Lab 4: Ifs and Loops...

Due date: Friday, October 23, 11:59pm.

Note: October 23 **is my furlough day.**

Lab Assignment

Assignment Preparation

Lab type. This is an **pair programming lab**. For this lab, you get to select your own partner. The only rule is that your partner **must be different** than your Lab 3 partner. Additionally, **no student who was on a three-person pair for Lab 3** can be on a three-person pair in Lab 4. See the pair programming handout from the first day of classes for more information on what is expected of you.

Collaboration. Students work in pairs, and it is considered cheating, if members of the team (pair) do not work together. Communication between pairs during lab time is allowed, but no direct sharing of code is allowed.

Purpose. The lab allows you to practice the use of loops.

Programming Style. All submitted C programs must adhere to the programming style described in detail at

http://users.csc.calpoly.edu/∼cstaley/General/CStyle.htm

When graded, the programs will be checked for style. Any stylistic violations are subject to a 10% penalty. Significant stylistic violations, epsecially those that make grading harder, may yield stricter penalties. Also note the the Lab 2 requirement for the content of the header comment in each file you submit applies **to each assignment** (lab, programming assignment, homework) in this course.

Testing and Submissions. Any submission that does not compile using the

```
gcc -ansi -Wall -Werror -lm
```

compiler settings will receive an automatic score of 0.

For each program you have to write, you will be provided with instructor's executable and with either a battery of tests or the instructor's output (PPM file). The programs you submit must pass all tests made available to you. You can check whether or not a program produces correct output by running the instructor's executable on the test case, then running yours, and comparing the outputs.

Program Outputs must co-incide. Any deviation in the output is subject to penalties. **PLEASE, USE BINARY EXECUTABLES PROVIDED BY THE INSTRUCTOR!**. The exception is made in case of floating point computations leading to differences in the last few decimal digits.

Note: The difference between PPM files cannot be easily checked using Linux `diff` command, however, it can be observed in a straighforward manner by loading two images into an image browser and viewing them in rapid succession.

Please, make sure you test all your programs prior to submission!

Portable Pixel Maps

PPM Format Explanation

Portable Pixel Map (`.ppm`) file format is a simple format for storing graphical images. Files in this format can easily be created using C programs.

Basics. An computer image is a two-dimensional grid of pixels. Each pixel represents the smallest undivisible part of the computer screen. An image file is an assignment of color to each pixel. Pixels are referred to by their Cartesian coordinates. The top left corner of an image has the coordinates (0,0). The bottom right corner has the coordinates (n, m) where n is the width of the image in pixels and m is the height of the image in pixels.

Colors. Portable pixel map files use RGB (Red, Green, Blue) color format to represent the color of each pixel. In RGB format, a color of a single pixel is separated into three components: the red component, the green component and the blue component. The final color of an RGB pixel is determined by combining the Red, Green and Blue components into a single color.

In our course, all individual RGB component intensities range from 0 (not visible) to 255 (highest intensity) and are represented as integer numbers. RGB color (0,0,0) is black, RGB color (255,255,255) is white. The table below contains the list of colors used in this lab and their RGB values.

Color	RGB Red	RGB Green	RGB Blue
black	0	0	0
white	255	255	255
red	255	0	0
green	0	255	0
blue	0	0	255
yellow	255	255	0
purple	255	0	255
orange	255	128	0

File format. There are two PPM formats: a "raw" PPM file and a "plain" (ASCII) PPM file. ASCII PPMs are human-readable, but they take too much space. Raw PPMs are smaller in size, but cannot be read by a human. In this lab you will be generating raw PPM files.

From http://netpbm.sourceforge.net/doc/ppm.html (with some modifications):

Each PPM image consists of the following:

1. *A "magic number" for identifying the file type. A ppm image's magic number is the two characters* `"P6"`.
2. *Whitespace (blanks, TABs, CRs, LFs).*
3. *A width, formatted as ASCII characters in decimal.*
4. *Whitespace.*
5. *A height, again in ASCII decimal.*
6. *Whitespace.*
7. *The maximum color value (Maxval), again in ASCII decimal. Must be less than 65536 and more than zero.*
8. *A single whitespace character (usually a newline).*
9. *A raster of Height rows, in order from top to bottom. Each row consists of Width pixels, in order from left to right. Each pixel is a triplet of red, green and blue intensities, in that order*

Representing Colors in C. Outputing raw PPM files is actually quite simple. The idea is to use `unsigned char` variables to store information about RGB intensities.

Variables of type `char` and `unsigned char` are treated by C both as a character and as a number in the range -128 – 127 or 0 — 255 respectively. The following code outputs an RGB triple to stdout.

```
unsigned char Rcolor, Bcolor, Gcolor;
Rcolor = 255;
Bcolor = 0;
Gcolor = 128;

printf("%c%c%c", Gcolor, Bcolor, Rcolor);
```

Viewing PPM files

On our Linux system, PPM files can be viewed using the default picture viewer (e.g., when double-clicking the PPM file icon in the directory explorer). The default picture viewer is `eog`, a.k.a., Eye of Gnome. To use it from command line, type:

```
> eog <fileName>.ppm &
```

On Windows, standard Windows picture viewers do not recognize PPM format. However, freeware picture viewers exist. One such viewer, `XnView` can be downloaded from http://www.xnview.com/en/xnview.html

Note, XnView is also available for MacOs, although I have no experience running it on Macintoshes.

