Lab 8: Functions, structures, and pointers

Goals

Demonstrate proficiency in creating functions that manipulate data passed to them via pointers to structures and tables of structures. Understand how to return a pointer to a structure.

Background

X Windows color definitions

The Unix graphical user interface (GUI) is built on the X Window system. X Windows was developed at MIT in the 1980's. A table of standard (R, G, B) color values and color names is typically included in an X windows distribution, and it has the following format:

```
255 235 205  BlanchedAlmond
240 255 240  honeydew
 25  25 112  MidnightBlue
123 104 238  MediumSlateBlue
188 143 143  RosyBrown
 65 105 225  royal blue
255 165   0  orange
 92 172 238  SteelBlue2
255 218 185  peach puff
 85 107  47  dark olive green
```

For each color the first three values are the (R, G, B) intensities. Note that the color name can *consist of multiple words*.

The *fgets()* function

The *fgets()* function will consume text from an input file until the first of the following three things occurs:

1. A newline character (\n) is found.
2. End of file is reached.
3. The the buffer provided is full.

Because the color names may consist of multiple words that comprise the remainder of the line on which they reside *fgets()* provides a handy way to read them. The prototype is shown here:

```
char *fgets(char *s, int size, FILE *stream);
```

The parameter *s* is a pointer to the input buffer (an array of characters), *size* is the size of the buffer, and stream is the FILE structure pointer. The value returned by the function is a copy of *s.*

If the operation halts because a newline is found the \n will be placed in the buffer. A byte having the value 0 will *always be placed in the buffer* just past the last character read so that the array may always be treated as a proper null terminated C string.

Example:

```
char name[60];
fgets(&name[0], 60, stdin);
```

Color Luminance

The luminance of a color is a measure of its "brightness" as perceived by a human viewer with normal vision. One commonly used standard for quantifying luminance was developed by the NTSC (national television systems committee) in 1953 as a component of the standard way for encoding color television signals.

Y (Luminance) = (Red x 0.30) + (Green x 0.59) + (Blue x 0.11)

The weights (0.30, 0.59, 0.11) are not equal because the human eye is more sensitive to colors in the middle of the visible spectrum (green) than it is to colors in the low end (red) or high end (blue). The standard way of producing a gray scale image from an (R, G, B) color image is to use luminance as the value of the gray scale pixel. Because the weights sum to 1, it is guaranteed that the gray pixel will also take on values in the range [0, 255].

Assignment:

Create a lab8 directory and copy the contents of the lab/lab8 directory to your lab8 directory. The *main()* function is found in main.c and is already complete. Your mission is to *complete three functions* in the color.c file. These functions will manipulate elements of the following two structure types:

```
typedef struct pixel_type
{
   unsigned char r;
   unsigned char g;
   unsigned char b;
}  pixel_t;

typedef struct color_type
{
   char name[64];            // name of the color
   struct pixel_type pixel; // rgb values
   unsigned char luminance; // luminance of color
}  color_t;
```

The `int get_color(FILE *in, color_t *color)` function should perform the following operations:

(1) Use *fscanf() to r*ead (r, g, b) pixel value from the file *in* into the three components of the *pix* element of the color_t structure pointed to by *color*. If the three (r, g, b) values are not successfully read, then the function should return(-1) and not try to read the color name.

(2) If the (r,g, b) pixel values are read successfully, use the *fgets()* function to read the color name from the file *in* into the *name* element of the structure pointed to by *color* and *return(0);*

The *void compute_luminance(color_t *color)* function should perform the following operation:

(1) Use the NTSC formula to compute the luminance of the pix element of the structure pointed to by *color* and store the value in the *luminance* element of the same structure.

The *color_t *find_brightest(color_t *colors, int count)* will be passed a pointer *colors* to an array of structures of type *color_t.* The value *count* will be the number of *color_t* structures in the array. The function should perform the following operation:

(1) Create a local variable called *color_t *max;* that will contain the address of the element of the array having maximum luminance. As usual initialize *max* to the address of the first element in the *colors* array.

(2) In a *for* loop, process the remaining elements in the array and whenever an element has greater luminance that the element pointed to by *max*, *max* should be updated to have the address of the element with the greater luminance.

(3) The function should return the pointer variable *max.*

Turn In Work

Show your TA that you completed the assignment. Then turn in your image.c program using the command:

sendlab.101.*section_number* 8 color.c

DO NOT TURN IN main.c, color.h input data files.