2014 EECS 22 Assignment 2

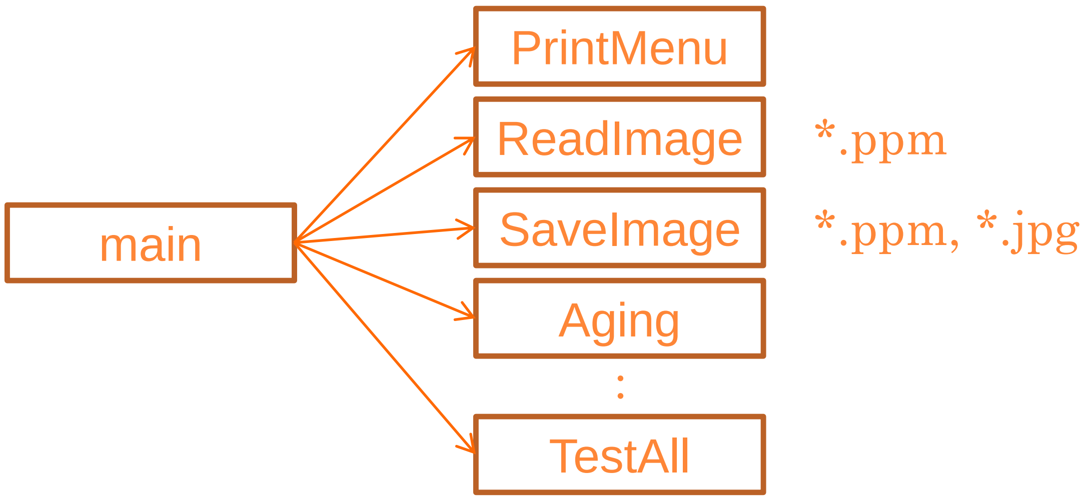
Updated by : Manjunath

Created by : Che-Wei Chang

Assignment 2

- A menu driven digital image processing program [100 pts + 10 bonus pts]
- Deadline : 10/28/2014, Tuesday, 11:00 pm

- Goal
  - Main function use function calls to input/output image, process image, and test all of the digital image process functions.



Menu Driven Digital Image Processing

```
eecs22@zuma.eecs.uci.edu:6 > ./PhotoLab
--------------------------------
 1:  Load a PPM image
 2:  Save an image in PPM and JPEG format
 3:  Change a color image to Black & White
 4:  Flip an image vertically
 5:  Mirror an image horizontally
 6:  Color filter an image
 7:  Sketch the edge of an image
 8:  Shuffle an image
 9:  BONUS: Add Border to an image
 10: Test all functions
 11: Exit
please make your choice:
```

INPUT FILE

- Format : ppm
- Option 1: input ppm file
  - `Load a PPM image`
  - example 1:

    -
      ```
      please make your choice: 1
      Please input the file name to load: RingMall
      RingMall.ppm was read successfully!
      ```
  - File extension is not needed.
  - example 2:

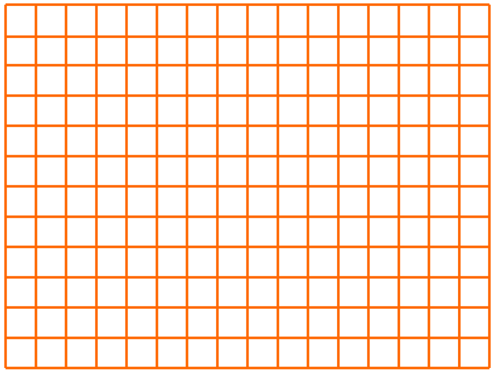
    -
      ```
      please make your choice: 1
      Please input the file name to load: RingMall.ppm
      Cannot open file " RingMall.ppm.ppm" for reading!
      ```
  - Function for reading image `ReadImage` is provided !

