#include <stdio.h>

#include <stdlib.h>

#include <math.h>

#include <iostream>

using namespace std;

//This program reads in an image file (of type BMP), copies it into a two-dimensional array,

// inverts the image, and writes the inverted image to an output file

//function prototypes to process a BMP file

long getImageInfo(FILE\*, long, int); // gets info from the header

void copyImageInfo(FILE\* inputFile, FILE\* outputFile); // copies header info to output file

void copyColorTable(FILE\* inputFile, FILE\* outputFile, int nColors); //copies color table to output

void main()

{

FILE \*bmpInput, \*invertedOutput;

unsigned char oneChar;

unsigned char invertedChar;

const int WHITE = 255;

//open input and output files

bmpInput = fopen("MRI.bmp", "rb");

invertedOutput = fopen("invertedMRI.bmp", "wb");

// Read the header info to get the rows and columns - first some constants for the offsets

const int COLUMN\_OFFSET = 18;

const int ROW\_OFFSET = 22;

const int COLORS\_OFFSET = 46;

const int SIZE = 4;

const int HEADER\_SIZE = 54;

int numberRows = getImageInfo(bmpInput, ROW\_OFFSET, SIZE);

int numberColumns = getImageInfo(bmpInput, COLUMN\_OFFSET, SIZE);

int nColors = getImageInfo(bmpInput, COLORS\_OFFSET, 4);

//dynamically allocate the two-dimensional array (matrix)

// first the rows, then for each row, the correct number of columns

unsigned char \* \* imageArray;

//copy image header info and color table to output BMP file

copyImageInfo(bmpInput, invertedOutput);

copyColorTable(bmpInput, invertedOutput, nColors);

//set file pointer past the headers in input and output files

fseek(bmpInput, (HEADER\_SIZE + 4\*nColors), SEEK\_SET);

fseek(invertedOutput, (HEADER\_SIZE + 4\*nColors), SEEK\_SET);

//read image data from file and put into 2-D array

for(int r=0; r<numberRows; r++)

{

for(int c=0; c<numberColumns; c++)

{

fread(&oneChar, sizeof(char), 1, bmpInput);

imageArray[r][c] = oneChar;

}

}

// invert char

//invertedChar = WHITE - oneChar;

//write inverted char to output file

fwrite(&oneChar, sizeof(char), 1, invertedOutput);

//invert all bytes and write to output file

//fwrite(&invertedChar, sizeof(char), 1, invertedOutput);

cout << "Open the output file to see the inverted image" <<endl;

system("PAUSE");

}

//function that reads the bytes from the specified offset from the file

long getImageInfo(FILE\* inputFile, long offset, int numberOfChars)

{

long value = 0L;

unsigned char temp;

int i;

fseek(inputFile, offset, SEEK\_SET); //point to correct part of file

for(i=1; i<=numberOfChars; i++)

{

fread(&temp, sizeof(char), 1, inputFile);

// calculate value based on adding bytes

value = (long)(value + (temp)\*(pow(256.0, (i-1))));

}

return(value);

}

//copy header info from input to output file

void copyImageInfo(FILE\* inputFile, FILE\* outputFile)

{

unsigned char temp;

int i;

fseek(inputFile, 0L, SEEK\_SET);

fseek(outputFile, 0L, SEEK\_SET);

const int HEADER\_SIZE = 50;

for(i=0; i<=HEADER\_SIZE; i++)

{

fread(&temp, sizeof(char), 1, inputFile);

fwrite(&temp, sizeof(char), 1, outputFile);

}

}

//copy color table from input to output file

void copyColorTable(FILE\* inputFile, FILE\* outputFile, int nColors)

{

unsigned char temp;

int i;

// point past the headers in both files

fseek(inputFile, 54L, SEEK\_SET);

fseek(outputFile, 54L, SEEK\_SET);

for(i=0; i<=(4\*nColors); i++) /\* there are (4\*nColors) bytes in color table \*/

{

fread(&temp, sizeof(char), 1, inputFile);

fwrite(&temp, sizeof(char), 1, outputFile);

}

}