

(1) 컬러 영상을 그레이 스케일 영상으로 변환하는 프로그램을 작성하라
입력영상: /image/puppy.ppm (10점)

```

#include <stdio.h>
int main(int argc, char * argv[]) //argc는 외부에서 입력한 문자열 개수,
{
    //argv[]는 외부에서 입력한 문자열들
    int image1[600][800*3]; //800x600 영상까지 처리가능 컬러영상
    int image2[600][800]; //800x600 영상까지 처리가능
    int x, y; //반복문에서 사용할 정수 변수 x, y 선언
    char M, N; //문자 변수 M과 N 선언
    int XX, YY, MAX; //정수 변수 XX, YY, MAX 선언
    FILE *file1, *file2; //파일 변수 file1과 file2 선언
    file1 = fopen(argv[1], "r"); //외부에서 입력한 두 번째 문자열을 읽기모드로 열
    fscanf(file1, "%c", &M); //헤더의 매직넘버를 읽어 들임 'P'
    fscanf(file1, "%c", &N); //헤더의 매직넘버를 읽어 들임 '3'
    fscanf(file1, "%d", &XX); //입력영상의 가로크기 읽어 들임
    fscanf(file1, "%d", &YY); //입력영상의 세로크기 읽어 들임
    fscanf(file1, "%d", &MAX); //입력영상의 최대 명암도 읽어 들임

    for(y = 0; y < YY; y++) //반복문을 사용하여 입력영상의 픽셀 값 읽어 들임
        for(x = 0; x < XX*3; x++) //컬러영상은 가로픽셀 당 값을 3개 가지므로 x3
            fscanf(file1, "%d", &image1[y][x]);

    for(y = 0; y < YY; y++) //반복문을 이용하여
        for(x = 0; x < XX; x++) //영상을 흑백화 시킴
            image2[y][x] = (image1[y][3*x]+image1[y][3*x+1]+image1[y][3*x+2])/3;
    //그레이 스케일 영상을 쓰자
    N='2'; // 컬러영상 N은 3, 그레이영상은 2 그레이영상 출력이므로 2
    file2 = fopen(argv[2], "w"); //외부에서 입력한 세 번째 문자열을 쓰기모드로 열
    fprintf(file2, "%c", M); //읽어 들인 두 번째 문자열 파일의 헤더 매직넘버 'P'
    fprintf(file2, "%cWn", N); //N에 2를 집어넣었기 때문에 2 입력
    fprintf(file2, "%d %dWn", XX, YY); //읽어 들인 가로축과 세로축 크기를 입력
    fprintf(file2, "%dWn", MAX); //읽어 들인 최대 명암도를 입력

    for(y = 0; y < YY; y++){
        for(x = 0; x < XX; x++){ //반복문을 이용하여 흑백화 시킨 픽셀 값을
            fprintf(file2, "%4d", image2[y][x]); //세 번째 문자열 파일에 입력
        }
        fprintf(file2, "Wn"); //줄 내림 입력
    }
    return 0;
}

```

What are you doing?

```

[kimhyungho@raspberrypi4 ~]gcc gray.c -o gray
[kimhyungho@raspberrypi4 ~]xv /image/puppy.ppm &
[1] 3822
[kimhyungho@raspberrypi4 ~]gray /image/puppy.ppm out.pgm
[kimhyungho@raspberrypi4 ~]xv out.pgm &
[2] 3824
[kimhyungho@raspberrypi4 ~]
    
```

fscanf를 읽기모드로 연 입력영상 /image/puppy.ppm의 헤더와 데이터 포맷 부분을 읽어 들였습니다. 읽어 들인 데이터 포맷을 그레이스케일 영상으로 바꾸려면 픽셀 당 3개의 값을 갖는 컬러영상의 픽셀 값을 하나로 바꿔주어야 합니다. 때문에 입력 컬러영상의 R, G, B 픽셀 더해서 하나의 픽셀 값으로 만든 다음에 3으로 나눠서 R, G, B의 평균 값을 구해주면 그레이스케일의 픽셀 값을 얻을 수 있습니다. 그리고 N='2'로 하여 출력영상의 헤더 매직넘버를 그레이 스케일 영상 PGM의 매직넘버로 바꿔줍니다. 때문에 사실상 위에서 fscanf를 입력영상의 매직넘버 '3'을 읽어 들인 것은 의미가 없습니다. 그리고 입력영상에서 읽어 들인 영상의 헤더 부분과 N=2로 바꿔준 매직넘버 그리고 그레이스케일 영상화 시킨 영상의 데이터 포맷을 쓰기모드로 연 out.pgm에 fprintf를 사용하여 입력했습니다. 이 후 실행과정에서 -o 옵션을 사용하여 컴파일 한 gray.c의 실행파일을 a.out 대신 gray로 만들어주었습니다. xv와 후면작업 &을 사용하여 입력영상 /image/puppy.ppm 과 gray 실행으로 얻어낸 출력영상 out.pgm을 동시에 출력하였습니다.