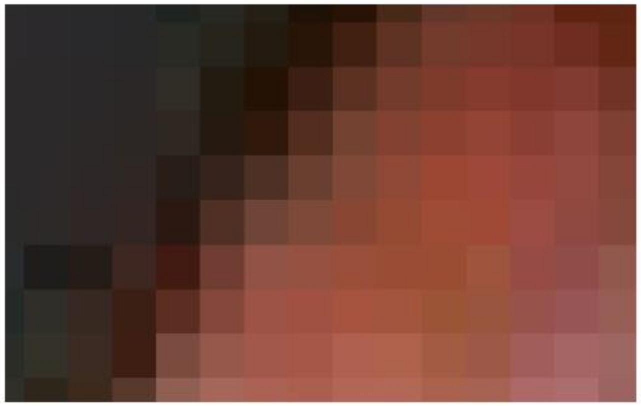
Output File

- Format : ppm, jpg
- Option 2: output ppm and jpg files
  - `Save an image in PPM and JPEG format`
  - example:
    ```
    Please make your choice: 2
    Please input the file name to save: bw
    bw.ppm was saved successfully.
    bw.jpg was stored for viewing.
    ```
  - File extension is not needed.
  - Function for saving image `SaveImage` is provided !

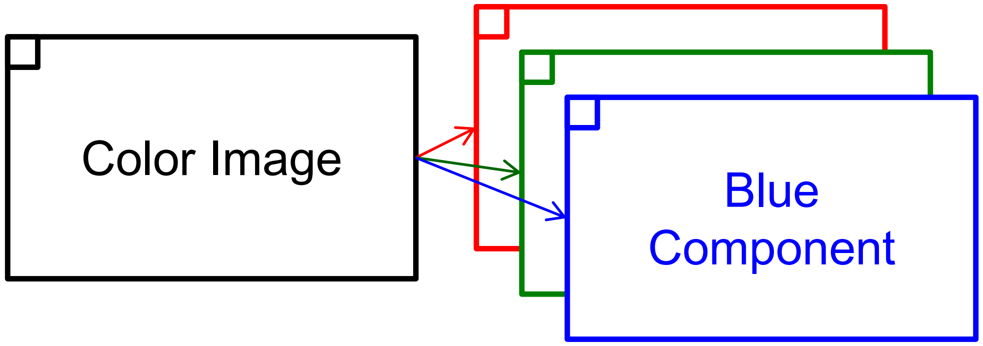
Picture in the program

- How to represent a picture in computer?
  - A picture is composed of pixels
  - One color for each pixel
  - Example: 16x12 = 192 pixels

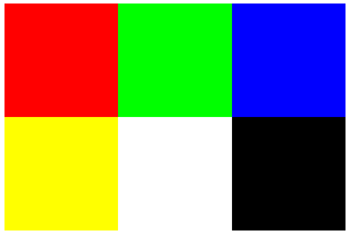




PICTURE IN THE PROGRAM



- 3-tuple (R, G, B)
  - R: intensity of Red
  - G: intensity of Green
  - B: intensity of Blue
  - For image in ppm format, the range of the intensity is [0,255], using **unsigned char** for each intensity
- Color examples:
  - Red (255, 0, 0), Green (0, 255, 0), Blue (0, 0, 255)
  - Yellow (255, 255, 0), Cyan (0, 255, 255), Magenta(255, 0, 255)
  - White (255, 255, 255), black(0, 0, 0)
- PPM example
- RGBRGBRGBRGB…
-
```
P3    (colors)
3 2   (3 columns, 2 rows)
255   (255 for max color)
255    0 0         0 255    0      0 0 255
255 255 0       255 255 255      0 0    0
```

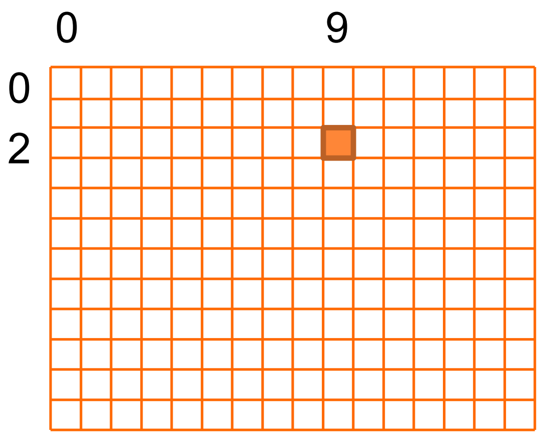


Picture in the Program

- The data structure to represent a picture in this assignment
  - Two-dimensional arrays for the intensities of each pixel
    - For an image of size 16x12…

```
unsigned char R[16][12];
unsigned char G[16][12];
unsigned char B[16][12];
```

