```c
#include <stdio.h>

int main(){
    int m, n, k, p, i, j, dx, dy;
    int iCoord, jCoord;
    scanf("%d %d %d", &m, &n, &k);
    int image[m][n];
    float kernel[k][k], ksum, sum;

    // Read in the image
    for(i = 0; i < m; i++)
        for(j = 0; j < n; j++)
            scanf("%d", &image[i][j]);

    // Read in the kernel as well as simultaneously
    // calculate the sum of kernel entries
    ksum = 0;
    for(i = 0; i < k; i++){
        for(j = 0; j < k; j++){
            scanf("%f", &kernel[i][j]);
            ksum += kernel[i][j];
        }
    }

    // Mid point of the kernel - it has odd number of rows and columns
    p = (k-1)/2;
    for(i = 0; i < m; i++){
        for(j = 0; j < n; j++){
            sum = 0;

            // Take the dot product of the kernel with image patch
            for(dx = -p; dx <= p; ++dx){
                for(dy = -p; dy <= p; ++dy){
                    // Find (wrapped around) x coordinate of image
                    // iCoord = i + dy if i + dy < m
                    // iCoord wraps around if i + dy >= m
                    iCoord = (i + dy + m) % m;
                    // Find (wrapped around) y coordinate of image
                    // jCoord = j + dx if j + dx < n
                    // jCoord wraps around if j + dx >= n
                    jCoord = (j + dx + n) % n;
                    sum += image[iCoord][jCoord]*kernel[p+dy][p+dx];
                }
            }

            printf("%d", (int)(sum / ksum));
            if(j < n - 1) printf(" "); // No trailing spaces
        }
        if(i < m - 1) printf("\n"); // No trailing newlines
    }

    return 0;
}
```