

CSE 241 Programming Assignment 2

This assignment is the first in a series of assignments which are all going to be about classes. In the end, you will have an image editor which is implemented with basic object oriented design principles kept in mind.

You should utilize object oriented principles as much as possible. Your submission may not be accepted if you ignore this advice.

Image Editor

- You can create a class which represents the whole image editor program. Your program won't have a GUI yet and you need to do is decide how to organize events and data in your program.
- Implement a menu system.
- Implement member functions for reading image files and writing image data to files.
- Currently, your image editor program opens only one image and operates on it. Later, you will extend your program to open more than one image at a time.
- Implement grayscale conversion function.

Image File Format

Your program will basically work on PPM format(Ascii format). Particularly P3 format. Read the description here: <http://netpbm.sourceforge.net/doc/ppm.html>

Many image editors can read and write ppm files. I advise you to use GIMP. Use Ascii format while exporting, otherwise you cannot read it.

Menu Of Your Program

The initial version of your program will have the following main menu structure:

Main Menu

```
0 - Exit          -----> Exits the program without saving the image data
1 - Open An Image(D) -----> A dialog entity in order for user to enter a file name.
2 - Save Image Data(D) -----> A dialog entity for saving the current state of the image data.
3 - Scripts(D)    -----> Various scripts
```

OPEN AN IMAGE MENU

```
0 - UP
1 - Enter The Name Of The Image File
```

SAVE IMAGE DATA MENU

```
0 - UP
1 - Enter A File Name
```

SCRIPTS MENU

```
0 - UP
1 - Convert To Grayscale(D)
```

CONVERT TO GRAYSCALE MENU

```
0 - UP
1 - Enter Coefficients For RED GREEN And BLUE Channels.
```

If a menu item has a (D) at the end, that means it is a dialog. If user enter the particular menu number, A new menu dialog will be shown.

Example

Suppose that you have the following file.

```
• test.ppm

P3
4 4
255
0 0 0 100 0 0      0 0 0 255 0 255
0 0 0 0 255 175    0 0 0 0 0 0
0 0 0 0 0 0        0 15 175 0 0 0
255 0 255 0 0 0    0 0 0 255 255 255
```

NOTE: > is a part of the terminal environment. **DO NOT PRINT IT**

Start your program

```
> ./myprogram
> MAIN MENU
> 0 - Exit
> 1 - Open An Image(D)
> 2 - Save Image Data(D)
> 3 - Scripts(D)
> 1 -----> User inputs 1 and presses RETURN
> OPEN AN IMAGE MENU
> 0 - UP -----> If user enters 0, MAIN MENU will be printed.
> 1 - Enter The Name Of The Image File
> 1 -----> User inputs 1 and presses RETURN
> test.ppm -----> User inputs test.ppm and presses RETURN
> OPEN AN IMAGE MENU -----> Your program reads test.ppm and prints the current dialog menu
> 0 - UP -----> If user enters 0, MAIN MENU will be printed.
> 1 - Enter The Name Of The Image File
> 0 -----> User inputs 0 and presses RETURN
> MAIN MENU -----> MAIN MENU is printed again
> 0 - Exit
> 1 - Open An Image(D)
> 2 - Save Image Data(D)
> 3 - Scripts(D)
.
.
.
```

Script: Convert To Grayscale

This script creates the following dialog:

```
CONVERT TO GRAYSCALE MENU
0 - UP
1 - Enter Coefficients For RED GREEN And BLUE Channels.
```

If user enters 1, then user can enter three float numbers which are in the range [0,1). You should check for any errors. If there is an error, ask user to re-enter the coefficients. Example input:

```
0.33 0.10 0.2
```

These numbers are coefficients (c_r, c_g, c_b) for RED, GREEN and BLUE channels. After getting the coefficients (c_r, c_g, c_b) you can convert each color to grayscale using the following formula:

```
RED = (c_r * RED) + (c_g * GREEN) + (c_b * BLUE)
GREEN = (c_r * RED) + (c_g * GREEN) + (c_b * BLUE)
BLUE = (c_r * RED) + (c_g * GREEN) + (c_b * BLUE)
```

You just need to update the values in the image representation these new values. If the new values are greater than 255, then that means they are saturated. Leave them as 255.

Saving Image Data

```
SAVE IMAGE DATA MENU
0 - UP
1 - Enter A File Name
```

If user enters 1, then user can enter a filename. Example: Assume user enters `output.ppm`. In this case, image data in the memory will be saved as `output.ppm`.