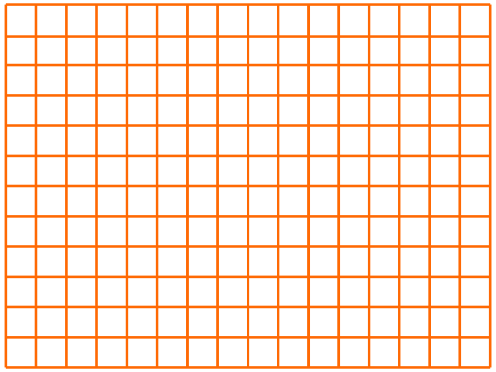
    - How to access a pixel in an image?
      - Coordinate of a pixel (x, y)
      - x = number of the column
      - y = number of the row
      - The color of the pixel (x, y) = (R[x][y], G[x][y], B[x][y])



Picture in the Program

- How to access every pixel in the picture?
  - List all possible coordinates of the pixel
  - Two for-loops to scan all the pixels in a 2-D array
  - Inner loop
    - fix the number of the column, iterate the pixel in the same column with different row numbers
  - Outer loop
    - iterate all the columns

```
int x, y ;
for (x=0; x < 16; x++){
 for (y=0; y < 12; y++){
  processing on pixel(x, y);
 }
}
```

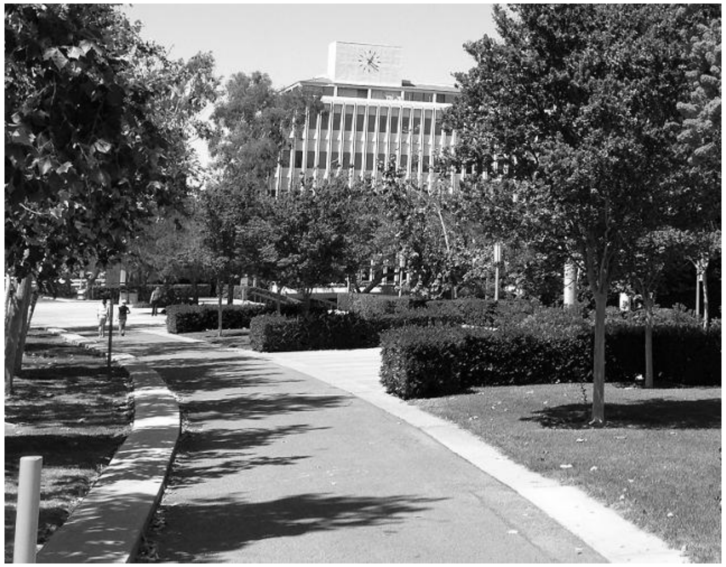


Digital Image Processing Function

```
eecs22@zuma.eecs.uci.edu:6 > ./PhotoLab
--------------------------------
 1:  Load a PPM image
 2:  Save an image in PPM and JPEG format
 3:  Change a color image to Black & White
 4:  Flip an image vertically
 5:  Mirror an image horizontally
 6:  Color filter an image
 7:  Sketch the edge of an image
 8:  Shuffle an image
 9:  BONUS: Add Border to an image
 10: Test all functions
 11: Exit
please make your choice:
```

- Note: Your program should respond as “Image is not in the program yet” if the user want to choose option 3~9 before using option 1 to read the image.