(2) 그레이 스케일 영상을 flip 영상으로 바꾸는 프로그램을 작성하라
 입력영상: /image/upsidedown.pgm (10점)

```

#include <stdio.h>
int main(int argc, char * argv[]) //argc는 외부에서 입력한 문자열 개수,
{
    //argv[]는 외부에서 입력한 문자열들
    int image1[600][800]; //800x600 영상까지 처리가능
    int image2[600][800]; //800x600 영상까지 처리가능
    int x, y; //반복문에서 사용할 정수 변수 x, y 선언
    char M, N; //문자 변수 M과 N 선언
    int XX, YY, MAX; //정수 변수 XX, YY, MAX 선언
    FILE *file1, *file2; //파일 변수 file1과 file2 선언
    file1 = fopen(argv[1], "r"); //외부에서 입력한 두 번째 문자열을 읽기모드로 열
    fscanf(file1, "%c", &M); //헤더의 매직넘버를 읽어 들임 'P'
    fscanf(file1, "%c", &N); //헤더의 매직넘버를 읽어 들임 '2'
    fscanf(file1, "%d", &XX); //입력영상의 가로크기 읽어 들임
    fscanf(file1, "%d", &YY); //입력영상의 세로크기 읽어 들임
    fscanf(file1, "%d", &MAX); //입력영상의 최대 명암도 읽어 들임

    for(y = 0; y < YY; y++) //반복문을 사용하여 입력영상의 픽셀 값 읽어 들임
        for(x = 0; x < XX; x++) //((영상의 세로 크기-1)-y좌표하면 상하 대칭
            fscanf(file1, "%d", &image1[y][x]);

    for(y = 0; y < YY; y++) //반복문을 사용하여
        for(x = 0; x < XX; x++) //영상을 x축 대칭시킴
            image2[(YY-1)-y][x] = image1[y][x];

    file2 = fopen(argv[2], "w"); //외부에서 입력한 세 번째 문자열을 쓰기모드로 열
    fprintf(file2, "%c", M); //읽어 들인 두 번째 문자열 파일의 헤더 매직넘버를
    fprintf(file2, "%cWn", N); //세 번째 문자열 파일에 입력
    fprintf(file2, "%d %dWn", XX, YY); //읽어 들인 가로축과 세로축 크기를 입력
    fprintf(file2, "%dWn", MAX); //읽어 들인 최대 명암도를 입력

    for(y = 0; y < YY; y++){
        for(x = 0; x < XX; x++){ //반복문을 이용하여 x축 대칭시킨 픽셀 값을
            fprintf(file2, "%4d", image2[y][x]); //세 번째 문자열 파일에 입력
        }
        fprintf(file2, "Wn"); //줄 내림 입력
    }

    return 0;
}

```

 What are you doing?

```

[kimhyungho@raspberrypi4 ~]gcc flip.c -o flip
[kimhyungho@raspberrypi4 ~]xv /image/puppy.pgm &
[1] 3910
[kimhyungho@raspberrypi4 ~]flip /image/puppy.pgm out.pgm
[kimhyungho@raspberrypi4 ~]xv out.pgm &
[2] 3912
[kimhyungho@raspberrypi4 ~]
    
```







fscnaf로 읽기모드로 연 입력영상 /image/puppy.pgm의 헤더와 데이터 포맷 부분을 읽어 들였습니다. 읽어 들인 데이터 포맷을 x축 대칭 시켰는데, x축 대칭을 시키려면 픽셀의 위치를 x축은 그대로 y축은 (영상의 세로축 크기-1) - y좌표를 해주시면 됩니다. -1을 해주는 이유는 픽셀의 좌표가 0부터 시작하기 때문입니다. 입력영상에서 읽어 들인 영상의 헤더 부분과 x축 대칭 시킨 영상의 데이터 포맷을 쓰기모드로 연 out.pgm에 fprintf를 사용하여 입력했습니다. 이 후 실행과정에서 -o 옵션을 사용하여 컴파일 한 flip.c의 실행파일을 flip으로 만들어주었습니다. 그 후 xv와 후면작업 &을 사용하여 입력영상 /image/puppy.pgm과 flip으로 얻어낸 출력영상 out.pgm을 동시에 출력하였습니다.

