lab14

```c
// EE231002 Lab14. Image Processing
// 108061213, 劉奕緯
// Dec. 28, 2019

#include <stdio.h>
#include <stdlib.h>

typedef struct sPIXEL {                         // a pixel
    unsigned char r, g, b;                      // three color components
} PIXEL;

typedef struct sIMG {                           // an image of PPM style
    char header[5];                             // header either P3 or P6
    int W, H;                                   // width and height of the image
    int level;                                  // intensity level of each color
    PIXEL **PX;                                 // 2-D array for all pixels
} IMG;

// This function opens the inFile, reads the image data and returns
// a pointer pointing to the newly created image data structure.
IMG *PPMin(char *inFile);
// This function writes the image pointed by p1 to the output file outFile.
void PPMout(IMG *p1, char *outFile);
// convert p1 to black and while, paste ee and nthu log, and make a box
// where (x1, y1) is low-left coner and where (x2, y2) is upper-light coner
IMG *PPMcvt(IMG *p1, IMG *ee, IMG *nthu, int x1, int y1, int x2, int y2);

int main(int argc, char *argv[])
{
    IMG *photo, *EE, *TH;                       // pointer to IMG, pic1, EE,
                                                // NTHU, respectively

    photo = PPMin(argv[1]);                     // input pic1,ppm
    EE = PPMin(argv[2]);                        // input EE.ppm
    TH = PPMin(argv[3]);                        // input NTHU.ppm
    PPMout(PPMcvt(photo, EE, TH, 1344, 1636, 1241, 1532), argv[4]);
                                                // convert and output to
                                                // argv[4] file
    return 0;
}
Need a blank line here.
// This function opens the inFile, reads the image data and returns
// a pointer pointing to the newly created image data structure.
IMG *PPMin(char *inFile)
{
    int i, j;                                   // index
    FILE *fin;                                  // address of input file
    IMG *image = (IMG *)malloc(sizeof(IMG));
```

```
48                                                  // pointer to this image
49
50      fin = fopen(inFile, "r");                   // open file
51      fscanf(fin, "%s", image->header);           // start reading
52      fscanf(fin, "%d%d", &image->W, &image->H);
53      fscanf(fin, "%d\n", &image->level);
54      // asking space for PIXELs
55      image->PX = (PIXEL **)malloc(image->W * sizeof(PIXEL*));
56      for (i = 0; i < image->W; i++)
57          image->PX[i] = (PIXEL *)malloc(image->H * sizeof(PIXEL));
58      // start reading PXIELs
59      for (j = 0; j < image->H; j++) {
60          for (i = 0; i < image->W; i++) {
61              fscanf(fin, "%c", &image->PX[i][j].r);
62              fscanf(fin, "%c", &image->PX[i][j].g);
63              fscanf(fin, "%c", &image->PX[i][j].b);
64          }
65      }
66      fclose(fin);
67      return image;
68  }
    Need a blank line here.
69  // This function writes the image pointed by p1 to the output file outFile.
70  void PPMout(IMG *p1, char *outFile)
71  {
72      FILE *fout;                                 // address of output file
73      int i, j;
74
75      fout = fopen(outFile, "w");                 // open file
76      fprintf(fout, "%s\n", p1->header);          // output in PPM format
77      fprintf(fout, "%d %d\n", p1->W, p1->H);
78      fprintf(fout, "%d\n", p1->level);
79      for (j = 0; j < p1->H; j++) {
80          for (i = 0; i < p1->W; i++) {
81              fprintf(fout, "%c", p1->PX[i][j].r);
82              fprintf(fout, "%c", p1->PX[i][j].g);
83              fprintf(fout, "%c", p1->PX[i][j].b);
84          }
85      }
86      fclose(fout);
87  }
    Need a blank line here.
88  // convert p1 to black and while, paste ee and nthu log, and make a box
89  // where (x1, y1) is low-left coner and where (x2, y2) is upper-light coner
90  IMG *PPMcvt(IMG *p1, IMG *ee, IMG *nthu, int x1, int y1, int x2, int y2)
91  {
92      int x, y, i, j;                             // index
93      PIXEL cyan = {0, 255, 255};                 // a cyan PIXEL
94
95      // convert to black-white image
```

```
96      for (x = 0; x < p1->W; x++)
97          for (y = 0; y < p1->H; y++)
98              if (x < x2 || y < y2 || x > x1 || y > y1) {
99                  p1->PX[x][y].b = p1->PX[x][y].g = p1->PX[x][y].r
100                     = p1->PX[x][y].r * 0.2126 + p1->PX[x][y].g * 0.7152
101                     + p1->PX[x][y].b * 0.0722;
102             }
103     // put a cyan box around my head
104     // vertical lines
105     for (x = x2; x < x1; x++) {
106         for (j = 0; j < 3; j++) {               // line width is 3px
107             p1->PX[x][y2 + j] = cyan;
108             p1->PX[x][y1 - j] = cyan;
109         }
110     }
111     // horizontal lines
112     for (y = y2; y < y1; y++) {
113         for (j = 0; j < 3; j++) {               // line width is 3px
114             p1->PX[x2 + j][y] = cyan;
115             p1->PX[x1 - j][y] = cyan;
116         }
117     }
118     // paste EE logo
119     for (x = 0, i = p1->W - ee->W; x < ee->W; x++, i++)
120         for (y = 0, j = p1->H -ee->H; y < ee->H; y++, j++)
        for (y = 0, j = p1->H - ee->H; y < ee->H; y++, j++)
121             // remove white PIXEL
122             if (ee->PX[x][y].r != 255 && ee->PX[x][y].g != 255
123                                       && ee->PX[x][y].b != 255)
124                 p1->PX[i][j] = ee->PX[x][y];
125     // paste NTHU logo
126     for (x = 0, i = (p1->W - nthu->W) / 2; x < nthu->W; x++, i++)
127         for (y = 0, j = (p1->H - nthu->H) / 2 ; y < nthu->H; y++, j++)
        for (y = 0, j = (p1->H - nthu->H) / 2; y < nthu->H; y++, j++)
128             // ignore white PIXEL
129             if (nthu->PX[x][y].r != 255 && nthu->PX[x][y].g != 255
130                                         && nthu->PX[x][y].b != 255) {
131                 p1->PX[i][j].g = nthu->PX[x][y].g;
132                 p1->PX[i][j].r = p1->PX[i][j].b = 255;
133                                                 // turn b, r, to max
134             }
135     return p1;
136 }
```

[Format] can be improved.
[Coding] lab14.c spelling errors: coner(4)
[NTHU] logo should be water-marked.
[Checking] condition for white pixel is not correct.
[Memory] leakage.

[PPMcvt] should return a 'new' image structure.

Score: 66