```
/*****************************************************************************
 * file - ip.h                                                               *
 *****************************************************************************/

/* typedefs */

typedef unsigned char *image_ptr; //char형 포인터
typedef double *double_ptr; //double형 포인터
typedef struct
    {//변수선언
    unsigned char r,g,b;
    } pixel; //pixel 구조체

typedef pixel *pixel_ptr; //구조체 포인터

typedef struct
    {//변수선언
    int width;
    int height;
    float *x_data;
    float *y_data;
    } mesh; //mesh 구조체

typedef struct
    {//변수선언
    double re;
    double im;
    } COMPLEX; //COMPLEX 구조체

typedef COMPLEX *complex_ptr; //구조체 포인터

typedef struct
    {//변수선언
    int x;
    int y;
    } POINT; //POINT 구조체

typedef struct
    {//변수선언
    POINT P;
    POINT Q;
    int dx, dy;
    float length;
```

```
    long length_squared;
    } LINE; //LINE 구조체

typedef struct
    {//변수선언
    POINT P;
    POINT Q;
    } LINE_SEGMENT; //LINE_SEGMENT 구조체

typedef struct
    {//변수선언
    int number;                /* number of segments to follow */
    LINE_SEGMENT line[100];
    char *filename; /* name of file holding the line list */
    } LINE_LIST; //LINE_LIST 구조체

/* defines */
//각종 전처리기 정의
#define PI    3.14159265358979323846
#define CLIP(val, low, high) {if(val<low) val=low; if(val>high) val=high;}
#define CLAMP(val, low, high) ((val<low) ? low : ((val>high) ? high : val))
#define MAX(A,B)          ((A) > (B) ? (A) : (B))
#define MIN(A,B)          ((A) < (B) ? (A) : (B))
#define IP_MALLOC(X) malloc(X)
#define IP_FREE(X) free(X)
#define PBM 4
#define PGM 5
#define PPM 6
```

```
/*****************************************************************************
 * File: iplib.c                                                             *
 *                                                                           *
 * Desc: general purpose image processing routines                           *
 *****************************************************************************/


#include <malloc.h>
#include <stdio.h>
#include <stdlib.h>
#include "ip.h"

image_ptr read_pnm(char *filename, int *rows, int *cols, int *type);
int getnum(FILE *fp);

/*****************************************************************************
 * Func: read_pnm                                                            *
 *                                                                           *
 * Desc: reads a portable bitmap file                                        *
 *                                                                           *
 * Params: filename - name of image file to read                             *
 *         rows - number of rows in the image                                *
 *         cols - number of columns in the image                             *
 *         type - file type                                                  *
 *                                                                           *
 * Returns: pointer to the image just read into memory                       *
 *****************************************************************************/

image_ptr read_pnm(char *filename, int *rows, int *cols, int *type)
    {
    int i;                      /* index variable */
    int row_size;               /* size of image row in bytes */
    int maxval;                 /* maximum value of pixel */
    FILE *fp;                   /* input file pointer */
    int firstchar, secchar;     /* first 2 characters in the input file */
    image_ptr ptr;              /* pointer to image buffer */
    unsigned long offset;       /* offset into image buffer */
    unsigned long total_size;   /* size of image in bytes */
    unsigned long total_bytes; /* number of total bytes written to file */
    float scale;                /* number of bytes per pixel */
```

```
/* open input file */
if((fp = fopen(filename, "rb")) == NULL) //읽기권한으로 파일 오픈
    {
    printf("Unable to open %s for reading\n",filename); //에러시 출력
    exit(1);
    }

firstchar = getc(fp); //첫번째 단어
secchar = getc(fp); //두번째 단어

if(firstchar != 'P') //첫단어가 'P'가 아니면
    {
    printf("You silly goof... This is not a PPM file!\n"); //출력