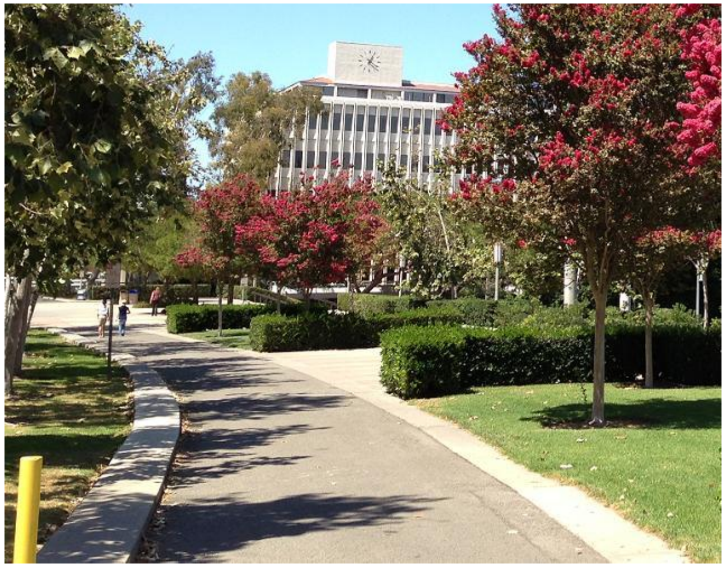
Black & White

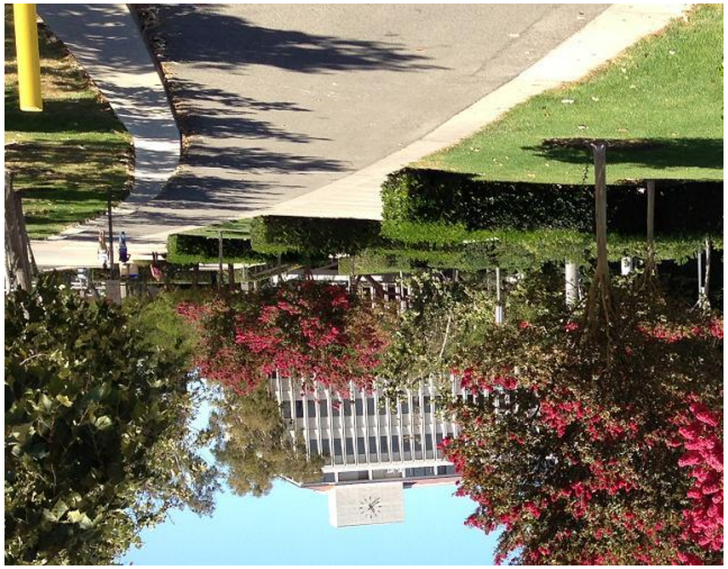


- Pseudo Code:

```
For all pixels in the picture
  average= R + G + B / 3
  R = G = B = average
```

VERTICALLY FLIP

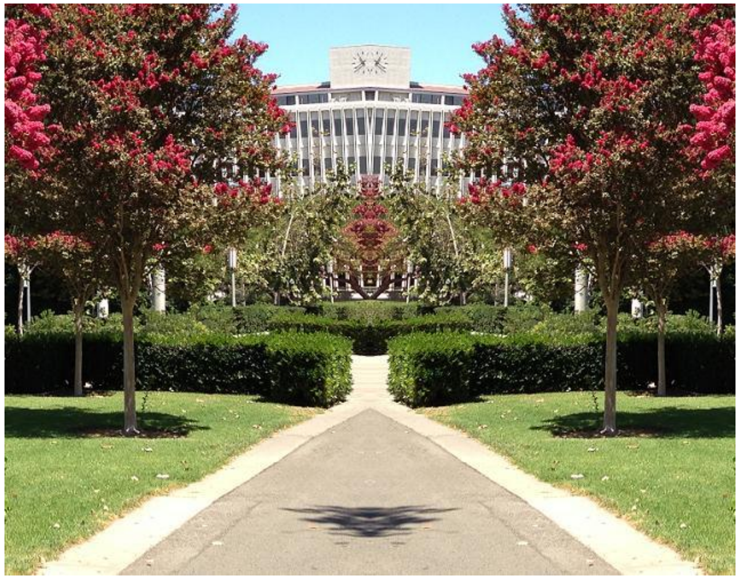




- For all pixels in the upper half picture, swap the color with the pixel in the lower half

