```cpp
#include <opencv2/opencv.hpp>
#include <iostream>
#include <stdio.h>
#include <stdlib.h>

using namespace std;
using namespace cv;


void ReadBytes(char *, char *);

/*
void ReadBytes_1(char *, char *);
void calcHistogram(IplImage *, int *);
void drawHistogram(const char *, int *);
*/


int main(int argc, char ** argv) {
    ReadBytes(argv[1], argv[2]);
    //ReadBytes_1(argv[1], argv[2]);
    return 0;
}

/// ///////////////////////////////////////////////////////////////
void ReadBytes(char * filename, char * out) {
    FILE * f = fopen(filename, "rb");

    if(f == NULL)
        throw "Argument Exception";

    unsigned char info[34];
    fread(info, sizeof(unsigned char), 34, f);    // read the 34-byte header

    // extract image height and width from header
    int width = *(short*)&info[30];
    int height = *(short*)&info[28];
    int IMOD = *(short*)&info[32];

    cout << endl;
    cout << "Width: " << width << endl;
    cout << "Height: " << height << endl;
    cout << "IMOD: " << IMOD << endl;
    cout << "Effective Height" << height * (IMOD + 1) << endl;


    uchar data[height * (IMOD + 1)][width];
    unsigned char temp;

    for(int i = 0; i < height * (IMOD + 1); i++) {
        for(int j=0; j < width; j++) {
            fread(&temp, sizeof(unsigned char), 1, f);
            data[i][j] = temp;
        }
    }

    fclose(f);

    Mat MMM = Mat(height * (IMOD + 1), width, CV_8U, data);
    MMM.convertTo(MMM, CV_32F, 1, 0);

    imwrite(out, MMM(Rect(9, 0, width - 10, height * (IMOD + 1))));
}






/*
void ReadBytes_1(char * filename, char * histName) {
    IplImage * img = cvLoadImage(filename, CV_LOAD_IMAGE_GRAYSCALE);
    int hist[256];
    calcHistogram(img, hist);
    drawHistogram(histName, hist);
}

void func(uchar c, int * h) {
    unsigned x = c;
    (*(h + x))++;
}

void calcHistogram(IplImage * gray, int hist []) {
        /// Initialize histograms
    for (int i = 0; i < 256; i++)
        hist[i] = 0;

        /// Calculate histogram
    for(int i = 0; i < gray->height; i++) {
        char * ptr = gray->imageData + i*gray->widthStep;
        for(int j = 0; j < gray->width; j++) {
                func((uchar)(*ptr), hist);
                ptr++;
        }
    }
}

void drawHistogram(const char * histName, int * hist) {
    int st = 4;             // To separate lines in the histograms, for better look ;)
    int dep = 256;

    /// Create image for the histogram
    IplImage * his = cvCreateImage(cvSize(st * dep, 600), IPL_DEPTH_8U, 3);
    cvSet(his, cvScalar(0,0,0));           // Initialize image (all black)
    his->origin = IPL_ORIGIN_BL;           // Set the origin in the bottom left corner

    for (int i = 0; i < 8 ; i++) {
        int x = dep / 8;
        cvLine(his, cvPoint(i * x * st, 0), cvPoint(i * x * st, 600), cvScalar
          (122,122,122), 1, 4);
    }

        /// Draw the histogram
    for (int i = 0; i < dep; i++)
        if (hist[i] != 0)
            cvLine(his, cvPoint(i * st, 0), cvPoint(i * st, hist[i] / 10), cvScalar
              (255, 0, 0), 1, 4);

    cvSaveImage(histName, his);

    cvReleaseImage(&his);
}
*/
```