    exit(1);
    }

*cols = getnum(fp); //행
*rows = getnum(fp); //열
*type = secchar - '0'; //타입 추출

switch(secchar) //secchar switch 문
    {
    case '4':             /* PBM */
        scale = 0.125;
        maxval = 1;
        break;
    case '5':             /* PGM */
        scale = 1.0;
        maxval = getnum(fp);
        break;
    case '6':              /* PPM */
        scale = 3.0;
        maxval = getnum(fp);
        break;
    default :              /* Error */
        printf("read_pnm: This is not a Portable bitmap RAWBITS file\n");
        exit(1);
        break;
    }

row_size = (*cols) * scale; //행사이즈 계산
```

```
    total_size = (unsigned long) (*rows) * row_size; //총 사이즈 계산

    ptr = (image_ptr) IP_MALLOC(total_size); //포인터

    if(ptr == NULL) //포인터가 NULL이면
        {
        printf("Unable to malloc %lu bytes\n",total_size); //출력
        exit(1);
        }

    total_bytes=0;
    offset = 0;
    for(i=0; i<(*rows); i++) //행값만큼 반복
        {
        total_bytes+=fread(ptr+offset, 1, row_size, fp); //ptr+offset에 fp에있는 1byte씩
row_size만큼 read후 total_bytes 증가
        offset += row_size; //오프셋 증가
        }

    if(total_size != total_bytes) // 총 사이즈와 총 바이트가 다르면
        {
        printf("Failed miserably trying to read %ld bytes\nRead %ld bytes\n",
                total_size, total_bytes); //실패문구 출력
        exit(1);
        }

    fclose(fp);//파일 클로즈
    return ptr;
    }

/*************************************************************************
 * Func: getnum                                                          *
 *                                                                       *
 * Desc: reads an ASCII number from a portable bitmap file header        *
 *                                                                       *
 * Param: fp - pointer to file being read                                *
 *                                                                       *
 * Returns: the number read                                              *
 *************************************************************************/

int getnum(FILE *fp)
```

```
    {
    char c;                 /* character read in from file */
    int i;                  /* number accumulated and returned */

    do
        {
        c = getc(fp);
        }
    while((c==' ') || (c=='\t') || (c=='\n') || (c=='\r')); //해당하면 do

    if((c<'0') || (c>'9'))
        if(c == '#')                    /* chew off comments */
            {
            while(c == '#')
                {
                while(c != '\n') //개행 전까지
                    c = getc(fp);
                c = getc(fp);
                }
            }
        else
            {
            printf("Garbage in ASCII fields\n"); //출력문
            exit(1);
            }

    i=0;
    do
        {
        i=i*10+(c-'0');          /* convert ASCII to int */
        c = getc(fp);
        }
    while((c>='0') && (c<='9')); //0~9까지 do반복

    return i;
    }

/*************************************************************************
 * Func: write_pnm                                                       *
 *                                                                       *
 * Desc: writes out a portable bitmap file                               *
```

```
*                                                                     *
* Params: ptr - pointer to image in memory                            *
*         filename _ name of file to write image to                   *
*         rows - number of rows in the image                          *
*         cols - number of columns in the image                       *
*         magic_number - number that defines what type of file it is  *
*                                                                     *
* Returns: nothing                                                    *
**********************************************************************/

void write_pnm(image_ptr ptr, char *filename, int rows,
               int cols, int magic_number)
   {
   FILE *fp;              /* file pointer for output file */
   long offset;           /* current offset into image buffer */