(3) 컬러 영상을 flip 영상으로 바꾸는 프로그램을 작성하라
 입력영상: /image/puppy.ppm (10점)

```

#include <stdio.h>
int main(int argc, char * argv[]) //argc는 외부에서 입력한 문자열 개수,
{
    //argv[]는 외부에서 입력한 문자열들
    int image1[600][800*3]; //800x600 영상까지 처리가능 컬러영상
    int image2[600][800*3]; //800x600 영상까지 처리가능 컬러영상
    int x, y; //반복문에서 사용할 정수 변수 x, y 선언
    char M, N; //문자 변수 M과 N 선언
    int XX, YY, MAX; //정수 변수 XX, YY, MAX 선언
    FILE *file1, *file2; //파일 변수 file1과 file2 선언
    file1 = fopen(argv[1], "r"); //외부에서 입력한 두 번째 문자열을 읽기모드로 열
    fscanf(file1, "%c", &M); //헤더의 매직넘버를 읽어 들임 'P'
    fscanf(file1, "%c", &N); //헤더의 매직넘버를 읽어 들임 '3'
    fscanf(file1, "%d", &XX); //입력영상의 가로크기 읽어 들임
    fscanf(file1, "%d", &YY); //입력영상의 세로크기 읽어 들임
    fscanf(file1, "%d", &MAX); //입력영상의 최대 명암도 읽어 들임

    for(y = 0; y < YY; y++) //반복문을 사용하여 입력영상의 픽셀 값 읽어 들임
        for(x = 0; x < XX*3; x++) //컬러영상은 가로픽셀 당 값을 3개 가지므로 x3
            fscanf(file1, "%d", &image1[y][x]);

    for(y = 0; y < YY; y++) //반복문을 사용하여 x축 대칭 시킴
        for(x = 0; x < XX*3; x++) //(영상의 세로 크기-1)-y좌표하면 상하 대칭
            image2[(YY-1)-y][x] = image1[y][x];

    file2 = fopen(argv[2], "w"); //외부에서 입력한 세 번째 문자열을 쓰기모드로 열
    fprintf(file2, "%c", M); //읽어 들인 두 번째 문자열 파일의 헤더 매직넘버를
    fprintf(file2, "%cWn", N); //세 번째 문자열 파일에 입력
    fprintf(file2, "%d %dWn", XX, YY); //읽어 들인 가로축과 세로축 크기를 입력
    fprintf(file2, "%dWn", MAX); //읽어 들인 최대 명암도를 입력



    for(y = 0; y < YY; y++){
        for(x = 0; x < XX*3; x++){ //반복문을 이용하여 x축 대칭시킨 픽셀 값을
            fprintf(file2, "%4d", image2[y][x]); //세 번째 문자열 파일에 입력
        }
        fprintf(file2, "Wn"); //줄 내림 입력
    }

    return 0;
}

```

What are you doing?

```
[kimhyungho@raspberrypi4 ~]gcc cflip.c -o cflip
[kimhyungho@raspberrypi4 ~]xv /image/puppy.ppm &
[1] 3960
[kimhyungho@raspberrypi4 ~]cflip /image/puppy.ppm out.ppm
[kimhyungho@raspberrypi4 ~]xv out.ppm &
[2] 3972
[kimhyungho@raspberrypi4 ~]
```

