%d ", i);
            count++;
        }
    }
    printf("\nCount = %d\n", count);
}

/* Q20: Fibonacci sequence of n terms */
void Q20() {
    int n;
    printf("Enter number of terms n: ");
    if (scanf("%d", &n) != 1 || n <= 0) { printf("Invalid n\n"); return; }
    long long a = 0, b = 1;
    printf("Fibonacci series (%d terms):\n", n);
    for (int i = 1; i <= n; ++i) {
        printf("%lld ", a);
        long long next = a + b;
        a = b; b = next;
    }
    printf("\n");
}

/* Q21: Biggest and smallest element in array of 10 elements */
void Q21() {
    int arr[10];
    printf("Enter 10 integers:\n");
    for (int i = 0; i < 10; ++i) scanf("%d", &arr[i]);
    int max = arr[0], min = arr[0];
    for (int i = 1; i < 10; ++i) {
        if (arr[i] > max) max = arr[i];
        if (arr[i] < min) min = arr[i];
    }
    printf("Max = %d, Min = %d\n", max, min);
}

/* Q22: Sort ascending (10 elements) and find biggest/smallest */
void Q22() {
    int n = 10, a[10];
    printf("Enter 10 integers:\n");
    for (int i = 0; i < n; ++i) scanf("%d", &a[i]);
    // simple bubble sort
    for (int i = 0; i < n-1; ++i)
        for (int j = 0; j < n-1-i; ++j)
            if (a[j] > a[j+1]) { int t = a[j]; a[j] = a[j+1]; a[j+1] = t; }
    printf("Sorted ascending:\n");
    for (int i = 0; i < n; ++i) printf("%d ", a[i]);
    printf("\nSmallest = %d, Biggest = %d\n", a[0], a[n-1]);
}

/* Q23: Sort descending (10 elements) and find biggest/smallest */
void Q23() {
    int n = 10, a[10];
    printf("Enter 10 integers:\n");
    for (int i = 0; i < n; ++i) scanf("%d", &a[i]);
    // sort descending
    for (int i = 0; i < n-1; ++i)
        for (int j = 0; j < n-1-i; ++j)
            if (a[j] < a[j+1]) { int t = a[j]; a[j] = a[j+1]; a[j+1] = t; }
    printf("Sorted descending:\n");
    for (int i = 0; i < n; ++i) printf("%d ", a[i]);
    printf("\nSmallest = %d, Biggest = %d\n", a[n-1], a[0]);
}

/* GCD helper */
int gcd(int x, int y) {
    if (x == 0) return y;
    if (y == 0) return x;
    x = abs(x); y = abs(y);
    while (y) { int t = y; y = x % y; x = t; }
    return x;
}
/* LCM helper */
long long lcm_longlong(long long a, long long b) {
    if (a == 0 || b == 0) return 0;
    return llabs(a / gcd(a,b) * b);
}

/* Q24: GCD of n integers using array */
void Q24() {
    int n;
    printf("Enter count of integers n: ");
    if (scanf("%d", &n) != 1 || n <= 0) { printf("Invalid n\n"); return; }
    int *arr = malloc(sizeof(int)*n);
    printf("Enter %d integers:\n", n);
    for (int i = 0; i < n; ++i) scanf("%d", &arr[i]);
    int g = arr[0];
    for (int i = 1; i < n; ++i) g = gcd(g, arr[i]);
    printf("GCD = %d\n", g);
    free(arr);
}

/* Q25: LCM of n integers using array */
void Q25() {
    int n;
    printf("Enter count of integers n: ");
    if (scanf("%d", &n) != 1 || n <= 0) { printf("Invalid n\n"); return; }
    long long *arr = malloc(sizeof(long long)*n);
    printf("Enter %d integers:\n", n);
    for (int i = 0; i < n; ++i) scanf("%lld", &arr[i]);
    long long res = arr[0];
    for (int i = 1; i < n; ++i) res = lcm_longlong(res, arr[i]);
    printf("LCM = %lld\n", res);
    free(arr);
}

/* Q26: Sum of diagonal elements of 3x3 matrix */
void Q26() {
    int mat[3][3];
    printf("Enter 3x3 matrix elements row-wise:\n");
    for (int i = 0; i < 3; ++i)
        for (int j = 0; j < 3; ++j) scanf("%d", &mat[i][j]);
    int sum = 0;
    for (int i = 0; i < 3; ++i) sum += mat[i][i]; // main diagonal
    printf("Sum of diagonal elements = %d\n", sum);
}

