C (CLang)/teme/tema7/main.c

```c
#include <stdio.h>

#define MAX 50

// void afisare_vector(int v, int ind_vec) {
//     for (int i = 0; i < ind_vec; i++) {  // afisare
//         printf("%d ", v[i]);
//     }
// }

int prim(int n) {
    for (int i = 2; i <= n / 2; i++) {
        if (n % i == 0) {
            return 0;
        }
    }
    return 1;
}

int main() {
    int mat[MAX][MAX], nl, nc, vector[MAX * MAX], indice_vector, vector1[MAX],
        vector2[MAX], vector3[MAX], indice_vector1, indice_vector2,
        indice_vector3, pas;
    // citire(mat[MAX][MAX], nl, nc);
    scanf("%d%d", &nl, &nc);        // citim nr de linii si nr de coloane
    for (int l = 0; l < nl; l++) {  // citim matricea
        for (int c = 0; c < nc; c++) {
            scanf("%d", &mat[l][c]);
        }
    }
    // problema1
    for (int l = 0; l < nl; l++) {  // citim matricea
        for (int c = 0; c < nc; c++) {
            if (mat[l][c] % 2 == 0) {
                vector1[indice_vector1++] = mat[l][c];
            }
        }
    }

    // problema2
    for (int l = 0; l < nl; l++) {  // citim matricea
        for (int c = 0; c < nc; c++) {
            if (prim(mat[l][c])) {
                vector2[indice_vector2++] = mat[l][c];
            }
        }
    }

    // problema3
    for (int suma = 0; suma < nl + nc;
         suma++) {  // parcurgem valorile pe care suma indicilor le poate avea
```

```
    if (suma % 2 == 0) {  // daca suma e para, (pentru alternanta):
        for (int l = 0; l < nl; l++) {  // parcurgem matricea normal
            for (int c = 0; c < nc; c++) {
                if (l + c == suma) {  // daca l + c = suma (asta se intampla
                                      // pe o singura diagonala)
                    vector[indice_vector++] = mat[l][c];
                }
            }
        }

    } else {
        for (int l = nl - 1; l >= 0; l--) {  // parcurgem matricea invers
            for (int c = nc - 1; c >= 0; c--) {
                if (l + c == suma) {
                    vector[indice_vector++] = mat[l][c];
                }
            }
        }
    }
}

// problema4
while (indice_vector3 < nl * nc) {
    for (int c = pas; c < nc - (pas + 1) && indice_vector3 < nl * nc; c++) {
        vector3[indice_vector3++] = mat[pas][c];
    }
    for (int l = pas; l < nl - (pas + 1) && indice_vector3 < nl * nc; l++) {
        vector3[indice_vector3++] = mat[l][nc - (pas + 1)];
    }
    for (int c = nc - (pas + 1); c > pas && indice_vector3 < nl * nc; c--) {
        vector3[indice_vector3++] = mat[nl - (pas + 1)][c];
    }
    for (int l = nl - (pas + 1); l > pas && indice_vector3 < nl * nc; l--) {
        vector3[indice_vector3++] = mat[l][pas];
    }
    // pt o matrice patratica elementul din centru nu satisface conditiile
    // din foruri asa ca il adaug manual
    if (nl == nc && nl % 2 == 1 && indice_vector3 == nl * nc - 1) {
        vector3[indice_vector3++] = mat[nl / 2][nc / 2];
    }

    pas++;
}

// afisari
for (int i = 0; i < nl * nc; i++) {  // afisare
    printf("%d ", vector[i]);
}
printf("\n");
for (int i = 0; i < indice_vector1; i++) {  // afisare
    printf("%d ", vector1[i]);
}
printf("\n");
for (int i = 0; i < indice_vector2; i++) {  // afisare
    printf("%d ", vector2[i]);
```

```
    }
    printf("\n");
    for (int i = 0; i < indice_vector3; i++) {  // afisare
        printf("%d ", vector3[i]);
    }

    return 0;
}
```