CSE4202//Lab 6: Assignment

File I/O:
Image processing using C programs

Report Prepared by:

Sadman Shaharier Mahim

220041133, Group-1A

Semester-2

CSE

TASK 1: Flipping an image horizontally.

We can flip an image horizontally basically by flipping the data for the horizontal pixels. This has been done in the execution of the program designed for flipping the image horizontally. Below we can have a block by block

The program was designed to be dynamic, that is it can not only work with the **provided "100dollar.tif"** but also any file. This can be passed in Command line arguments as well as manually inputted in the terminal.

```
int main(int argc,char *argv[]){

    char fl_nam[200];
    if (argc<2)
    {
        printf("Give the image file name (tif file only)=");
        fflush(stdin);
        //fgets(fl_nam,199,stdin);
        scanf("%s",fl_nam);
    }
    else{
        strncpy(fl_nam,argv[1],sizeof(fl_nam));
    }
    printf("the output file name is %s",fl_nam);
```

Here we take main function arguments "argc" and "argv". They may not be passed as we may also execute the program by double clicking in the file explorer and not by "./task1" in the terminal. So we check the value of command line arguments passed by seeing if argc is less than 2. If so then the program didn't receive the file name when it was executed.

```
    }

    char dest[200];
    pros_name(fl_nam,dest);
    printf("the output file name is %s",dest);
```

Next we run the file name processor function to make the file name for the output file for us. The general format is "inputname_flipped.tif"

```c
int pros_name(char str[],char str2[]){

    char copy[200];
    strcpy(copy,str);
    char *tok=strtok(copy,".");
    strcpy(copy,tok);
    strcat(copy,"_flipped.tif");
    strcpy(str2,copy);
}
```

The file processor function takes in two input char array pointers. The first one contains the source file name which is then used to make the target file name using string tokenizations.

```c
FILE *src=fopen(fl_nam,"rb");
FILE *targ=fopen(dest,"wb");

if (src==NULL)
{
    perror("ERROR failed to open source image");
    return -1;
}
else if (targ==NULL)
{
    perror("ERROR failed to target image");
    return -1;
}
```

We then open the file using fopen and check if they have been successfully opened. If failed we output the failed reason using perror.

The initial 8 byte metadata is read and printed into the destination file and then we read an HxW array of data and store them into an array[H][W]. This is essentially the pixel wise data.

```c
char ch;
for (int i = 0; i <8; i++)   //GET THE META DATA
{
    ch=fgetc(src);
    fputc(ch, targ);
}

//SETTING WIDTH AND HEIGHT
const int H=500;
const int W=1192;

//READING AND STORING PIXEL DATA
int pix[H][W];
for(int i=0; i<H; i++){
    for(int j=0; j<W; j++){

        pix[i][j]=fgetc(src);
    }

}
```

```
    //SAVING THE PIXELS IN HORIZONTAL FLIP TO FILE
    for (int i = 0; i < H; i++)
    {
        for (int j = W-1; j >=0; j--)
        {
            fputc(pix[i][j],targ);
        }

    }
```

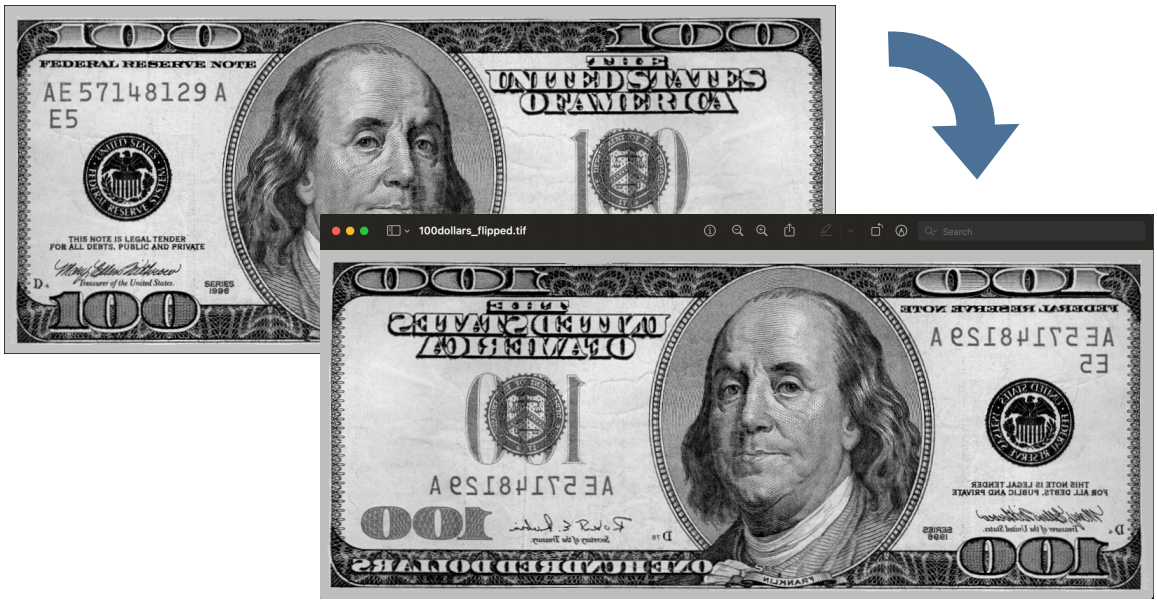
Here we are printing the pixel data in horizontal flip by starting from W-1 instead from 0 and decrementing gradually in every iteration.

```
    //COPYING TAILING METADATAS
    do
    {
        ch=fgetc(src);
        fputc(ch, targ);
    } while (ch!=EOF);


    fclose(src);
    fclose(targ);
    printf("\nOPERATION SUCCEEDED");
    return 0;

}
```

Finally the last bit of metadata is copied onto the target file and the files are closed giving us a horizontally flipped image.



