```c
#include "definitions.h"
#include "myFunctions.h"
#include "params.h"
#include "typedefs.h"

INT main(INT argc, char *argv[])
{
    if (argc < 4) {
        perror("Command-line usage: executableName <m><n><k>");
        exit(1);
    }

    INT   m  = atoi(argv[ 1 ]);
    INT   n  = atoi(argv[ 2 ]);
    INT   k  = atoi(argv[ 3 ]);
    REAL *a  = malloc(sizeof(*a) * m * n);
    REAL *b  = malloc(sizeof(*b) * n * k);
    REAL *c  = calloc(m * k, sizeof(*c));
    REAL *c2 = calloc(m * k, sizeof(*c2));
    REAL *c3 = calloc(m * k, sizeof(*c2));
// initializing the two matrices
    InitializeMatrices(a, b, m, n, k);

    printf(GREEN "\n=====MATRIX A[%d X %d]==========\n" RESET, m, n);
    printMatrix(a, m, n);

    printf(GREEN "=====MATRIX B[%d X %d]=======\n" RESET, n, k);

    printMatrix(b, n, k);

    REAL start, finish, elapsedTime;

    GET_TIME(start);
// multiplying the matrices
    matrixMultiply(a, b, c, m, n, k);

    GET_TIME(finish);
    printf(CYAN "****************** CPU FUNCTION RESULT ******************\n" RESET);
    printf(GREEN "=====MATRIX C[%d X %d]=====\n" RESET, m, k);
    printMatrix(c, m, k);
    elapsedTime = finish - start;

    printf(GREEN "elapsed wall time = " YELLOW " %.5f seconds" RESET "\n" RESET, elapsedTi
me);
    printf(CYAN "\n************* CPU CBLAS_DDOT FUNCTION RESULT *************** \n" RE
SET);

    GET_TIME(start);
// multiplying the matrices involving cblas ddot() function
    ddot_Matrix_Mult(a, b, c, m, n, k);

    GET_TIME(finish);

    printf(GREEN "=====MATRIX C[%d X %d]=====\n" RESET, m, k);

    printMatrix(c, m, k);
    elapsedTime = finish - start;

    printf(GREEN "elapsed wall time =" YELLOW " %.5f seconds" RESET "\n" RESET, elapsedTim
e);

    printf(CYAN "\n************* CPU CBLAS_DAXPY FUNCTION RESULT *************\n" RES
ET);
    GET_TIME(start);
// multiplying the matrices involving cblas daxy() function
    daxpy_Matrix_Mult(a, b, c2, m, n, k);

    GET_TIME(finish);
    printf(GREEN "=====MATRIX C[%d X %d]=====\n" RESET, m, k);
```



```c
    printMatrix(c2, m, k);
    elapsedTime = finish - start;

    printf(GREEN "elapsed wall time =" YELLOW " %.5f seconds" RESET " \n\n" RESET, elapsedT
ime);

    printf(CYAN "\n************* CPU CBLAS_DGEMM FUNCTION RESULT *************\n" RES
ET);
    GET_TIME(start);
// multiplying the matrices using cblas dgemm function
    dgemm_Matrix_Mult(a, b, c3, m, n, k);

    GET_TIME(finish);
    printf(GREEN "=====MATRIX C[%d X %d]=====\n" RESET, m, k);

    printMatrix(c3, m, k);
    elapsedTime = finish - start;

    printf(GREEN "elapsed wall time =" YELLOW " %.5f seconds" RESET " \n\n" RESET, elapsedT
ime);

    free(a);
    free(b);
    free(c);
    free(c2);
    free(c3);

    return EXIT_SUCCESS;
}
```