/* Q27: Factorial using iterative and recursive */
unsigned long long fact_recursive(unsigned int n) {
    if (n == 0) return 1;
    return n * fact_recursive(n-1);
}
void Q27() {
    unsigned int n;
    printf("Enter n for factorial: ");
    if (scanf("%u", &n) != 1) { printf("Invalid\n"); return; }
    printf("Iterative: %u! = %llu\n", n, fact(n));
    printf("Recursive: %u! = %llu\n", n, fact_recursive(n));
}

/* Q28: Sum 1! + 2! + ... + n! using function */
void Q28() {
    unsigned int n;
    printf("Enter n: ");
    if (scanf("%u", &n) != 1 || n == 0) { printf("Invalid n\n"); return; }
    unsigned long long sum = 0;
    for (unsigned int i = 1; i <= n; ++i) sum += fact(i);
    printf("Sum = %llu\n", sum);
```

```c
}

/* Q29: Sum 1! + 3! + 5! + ... + n! (odd terms) using function */
void Q29() {
    unsigned int n;
    printf("Enter max n (odd max or will include odds up to it): ");
    if (scanf("%u", &n) != 1 || n == 0) { printf("Invalid n\n"); return; }
    unsigned long long sum = 0;
    for (unsigned int i = 1; i <= n; i += 2) sum += fact(i);
    printf("Sum = %llu\n", sum);
}

/* Q30: Sum 2! + 4! + 6! + ... + n! (even terms) using function */
void Q30() {
    unsigned int n;
    printf("Enter max n (even max or will include evens up to it): ");
    if (scanf("%u", &n) != 1 || n < 2) { printf("Invalid n\n"); return; }
    unsigned long long sum = 0;
    for (unsigned int i = 2; i <= n; i += 2) sum += fact(i);
    printf("Sum = %llu\n", sum);
}

/* Q31: Sum = x + x^2/2 + x^3/3 + ... + x^n/n (using function) */
void Q31() {
    double x; int n;
    printf("Enter x and n: ");
    if (scanf("%lf %d", &x, &n) != 2 || n < 1) { printf("Invalid input\n"); return; }
    double sum = 0.0;
    for (int i = 1; i <= n; ++i) sum += pow(x, i) / (double)i;
    printf("Sum = %.6f\n", sum);
}

/* Q32: Sum = x + x^2/2! + x^3/3! + ... + x^n/n! (using function) */
void Q32() {
    double x; int n;
    printf("Enter x and n: ");
    if (scanf("%lf %d", &x, &n) != 2 || n < 1) { printf("Invalid input\n"); return; }
    double sum = 0.0;
    for (int i = 1; i <= n; ++i) sum += pow(x, i) / (double)fact(i);
    printf("Sum = %.8f\n", sum);
}

/* Q33: Product = (Sum1 * Sum2) / Sum3 where Sum1=1..m, Sum2=1..n, Sum3=1..p */
void Q33() {
    int m, n, p;
    printf("Enter m n p: ");
    if (scanf("%d %d %d", &m, &n, &p) != 3) { printf("Invalid\n"); return; }
    long long sum1 = 0, sum2 = 0, sum3 = 0;
    for (int i = 1; i <= m; ++i) sum1 += i;
    for (int i = 1; i <= n; ++i) sum2 += i;
    for (int i = 1; i <= p; ++i) sum3 += i;
    if (sum3 == 0) { printf("Division by zero\n"); return; }
    double product = (double)sum1 * (double)sum2 / (double)sum3;
    printf("Sum1=%lld, Sum2=%lld, Sum3=%lld, Product=(S1*S2)/S3 = %.6f\n", sum1, sum2, sum3, product);
}

/* Q34: p = a! * b! / c! */
void Q34() {
    unsigned int a, b, c;
    printf("Enter a b c: ");
    if (scanf("%u %u %u", &a, &b, &c) != 3) { printf("Invalid\n"); return; }
    unsigned long long A = fact(a), B = fact(b), C = fact(c);
    if (C == 0) { printf("0 factorial? invalid\n"); return; }
    double p = (double)A * (double)B / (double)C;
    printf("p = %.6f\n", p);
}

/* Q35: k = a!*b!*c! / (a! + b!) */
void Q35() {
    unsigned int a, b, c;
    printf("Enter a b c: ");
    if (scanf("%u %u %u", &a, &b, &c) != 3) { printf("Invalid\n"); return; }
    unsigned long long A = fact(a), B = fact(b), C = fact(c);
    unsigned long long denom = A + B;
    if (denom == 0) { printf("Division by zero\n"); return; }
    double k = (double)A * (double)B * (double)C / (double)denom;
    printf("k = %.6f\n", k);
}