1 2 3 4 5	3 4 5 6 7
0 1 2 3 4	0 1 2 3 4
3 4 5 6 7	1 2 3 4 5

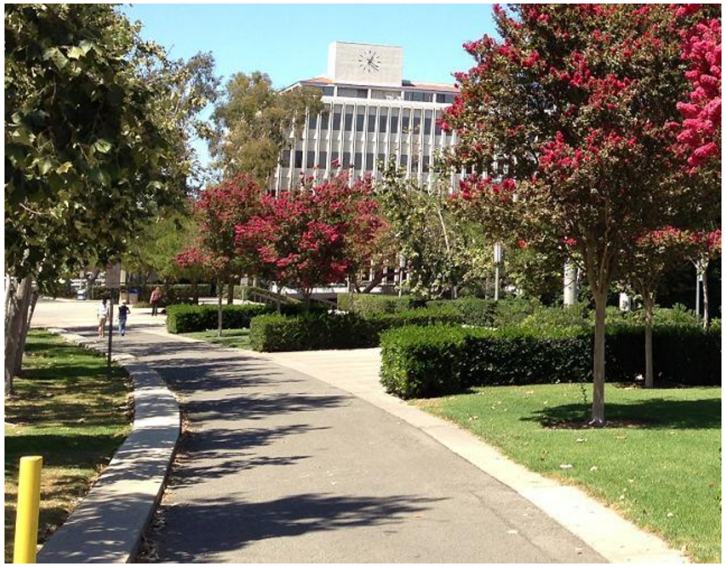
Horizontally Mirror

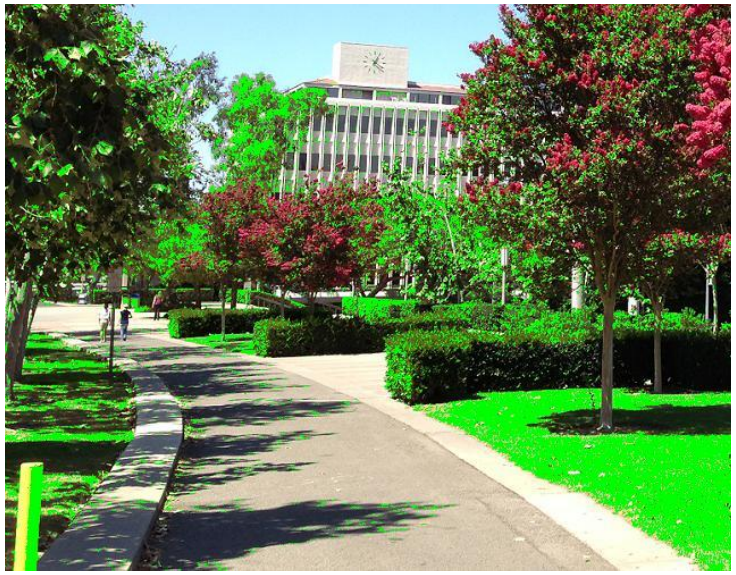


- For all pixels in the left half of the picture, replace the color to the color of pixel in the right half.

$$\begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \\ 4 & 3 & 2 & 1 & 0 \\ 3 & 4 & 5 & 6 & 7 \end{array} \quad \begin{array}{ccccc} 5 & 4 & 3 & 4 & 5 \\ 0 & 1 & 2 & 1 & 0 \\ 7 & 6 & 5 & 6 & 7 \end{array}$$