   long total_bytes;      /* number of bytes written to output file */
   long total_size;       /* size of image buffer */
   int row_size;          /* size of row in bytes */
   int i;                 /* index variable */
   float scale;           /* number of bytes per image pixel */

   switch(magic_number) //magic_number의 switch문
       {
       case 4:            /* PBM */
           scale = 0.125;
           break;
       case 5:            /* PGM */
           scale = 1.0;
           break;
       case 6:             /* PPM */
           scale = 3.0;
           break;
       default :            /* Error */
           printf("write_pnm: This is not a Portable bitmap RAWBITS file\n");
           exit(1);
           break;
       }

   /* open new output file */
   if((fp=fopen(filename, "wb")) == NULL) //쓰기권한으로 파일 오픈
       {
```

```
        printf("Unable to open %s for output\n",filename); //에러시 출력
        exit(1);
        }

    /* print out the portable bitmap header */
    fprintf(fp, "P%d\n%d %d\n", magic_number, cols, rows); //파일에 magic_number
와 행열 print
    if(magic_number != 4)
        fprintf(fp, "255\n");// magic_number가 4 아닐시 print

    //사이즈 계산 및 초기화
    row_size = cols * scale;
    total_size = (long) row_size *rows;
    offset = 0;
    total_bytes = 0;

    for(i=0; i<rows; i++) //행만큼 반복
        {
        total_bytes += fwrite(ptr+offset, 1, row_size, fp); //fp에 ptr+offset있는 1byte씩
row_size만큼 write후 total_bytes 증가
        offset += row_size; //오프셋 증가
        }

    if(total_bytes != total_size) // 사이즈 다를시 에러문 출력
        printf("Tried to write %ld bytes...Only wrote %ld\n",
                total_size, total_bytes);

    fclose(fp); //파일 클로즈
    }

/*************************************************************************
 * Func: pnm_open                                                        *
 *                                                                       *
 * Desc: opens a pnm file and determines rows, cols, and maxval          *
 *                                                                       *
 * Params: rows- pointer to number of rows in the image                  *
 *         cols - pointer number of columns in the image                 *
 *         maxval - pointer to max value                                 *
 *         filename - name of image file                                 *

*************************************************************************/
```

```
FILE *pnm_open(int *rows, int *cols, int *maxval, char *filename)
    {
    //변수 선언
    int firstchar, secchar;
    float scale;
    unsigned long row_size;
    FILE *fp;

    if((fp = fopen(filename, "rb")) == NULL) //읽기권한으로 오픈
        {
        printf("Unable to open %s for reading\n",filename); //실패시 에러문 출력
        exit(1);
        }

    firstchar = getc(fp);
    secchar = getc(fp);

    if(firstchar != 'P')
        {
        printf("You silly goof... This is not a PPM file!\n"); //PPM아닐시 출력
        exit(1);
        }

    *cols = getnum(fp); //열
    *rows = getnum(fp); //행

    switch(secchar)
        {
        case '4':             /* PBM */
            scale = 0.125;
            *maxval = 1;
            break;
        case '5':             /* PGM */
            scale = 1.0;
            *maxval = getnum(fp);
            break;
        case '6':              /* PPM */
            scale = 3.0;
            *maxval = getnum(fp);
            break;
```

```
        default :               /* Error */
            printf("read_pnm: This is not a Portable bitmap RAWBITS file\n");
            exit(1);
            break;
        }

    row_size = (*cols) * scale; //행사이즈 계산
    return fp; //반환
    }


/*****************************************************************************
 * Func: read_mesh                                                           *
 *                                                                           *
 * Desc: reads mesh data into a mesh structure
*
 *                                                                           *
 * Params: filename - name of input mesh file                                *
 *                                                                           *
 * Returns: mesh structure storing width, height, x data  and y data         *