/* Q36: R1= x*x+y+z, R2=x+y*y+z, K = (R1*R2)/(R1+R2) */
void Q36() {
    double x, y, z;
    printf("Enter x y z: ");
    if (scanf("%lf %lf %lf", &x, &y, &z) != 3) { printf("Invalid\n"); return; }
    double R1 = x*x + y + z;
    double R2 = x + y*y + z;
    if ((R1 + R2) == 0) { printf("Division by zero\n"); return; }
    double K = (R1 * R2) / (R1 + R2);
    printf("R1=%.6f, R2=%.6f, K=%.6f\n", R1, R2, K);
}

/* Q37: R = (x*x + y + z*2) * (x + y*y + z) / (x + y + z) */
void Q37() {
    double x, y, z;
    printf("Enter x y z: ");
    if (scanf("%lf %lf %lf", &x, &y, &z) != 3) { printf("Invalid\n"); return; }
    double num1 = x*x + y + 2*z;
    double num2 = x + y*y + z;
    double den = x + y + z;
    if (den == 0) { printf("Division by zero\n"); return; }
    double R = (num1 * num2) / den;
    printf("R = %.6f\n", R);
}

/* Q38: nCr using multiplicative method */
unsigned long long nCr(unsigned int n, unsigned int r) {
    if (r > n) return 0;
    if (r > n - r) r = n - r;
    unsigned long long result = 1;
    for (unsigned int i = 1; i <= r; ++i) {
        result = result * (n - r + i) / i;
    }
    return result;
}
void Q38() {
    unsigned int n, r;
    printf("Enter n and r: ");
    if (scanf("%u %u", &n, &r) != 2) { printf("Invalid\n"); return; }
    unsigned long long comb = nCr(n, r);
    printf("C(%u, %u) = %llu\n", n, r, comb);
}

/* Q39: Structure to display a marks table for 5 students */
struct Student {
    int roll;
    char name[50];
    int sub1, sub2, sub3;
};
void Q39() {
    struct Student s[5];
    printf("Enter details for 5 students (Roll Name sub1 sub2 sub3):\n");
    for (int i = 0; i < 5; ++i) {
        printf("Student %d: ", i+1);
        scanf("%d %49s %d %d %d", &s[i].roll, s[i].name, &s[i].sub1, &s[i].sub2, &s[i].sub3);
    }
    printf("\nMarks Table\nCourse Name: _________  Semester: _________\n");
    printf("S.No  Roll   Name        Sub1  Sub2  Sub3  Total  Per(%%)\n");
    for (int i = 0; i < 5; ++i) {
        int total = s[i].sub1 + s[i].sub2 + s[i].sub3;
        double per = total / 3.0;
        printf("%-5d %-6d %-10s %-5d %-5d %-5d %-6d %-6.2f\n", i+1, s[i].roll, s[i].name, s[i].sub1, s[i].sub2, s
    }
}

/* MAIN MENU for Q16 - Q39 */
int main() {
    int choice;
    do {
        printf("\n--- MENU (Q16-Q39) ---\n");
        printf("16.Evens(50-100) 17.Odds(50-100) 18.Div7or11 19.Primes(5-50) 20.Fibonacci\n");
        printf("21.ArrayMinMax 22.SortAsc 23.SortDesc 24.GCD 25.LCM 26.DiagSum\n");
        printf("27.Factorial:Iter+Rec 28.SumFact 29.SumOddFact 30.SumEvenFact\n");
        printf("31.Series x^i/i 32.Series x^i/i! 33.Product(Sum1*Sum2)/Sum3\n");
        printf("34.P = a!*b!/c! 35.k = a!*b!*c!/(a!+b!) 36.Expr K 37.Expr R 38.nCr 39.MarksTable\n");
        printf("0.Exit\nEnter choice: ");
        if (scanf("%d", &choice) != 1) break;
        switch(choice) {
            case 16: Q16(); break;
            case 17: Q17(); break;
            case 18: Q18(); break;
            case 19: Q19(); break;
            case 20: Q20(); break;
            case 21: Q21(); break;
            case 22: Q22(); break;
            case 23: Q23(); break;
            case 24: Q24(); break;
            case 25: Q25(); break;
            case 26: Q26(); break;
            case 27: Q27(); break;
            case 28: Q28(); break;
            case 29: Q29(); break;
            case 30: Q30(); break;
            case 31: Q31(); break;
            case 32: Q32(); break;
            case 33: Q33(); break;
            case 34: Q34(); break;
            case 35: Q35(); break;
            case 36: Q36(); break;
            case 37: Q37(); break;
            case 38: Q38(); break;
            case 39: Q39(); break;
            case 0: printf("Exiting...\n"); break;
            default: printf("Invalid choice\n"); break;
        }
    } while(choice != 0);
    return 0;
}
----- CODE END -----
```

End of file.