#include <stdio.h>

#include <stdlib.h>

#include <opencv/cv.h>

#include <opencv/highgui.h>

int main(int argc, char\*\* argv) {

if (argc != 2) {

printf("Usage: %s image\_file\_name\n", argv[0]);

return EXIT\_FAILURE;

}

//CV\_LOAD\_IMAGE\_COLOR = 1 forces the resultant IplImage to be colour.

//CV\_LOAD\_IMAGE\_GRAYSCALE = 0 forces a greyscale IplImage.

//CV\_LOAD\_IMAGE\_UNCHANGED = -1

IplImage\* Img1 = cvLoadImage(argv[1], CV\_LOAD\_IMAGE\_COLOR);

IplImage\* ImgAnd = cvCreateImage(cvSize(640, 480),Img1->depth , 4);

IplImage\* ImgMin = cvCreateImage(cvSize(640, 480), Img1->depth, 4);

// Always check if the program can find a file

if (!ImgAnd) {

printf("Error: fichero %s no leido\n", argv[1]);

return EXIT\_FAILURE;

}

// a visualization window is created with title 'image'

cvNamedWindow("android", CV\_WINDOW\_NORMAL);

cvNamedWindow("minion", CV\_WINDOW\_NORMAL);

// img is shown in 'image' window

cvShowImage("android", ImgAnd);

cvShowImage("minion", ImgMin);

cvWaitKey(0);

for (int fila = 0; fila < ImgAnd->height; fila++) {

\_\_m128i \*pImgAnd = (\_\_m128i \*) (ImgAnd->imageData + fila \* ImgAnd->widthStep);

\_\_m128i \*pImgMin = (\_\_m128i \*) (ImgMin->imageData + fila \* ImgMin->widthStep);

for (int columna = 0; columna < ImgAnd->widthStep; columna = columna + 16) {

\*pImgAnd = \_mm\_set1\_epi32 (0x00AC639);

pImgAnd++;

\*pImgMin = \_mm\_set1\_epi32 (0x00F5E050);

pImgMin++;

}

}

cvShowImage("android", ImgAnd);

cvShowImage("minion", ImgMin);

cvWaitKey(1);

cvWaitKey(0);

// memory release for img before exiting the application

cvReleaseImage(&ImgAnd);

// Self-explanatory

cvDestroyWindow(argv[1]);

return EXIT\_SUCCESS;

}