Remarks

- Error checking is important.
- Your program should be immune to the whitespace before any user input.
- Do not submit your code without testing it with several different scenarios.
- Write comments in your code. Extensive commenting is required. Comment on every variable, constant, function and loop. (10pts)
- Do not use `#Define` and define macros. Instead use `constant` keyword and define constant variables. If you use macros, you will lose 5 points for each of them.
- Everything happens through `stdin` and `stdout`.
- Be very careful about the input and output format. Don't print anything extra(including spaces).
- You have to use `Class`. You should not use inheritance and other advanced OOP mechanisms yet.
- You can use `std::string` and `std::vector`.

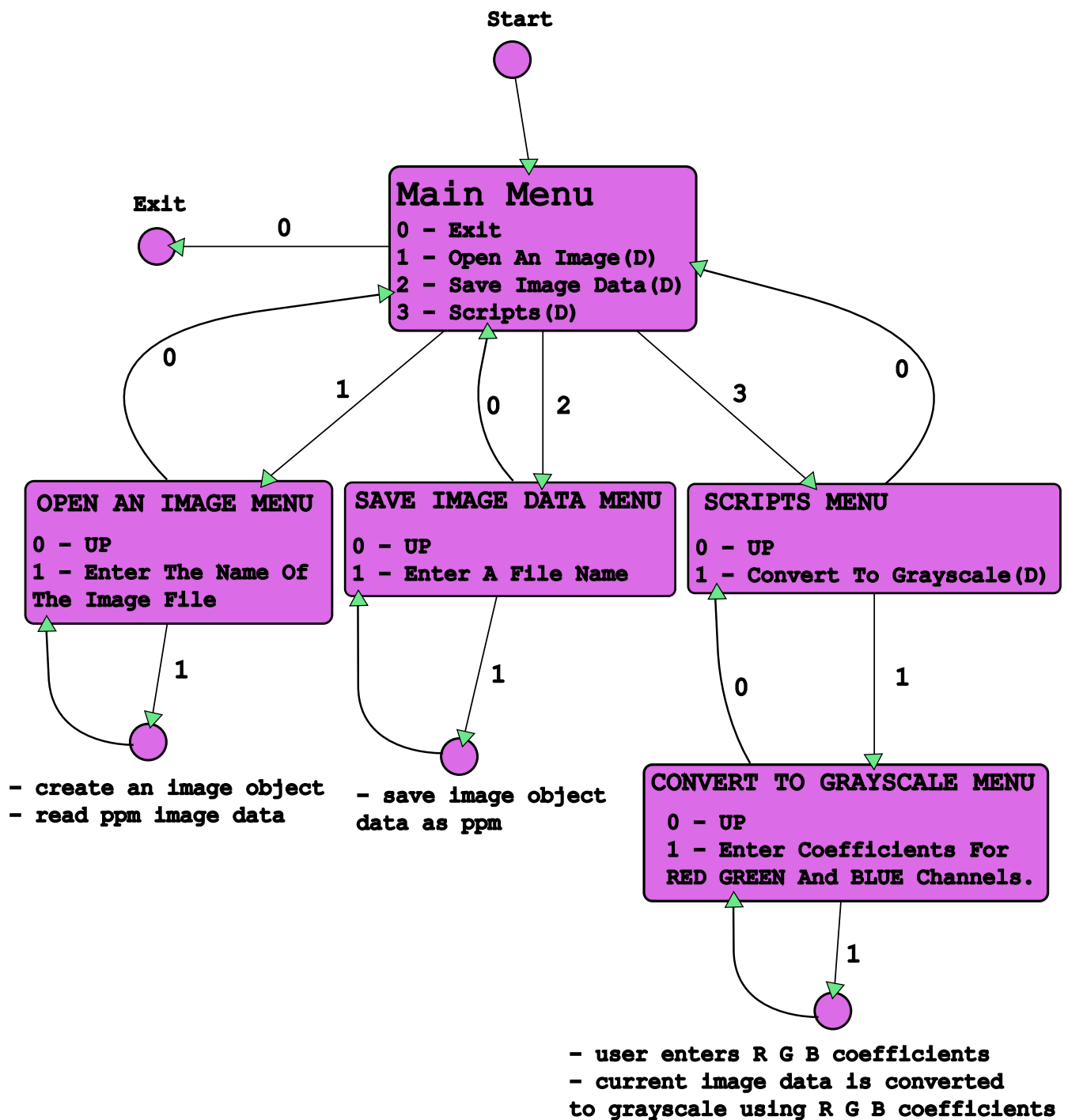


Figure 1: Program Flow