Assignment 1

Student Name : Wei Li

Part 1:

The code should read the mri.pgm image and output the mri2.pgm image. The objective of this part of the assignment is to black out the right-two-quadrant of the image.

Code:

```
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[]){
unsigned sizeX;            //image width
unsigned sizeY;            //image height
unsigned char *image;   //image 1D array
unsigned levels;

//read image from file
FILE *iFile = fopen("mri.pgm","r"); // iFile point to the image
if(iFile == 0) return 1;
if(3!=fscanf(iFile, "P5 %d %d %d ", &sizeX, &sizeY, &levels)) return 1;  //%d decimal , level means brightness
image = (unsigned char *) malloc(sizeX*sizeY);
fread(image, sizeof(unsigned char), sizeX*sizeY, iFile);
fclose(iFile);

int max = 255;
for(int i=0; i<256; i++){
            for(int j=i+128; j<i+256; j++)
                        image[i*max + j] = 0;
}

//write image to file
iFile = fopen("mri2.pgm","w");
if(iFile == 0) return 1; //error handling
fprintf(iFile, "P5 %d %d %d ",sizeX,sizeY, 255);//write header
fwrite(image, sizeof(unsigned char), sizeX*sizeY, iFile);//write binary image
fclose(iFile);


return 0;
}
```

Part2:

The objective of this part of the assignment is to discard the odd pixels and resize the image to 1/4 of the original size. I create another variable called ‘store’ to store the ¼ size of the image and use nested for loop to go through 128x128 pixels and put the even pixels from the original image.

Code:

```
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[]){
unsigned sizeX;             //image width
unsigned sizeY;             //image height
unsigned char *image;   //image 1D array
unsigned char *store;   //store resized image
unsigned levels;

//read image from file
FILE *iFile = fopen("mri.pgm","r"); // iFile point to the image
if(iFile == 0) return 1;
if(3!=fscanf(iFile, "P5 %d %d %d ", &sizeX, &sizeY, &levels)) return 1;  //%d decimal , level means brightness
image = (unsigned char *) malloc(sizeX * sizeY);
fread(image, sizeof(unsigned char), sizeX*sizeY, iFile);
fclose(iFile);

store = (unsigned char *) malloc(sizeX/2 * sizeY/2);

int max = 255;
for(int i=0; i<128; i++){
        for(int j=0; j<128; j++){
                store[i*(max/2) + j] = image[i*max*2 + j*2];
        }
}

free(image);

//write image to file
iFile = fopen("mri2-2.pgm","w");
if(iFile == 0) return 1; //error handling
fprintf(iFile, "P5 %d %d %d ", sizeX/2,sizeY/2, 255); //write header
fwrite(store, sizeof(unsigned char), sizeX/2*sizeY/2, iFile); //write binary image
fclose(iFile);
return 0;
}
```