*****************************************************************************/
mesh *read_mesh(char *filename)
    {
    //변수 선언
    FILE *fp;
    mesh *mesh_data;
    int width, height, mesh_size;

    /* open mesh file for input */
    if((fp = fopen(filename, "rb")) == NULL)  //읽기권한 오픈
        {
        printf("Unable to open mesh file %s for reading\n", filename); //에러시 출력
        exit(1);
        }

    mesh_data = malloc(sizeof(mesh)); //동적할당
    /* read dimensions of mesh */
    fread(&width, sizeof(int), 1, fp); //width값 read
    fread(&height, sizeof(int), 1, fp); //height값 read
    mesh_data->width = width; //mesh_data에 저장
```

```
    mesh_data->height = height; //mesh_data에 저장
    mesh_size = width * height; //mesh_size 초기화

    /* allocate memory for mesh data */
    mesh_data->x_data = malloc(sizeof(float) * mesh_size); //x_data 메모리 할당
    mesh_data->y_data = malloc(sizeof(float) * mesh_size); //y_data 메모리 할당

    fread(mesh_data->x_data, sizeof(float), mesh_size, fp); //fp에서
mesh_data->x_data로 데이터 읽어옴
    fread(mesh_data->y_data, sizeof(float), mesh_size, fp); //fp에서
mesh_data->y_data로 데이터 읽어옴

    return(mesh_data); //반환
    }
```

```c
/*************************************************************************
 * File: arithlut.c                                                      *
 *                                                                       *
 * Desc: This program performs arithmetic point operations via LUTs      *
 *************************************************************************/


#include <stdio.h>
#include <string.h>
#include <malloc.h>
#include "ip.h"

#define operation(VALUE)          ((float) VALUE * 1.9)

extern void write_pnm(image_ptr ptr, char filein[], int rows,
                      int cols, int magic_number); //외부 함수 선언
extern image_ptr read_pnm(char *filename, int *rows, int *cols,
                      int *type); //포인터

int main(int argc, char *argv[])
    {
    char filein[100];                   /* name of input file */
    char fileout[100];                  /* name of output file */
    int rows, cols;                     /* image rows and columns */
    unsigned long i;                    /* counting index */
    unsigned long bytes_per_pixel;      /* number of bytes per image pixel */
    unsigned char LUT[256];             /* array for Look-up table */
    image_ptr buffer;                   /* pointer to image buffer */
    unsigned long number_of_pixels;     /* total number of pixels in image */
    int temp;                           /* temporary variable */
    int type;                           /* what type of image data */


    /* set input filename and output file name */
    if(argc == 3) //명령행 한번에 입력시
        {
        strcpy(filein, argv[1]);     //input name
        strcpy(fileout, argv[2]);    //output name
        }
    else//한개씩 입력시
        {
        printf("Input name of input file\n");
        gets(filein);//input name
```

```
        printf("\nInput name of output file\n");
        gets(fileout);//output name
        printf("\n");
        }

buffer = read_pnm(filein, &rows, &cols, &type); //버퍼에 내용 저장

/* initialize Look-up table */
for(i=0; i<256; i++) //LUT과정
        {
        temp = operation(i);
        CLIP(temp, 0, 255);
        LUT[i] = temp;
        }

/* determine bytes_per_pixel, 3 for color, 1 for gray-scale */
if(type == PPM) //type이 PPM이면
        bytes_per_pixel = 3;      //bytes_per_pixel을 3으로
else //아니면
        bytes_per_pixel = 1;      //bytes_per_pixel을 1으로

number_of_pixels = bytes_per_pixel * rows * cols; //number_of_pixels 설정

/* process image via the Look-up table */
for(i=0; i<number_of_pixels; i++) //number_of_pixels만큼 반복
        buffer[i] = LUT[buffer[i]]; //버퍼에 LUT적용시킨것으로 변경

write_pnm(buffer, fileout, rows, cols, type);     //출력
IP_FREE(buffer); //buffer free
return 0;
}
```


573 x 429 birdgirl.pgm
열기(O) 붙여넣기(P) 사이즈 조정(V) Interpolation(I) PNM Viewer 정보(A)...
준비


573 x 429 birdoutput.pgm
열기(O) 붙여넣기(P) 사이즈 조정(V) Interpolation(I) PNM Viewer 정보(A)...
준비


384 x 256 bullfight.ppm
열기(O) 붙여넣기(P) 사이즈 조정(V) Interpolation(I)
준비


384 x 256 output.ppm
열기(O) 붙여넣기(P) 사이즈 조정(V) Interpolation(I)
준비