Color Filter

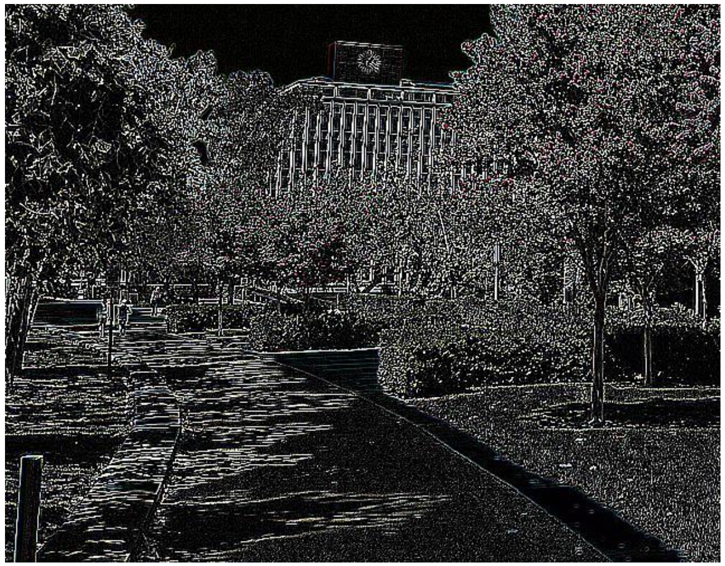




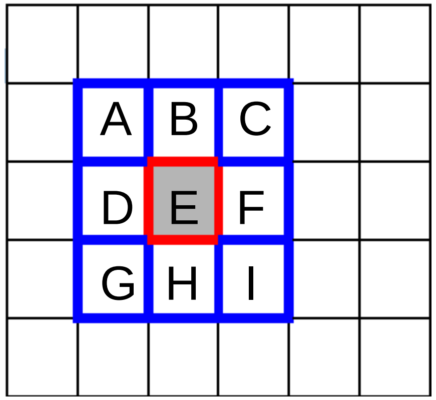
```
For all pixels in the picture
   if (R in the range of [target_r - threshold, target_r + threshold]) and
      (G in the range of [target_g - threshold, target_g + threshold]) and
      (B in the range of [target_b - threshold, target_b + threshold])
         R = replace_r ;
         G = replace_g ;
         B = replace_b ;
   else
         keep the current color
```

target_r = 180   replace_r = 0
target_g = 180   replace_g = 255
target_b = 50   replace_b = 0
Threshold = 70

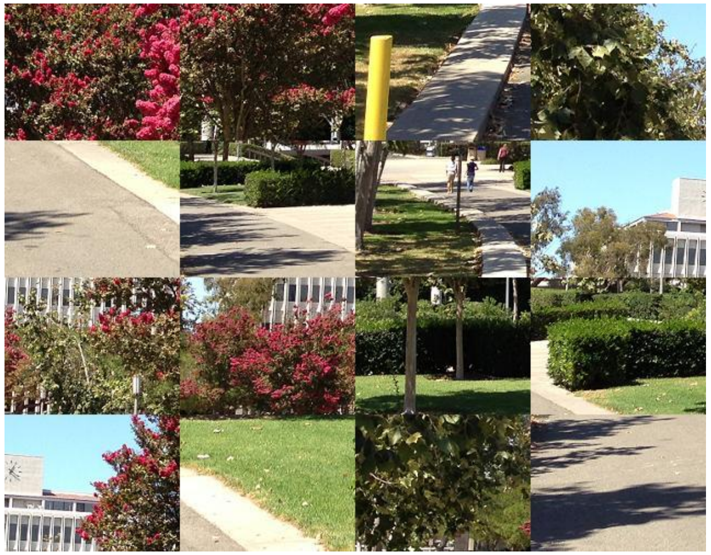
EDGE



- Set the pixel's color at E with equation: new_E = 8*E – A – B – C – D – F – G – H – I
- Use temporary array to avoid computing with containmiated color intensities.
- Pixels on the corners and the edges have fewer neighbors.
- new_E should be in the range [0, 255]



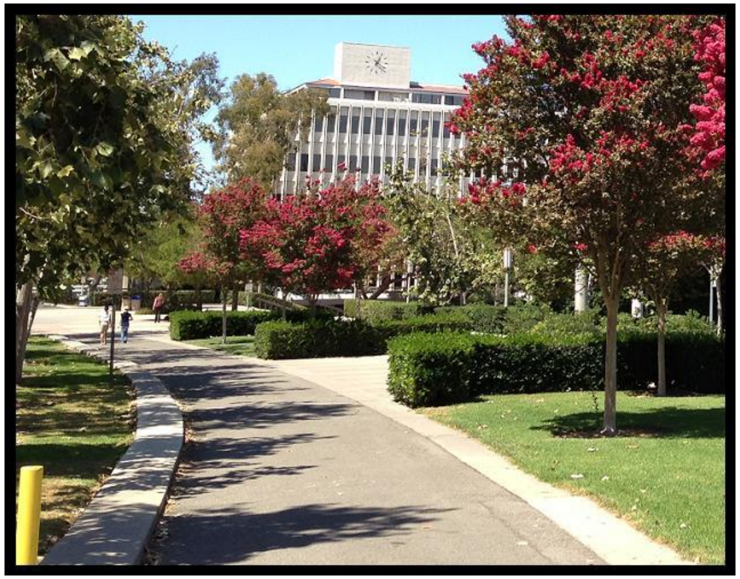
Shuffle



- Segment the image into 4X4 sub block (16 equally sized sub blocks)

```
For all un swapped image sub blocks
    block_1 =  random un swapped image sub block
    block_2  =  random un swapped image sub block
    Swap(block_1, block_2)
    mark block_1 swapped
    mark block_2 swapped
```

