Assignment 5 - Color Blindness Simulator

Chucheng Xie

CSE 13S – Spring 2023

Purpose

In this assignment, we will write an image-processing program that allows someone with normal color vision to appreciate the range of colors experienced by someone who has deuteranopia.

Program Design

Pseudocode:

io.c:

```
include io.h, fcntl.h, stdio.h, stdlib.h, unistd.h

Buffer *read_open(const char *filename) {
    int fd = open(filename, 0_RDONLY);
    if (fd < 0) { return NULL; }
    Buffer *buf = malloc(sizeof(Buffer));
    if (buf == NULL) {
        close(fd);
        return NULL;
    }
    buf -> fd = fd;
    buf -> offset = 0;
    buf -> num_remaining = 0;
    return buf;
}

void read_close(Buffer **pbuf) {
    if (pbuf == NULL || *pbuf == NULL) { return; }
    close((*pbuf) -> fd);
    free(*pbuf);
    *pbuf = NULL;
}

bool read_uint8(Buffer *buf, uint8_t *x) {
    if (buf -> num_remaining == 0) {
        ssize_t rc = read(buf -> fd, buf -> a, sizeof(buf -> a));
        if (rc < 0) { return false; }
        if (rc == 0) { return false; }
        buf -> num_remaining = rc;
        buf -> offset = 0;
    }
    *x = buf -> a[buf -> offset++];
    buf -> num_remaining--;
    return true;
}

Buffer *write_open(const char *filename) {
    int fd = creat(filename, 0664);
    if (fd < 0) { return NULL; }
    Buffer *buf = malloc(sizeof(Buffer));
    if (buf == NULL) { close(fd); return NULL; }
    buf -> fd = fd;
    buf -> offset = 0;
    buf -> num_remaining = 0;
    return buf;
}

void write_close(Buffer **pbuf) {
    if (pbuf == NULL || *pbuf == NULL) { return; }
    write_uint8(*pbuf, 0);
    close((*pbuf) -> fd);
    free(*pbuf);
    *pbuf = NULL;
```

```
}

void write_uint8(Buffer *buf, uint8_t x){
    buf -> a[buf -> offset++] = x;
    if (buf -> offset == BUFFER_SIZE) {
        uint8_t *start = buf -> a;
        int num_bytes = buf -> offset;
        do {
            ssize_t rc = write(buf -> fd, start, num_bytes);
            if (rc < 0) { return; }
            start += rc;
            num_bytes -= rc;
        } while (num_bytes > 0);
        buf -> offset = 0;
    }
}
```

bmp.c:

```
include io.h, bmp.h, stdio.h, stdlib.h

typedef struct color {
    uint8_t red;
    uint8_t green;
    uint8_t blue;
} Color;

typedef struct bmp {
    uint32_t height;
    uint32_t width;
    Color palette[MAX_COLORS];
    uint8_t **a;
} BMP;
```

```
void bmp_write(const BMP *bmp, Buffer *buf);
    int32_t rounded_width = (width + 3) & ~3
    int32_t image_size = height * rounded_width
    int32_t file_header_size = 14
    int32_t bitmap_header_size = 40
    int32_t num_colors = 256
    int32_t palette_size = 4 * num_colors
    int32_t bitmap_offset = file_header_size + bitmap_header_size + palette_size
    int32_t file_size = bitmap_offset + image_size
use the giving list table
    for i from 0 to num_colors - 1
        8 | bmp -> palette[i].blue
        8 | bmp -> palette[i].green
        8 | bmp -> palette[i].red
        8 | 0
    for y from 0 to bmp -> height - 1
        for x from 0 to bmp -> width - 1
            8 | bmp -> a[x][y]
        for x from bmp -> width to rounded_width - 1
            8 | 0
```

```
BMP *bmp_create(Buffer *buf) {
    BMP *bmp = calloc(1, sizeof(BMP));
    use the giving list table
    verify list variables
    for (int i = 0; i < num_colors; i++) {
        8 | bmp -> palette[i].blue
        8 | bmp -> palette[i].green
        8 | bmp -> palette[i].red
        8 | (skip one byte)
    }
}
```

```
void bmp_free(BMP **bmp);
```

```
    uint32_t rounded_width = ((*bmp) -> width + 3) & ~3
    for i from 0 to rounded_width - 1
        free((*bmp) -> a[i])
    free((*bmp) -> a)
    free(*bmp)
    *bmp = NULL
```

```
void bmp_reduce_palette(BMP *bmp);
// It is given in the pdf
```

colorb.c

```
set command line options
read the input BMP file
bmp_create()
color transformation use bmp_reduce_palette()
write the transformed BMP to the output file
bmp_write()
free all memory
```

Result

we can use the command line options to run the program like:

```
./colorb -i bmps/apples-orig.bmp -o bmps/apples-colorb.bmp
./colorb -i bmps/cereal-orig.bmp -o bmps/cereal-colorb.bmp
./colorb -i bmps/froot-orig.bmp -o bmps/froot-colorb.bmp
./colorb -i bmps/produce-orig.bmp -o bmps/produce-colorb.bmp
```

In this assignment, we write an image processing program that enables people with normal color vision to appreciate the range of colors experienced by deuteranopia patients. This program is very fun and a very interesting experience.