Here is an example of how the file may be passed in two different ways:

```
Give the image file name (tif file only)=100dollars.tif
the output file name is 100dollars_flipped.tif%
sadmanshaharier@MacBook-Air-M2 my code %
```

Here we are manually giving the input into the program

```
sadmanshaharier@MacBook-Air-M2 my code % ./task1 100dollars.tif
the output file name is 100dollars_flipped.tif%
sadmanshaharier@MacBook-Air-M2 my code %
```

Here we are passing the filename using command like arguments

It is guaranteed that doth the cases will make the same flipped image file with the same image input.

TASK 2: Blurring an image.

We can use similar techniques to make an image blurry using file operations. For instance all the execution and logic are same till the printing of the pixels into the file. Previously after taking the input from the file, we printed the pixel data in an horizontal reverse order. But this time we will not do so. To blur an image we have to follow any one of the blurring methods and the corresponding algorithm. Here I have implemented **Box blur** to keep things simple.

```
//SETTING WIDTH AND HEIGHT & BLUR STRENGTH
const int H=500;
const int W=1192;
int blr;
printf("\nGive blur strenght:");
scanf("%d",&blr);
```

We first take the value of the blur strength the user wants to achieve.

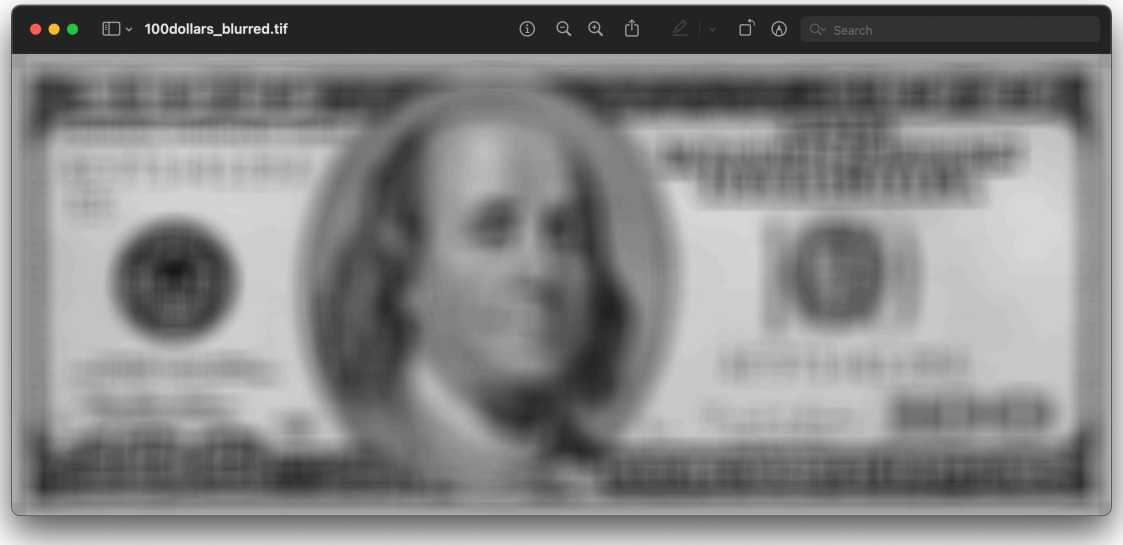
```
Give the image file name (tif file only)=100dollars.tif
the output file name is 100dollars_blurred.tif
Give blur strenght:15
OPERATION SUCCEEDED%
sadmanshaharier@MacBook-Air-M2 my code %
```

```
//BOX BLUR
//AVRAGING THE VALUE OF N BY N PIXLES AROUND TARGETED PIXEL
for (int i = 0; i < H; i++)
{
    for (int j = 0; j < W; j++)
    {
        int avg=0;
        for (int x = i-blr; x < i+blr; x++)
        {
            for (int y = j-blr; y < j+blr; y++)
            {
                avg+=pix[(x+H)%H][(y+W)%W];
            }

        }
        int divd=((blr*2)+1)*((blr*2)+1);
        avg/=divd;
        fputc(avg,targ);
    }

}
```

Here we have iterated over all the pixels in the first two for loops. In each iteration we have taken the value of 'n' pixels in a square grid around the targeted pixel. And thus we have achieve the average of said grid and placed the average value in its place. This is how basically box blur works.



This is the image with a blur strength of 15.

Finally we close the file streams to release the memories allocated to them. And even in this we can pass both manual input and command line argument.

Ps. The icons used are interactive and have the source linked to google drive.