General Instructions

1. All programs that output ppm images should use stdout. We will be creating files using output redirection.
2. For this assignment, use exactly the colors specified in the color table above.
3. All images you are asked to produce are also available from the course web page.

Image 1: Dutch Flag: netherlands.c

Your first image is the flag of The Netherlands. The flag of The Netherlands consists of three equal-sized horizontal fields. The colors (top-to-bottom) are `red`, `white`, `blue`. Your program shall output the .ppm file representing the Russia flag. Name your program `netherlands.c`.

Image dimenstions: 800×600.

Image 2: Italian Flag: italy.c

The image is the flag of Italy. The flag of Italy consists of three equal-sized vertical fields. The colors (left-to-right) are `green`, `white`, `red`.

Your program shall output the .ppm file representing the flag of Italy. Name your program `italy.c`.

Image dimenstions: 600×400.

Image 3: Swiss Flag: switzerland.c

The image is the flag of Switzerland. The flag is a square field colored `red` with a `white` cross in the center. Use the following dimensions:

Feature	Size/Dimensions
Image size:	500 × 500
Cross bars:	300 × 100

Image 4: Finnish flag: finland.c

The image is the flag of Finland. The flag is a white field with a `blue` cross on it.

Image dimensions:	900 × 550.
Cross bar width:	150.
Left rectangles (top and bottom):	250 × 200
Right rectangles (top and bottom):	500 × 200

Image 5: Flag of Laos: laos.c

The image is the flag of Laos. The flag of Laos consists of three horizontal stripes: `red` at the top and bottom, `blue` in the middle, with a white circle centered inside the blue stripe. The dimensions are as follows:

Image dimensions:	600×400
Red stripe height (top and bottom):	100
Blue stripe height:	200
Circle radius:	80
Circle center:	$(300, 200)$

Image 6: Flag of Greenland: greenland.c

The image is the flag of Greenland. The flag consists of two equal horizontal stripes: `white` at the top and `red` at the bottom, with a circle on top. The center point of the circle is shifted to the left. The top half of the circle is `red` and the bottom half of the circle is `white` making the top and the bottom halves of the flag negative images of each other. The official flag description is:

> *The flag is 12 parts by 18, the white and red stripe are both 6 parts. The centre of the circle is set 7 parts from the hoist along the dividing line between the white and red, the radius being 4 parts. The upper part of the circle is red, the lower white.*

Image dimensions:	600×400
Center of the circle:	200×200
Radius of the circle:	125

Image 7: Flag of the Czech Republic: czech.c

The image is the flag of the Czech Republic. The flag consists of three fields: a `blue` *equilateral triangle* on the left, and two horizontal stripes of equal size and shape: `white` at the top and `red` at the bottom, filling out the rest of the space.

Image dimensions: 400×800

Image 8: Flag of Scotland: scotland.c

The image is the flag of Scotland. The flag, otherwise known as "St. Andrew's flag" is a *saltire*, i.e., a *diagonal cross*. On the flag of Scotland, the `white` saltire splits the `blue` field into four triangles.

Image dimensions	500×300
Horizontal offset (white)	50
Vertical offset (white)	30

Note: The horizontal and vertical offsets specify respectively the number of `white` pixels in the first row and the first column from the (1,1) point and to the first `blue` pixel.

Submission.

Files to submit. You shall submit **nine** files:

```
team.txt,
netherlands.c,
italy.c,
switzerland.c,
finland.c,
laos.c,
greenland.c,
czech.c,
scotland.c
```

`team.txt` file shall contain the name of the team and the names of the two team members in each pair, and the Cal Poly IDs of each. E.g, if I were on the team with Dr. John Bellardo, my `team.txt` file would be

```
Go, Poly!
John Bellardo, bellardo
Alex Dekhtyar, dekhtyar
```

No other files can be submitted. In fact, if you submit *other* file, or submit one of the three files about with an *incorrect filename*, you will receive an email informing you about a submission error, and asking you to resubmit.

Files can be submitted one-by-one, or all-at-once.

Submission procedure. Please note the changes!!!

You will be using `handin` program to submit your work. The procedure is as follows:

- `ssh` to `vogon` (`vogon.csc.calpoly.edu`).
- Execute the following submission command:

```
> handin dekhtyar lab04  <your files go here>
```

`handin` is set to stop accepting submissions 24 hours after the due time.

Grading

Each program is worth 12.5% of the lab grade.

Any submitted program that does not compile earns 0 points.

All programs will be checked for style conformance. Any style violation will be noted. The program will receieve a 10% penalty. (In particular, declare all "magic" numbers as constants).

Appendix A. Testing

Instructor's Images. The PPM files generated by the instructor's programs are provided to you on the Lab 4 web page.