BONUS : ADD BORDER



```
void AddBorder(unsigned char R[WIDTH][HEIGHT], unsigned
char G[WIDTH][HEIGHT], unsigned char B[WIDTH][HEIGHT],
char color[SLEN], int border_width)
```

Initial Setup

- `mkdir hw2`
- `cd hw2`
- `cp /users/grad2/doemer/eecs22/hw2/PhotoLab.c .`
- `cp /users/grad2/doemer/eecs22/hw2/RingMall.ppm .`

Provided Function

```
#define WIDTH  640       /* Image width */
#define HEIGHT 500       /* image height */
#define SLEN    80       /* maximum length of file names */

int main()
{
 /*
  * Two dimensional arrays to hold the current image data
  * One array for each color component
  */
    unsigned char   R[WIDTH][HEIGHT];
    unsigned char   G[WIDTH][HEIGHT];
    unsigned char   B[WIDTH][HEIGHT];
/*  Please replace the following code with proper menu  */
/*  with function calls for DIP operations              */
    AutoTest(R, G, B);
/*  end of replacing*/

    return 0;
}
```

Provided Function

- Image Input / Output

  -
    ```
    int ReadImage (char fname[SLEN],
                   unsigned char R[WIDTH][HEIGHT],
                   unsigned char G[WIDTH][HEIGHT],
                   unsigned char B[WIDTH][HEIGHT]) ;
    ```

  -
    ```
    int SaveImage (char fname[SLEN],
                   unsigned char R[WIDTH][HEIGHT],
                   unsigned char G[WIDTH][HEIGHT],
                   unsigned char B[WIDTH][HEIGHT]) ;
    ```
  - Arguments are passed to the function by reference.
  - EECS10 lecture slide lecture 7.2 page 2 for "pass by reference"

- Use `scanf("%s", fname)` to input file name

Provided Function

- Aging function – as the sample of DIP function

  -
    ```
    void Aging(unsigned char R[WIDTH][HEIGHT],
               unsigned char G[WIDTH][HEIGHT],
               unsigned char B[WIDTH][HEIGHT])
    {
      int x, y;
      for( y = 0; y < HEIGHT; y++ )
        for( x = 0; x < WIDTH; x++ ) {
          B[x][y] = ( R[x][y]+G[x][y]+B[x][y] )/5;
          R[x][y] = (unsigned char) (B[x][y]*1.6);
          G[x][y] = (unsigned char) (B[x][y]*1.6);
      }
    }
    ```

Provided Function

- AutoTest
  - test all DIP functions and save the processed image.

  -
    ```
    void  AutoTest (unsigned char R[WIDTH][HEIGHT],
                    unsigned char G[WIDTH][HEIGHT],
                    unsigned char B[WIDTH][HEIGHT])
    ```

```
{
  char  fname[SLEN] = “RingMall";
  char  sname[SLEN];

  ReadImage(fname, R, G, B);
  Aging(R, G, B);
  strcpy(sname, “aging");
  SaveImage(sname, R, G, B);
  printf(“Aging tested!\n\n”);

  /*
    Filling this part with the call to your DIP functions
  */
}
```

Compile/Execute/View/Submit Your Work

- For each DIP options and the AutoTest, a corresponding function has to be created for it.
- Compile your program
  - `gcc Photolab.c -o Photolab -Wall -ansi`
- View your processed image
  - *http://newport.eecs.uci.edu/~youruserid*
- Name your files `bw`, `vflip`, `hmirror`, `colorfilter`, `edge`, `shuffle`, and `border` for the corresponding function.
- Required files : `Photolab.c`, `Photolab.txt`, and `Photolab.script`.
- In the Photolab.script, the following commands are required.
  - Compilation of the Photolab.c
  - Run your Photolab
  - Use option “`Test all functions`” to test all DIP functions.