Homework 6: Introduction to Pointer

Released Monday 10/03/16

Due Friday 11:59pm

Goals:

- Learn how to use type casting in C programming
- Learn how to manipulate pointers in C programming
- Learn about big endian and little endian

Introduction to Pointer:

In this homework, you will use pointer to manipulate variables in C. Anything you did in previous lab, you could do it in a pointer version. Pointer is a C's special way to access data or variables stored in memory. In task one, you need to increase intensity of an image by using pointers. And in task two, you need to use pointer to check the endianness of PI.

Task 1: Increase Intensity

In this task, you will need to get an image from the image file and increase the intensity of the image. **You will save this image as filename_int.jpg where for example, the image name could be filename is ../images/filename.type. The image does not need to start out as a jpg.**

To do this you will have to first open up the image. Next, you will have to get the pixel(color) like you did in homework 3. Next convert this pixel from type int to be able to access each color individually in a char pointer. From there, you can implement your program to change the intensity of each color by multiplying it by the change in intensity. Next, make sure that you convert the pointer back. Make sure you update the pixel array. Finally save the image as filename_int.jpg.

Note: you must use type conversion, you cannot use bit manipulation, like in homework 3.

Note: Please do this task on lab machines, or ssh (PuTTY) to data.cs.purdue.edu

Given:

- **task1/Makefile** - compiles source files (imageio.c, main.c, intensity.c), links image libraries, and generate the final executable file--**intensity**.
- **task1 /intensity.c -** the function you should complete

- t**ask1/intensity.org** - the functions shows what should happen when you run the function takes intensity.org filename intensity_change. Example intensity.org ../images/LWSN.jpg 1.1
- **lib/imageio.o** - Image I/O library function definitions
- **lib/imageio.h** - Header of the image I/O functions
- **task1/test** - test script used to test your program. To test your program, simply run command **./testall**

Steps:

1. Set up a way to receive each pixel from an image, like you did in homework 3.
2. Convert each pixel to a char pointer.
3. Multiply each part of the Blue,Green, Red by the float intensity_change
4. Convert back to type int.
5. Update the pixel array.
6. After changing every pixel, save the image to filename_int.jpg
7. Run **make** to compile your program.
8. Run the ./testall to make sure your program runs correctly

Task 2: Check the Endianess of PI

Integers can be represented in memory in two ways: Little-Endian and Big-Endian. So does float. In Little- Endian, the least significant byte of the integer is stored at the lowest address in memory. In Big-Endian, the least significant byte is stored at the highest address in memory. In this task, you need to complete **isBigEndian** function given in **endian.c** file, so the function returns 1 if the given input is Big-Endian and returns 0 if it's Little-Endian. In this task, you need to test the endianness of PI(3.1415926….). But we will pass PI's hexadecimal format either in big endian or little endian as input. So take it easy.

So what is Big Endian and Little Endian?

Suppose we are on a 32-bit machine. Let x be 1.

If it is little endian, the `x` in the memory will be something like:

```
      higher memory
         ----->
   +----+----+----+----+
   |0x01|0x00|0x00|0x00|
   +----+----+----+----+
   A
   |
  &x
```

Bits start from lower memory to higher memory.

If it is big endian, it will be:

```
  +----+----+----+----+
  |0x00|0x00|0x00|0x01|
  +----+----+----+----+
  A
  |
 &x
```

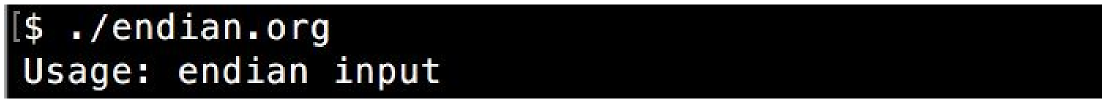
Bits start from higher memory to lower memory

Given:

- **endian.c** - implement your **isBigEndian** function here
- **endian.org** - reference executable file showing how the program should work
- **testall** - test script testing your program
- **Makefile** - to compile the source files to the executable file **endian**

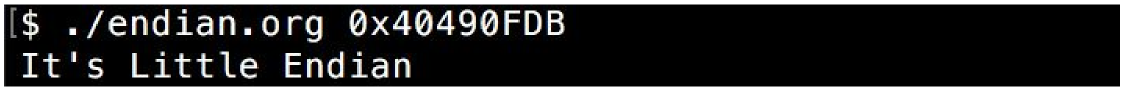
Demo:

Usage:



```
[$ ./endian.org
 Usage: endian input
```

Run with Littlen Endian



```
[$ ./endian.org 0x40490FDB
 It's Little Endian
```

Steps:

1. Implement your program to check endianess.
2. Run **make** to compile your program.
3. Run **testall** to check if your program works as expected

Submission:

These are the files that need to be in the Vocareum "work" directory for this assignment:

intensity.c
endian.c