fscanf로 읽기모드로 연 입력영상 /image/puppy.ppm의 헤더와 데이터 포맷 부분을 읽어 들였습니다. 읽어 들인 데이터 포맷을 x축 대칭 시켰는데, x축 대칭을 시키려면 픽셀의 위치를 x축은 그대로 y축은 (영상의 세로축 크기-1) - y좌표를 해주시면 됩니다. -1을 해주는 이유는 픽셀의 좌표가 0부터 시작하기 때문입니다. *3을 해준 이유는 컬러영상은 픽셀당 값을 3개 가지기 때문입니다. 입력영상에서 읽어 들인 영상의 헤더 부분과 x축 대칭 시킨 영상의 데이터 포맷을 쓰기모드로 연 out.ppm에 fprintf를 사용하여 입력했습니다. 이 후 실행과정에서 -o 옵션을 사용하여 컴파일 한 cflip.c의 실행파일을 cflip으로 만들어주었습니다. 그 후 xv와 후면작업 &을 사용하여 입력영상 /image/puppy.ppm과 flip으로 얻어낸 출력영상 out.ppm을 동시에 출력하였습니다.

(4) 그레이 스케일 영상을 mirror 영상으로 바꾸는 프로그램을 작성하라
 입력영상: /image/puppy.pgm (10점)

```

#include <stdio.h>
int main(int argc, char * argv[]) //argc는 외부에서 입력한 문자열 개수,
{
    //argv[]는 외부에서 입력한 문자열들
    int image1[600][800]; //800x600 영상까지 처리가능
    int image2[600][800]; //800x600 영상까지 처리가능
    int x, y; //반복문에서 사용할 정수 변수 x, y 선언
    char M, N; //문자 변수 M과 N 선언
    int XX, YY, MAX; //정수 변수 XX, YY, MAX 선언
    FILE *file1, *file2; //파일 변수 file1과 file2 선언
    file1 = fopen(argv[1], "r"); //외부에서 입력한 두 번째 문자열을 읽기모드로 열
    fscanf(file1, "%c", &M); //헤더의 매직넘버를 읽어 들임 'P'
    fscanf(file1, "%c", &N); //헤더의 매직넘버를 읽어 들임 '2'
    fscanf(file1, "%d", &XX); //입력영상의 가로크기 읽어 들임
    fscanf(file1, "%d", &YY); //입력영상의 세로크기 읽어 들임
    fscanf(file1, "%d", &MAX); //입력영상의 최대 명암도 읽어 들임

    for(y = 0; y < YY; y++) //반복문을 사용하여 입력영상의 픽셀 값 읽어 들임
        for(x = 0; x < XX; x++)
            fscanf(file1, "%d", &image1[y][x]);


    for(y = 0; y < YY; y++) //반복문을 사용하여 y축 대칭 시킴
        for(x = 0; x < XX; x++) //((영상의 가로 크기-1)-x좌표하면 좌우 대칭
            image2[y][(XX-1)-x] = image1[y][x];

    file2 = fopen(argv[2], "w"); //외부에서 입력한 세 번째 문자열을 쓰기모드로 열
    fprintf(file2, "%c", M); //읽어 들인 두 번째 문자열 파일의 헤더 매직넘버를
    fprintf(file2, "%cWn", N); //세 번째 문자열 파일에 입력
    fprintf(file2, "%d %dWn", XX, YY); //읽어 들인 가로축과 세로축 크기를 입력
    fprintf(file2, "%dWn", MAX); //읽어 들인 최대 명암도를 입력

    for(y = 0; y < YY; y++){
        for(x = 0; x < XX; x++){ //반복문을 이용하여 y축 대칭시킨 픽셀 값을
            fprintf(file2, "%4d", image2[y][x]); //세 번째 문자열 파일에 입력
        }
        fprintf(file2, "Wn"); //줄 내림 입력
    }


    return 0;
}

```



 What are you doing?

```

[kimhyungho@raspberrypi4 ~]gcc mirror.c -o mirror
[kimhyungho@raspberrypi4 ~]xv /image/puppy.pgm &
[1] 4001
[kimhyungho@raspberrypi4 ~]mirror /image/puppy.pgm out.pgm
[kimhyungho@raspberrypi4 ~]xv out.pgm &
[2] 4005
[kimhyungho@raspberrypi4 ~]
    
```







fscnaf로 읽기모드로 연 입력영상 /image/puppy.pgm의 헤더와 데이터 포맷 부분을 읽어 들였습니다. 읽어 들인 데이터 포맷을 y축 대칭 시켰는데, y축 대칭을 시키려면 픽셀의 위치를 y축은 그대로 x축은 (영상의 가로축 크기-1) - x좌표를 해주시면 됩니다. -1을 해주는 이유는 픽셀의 좌표가 0부터 시작하기 때문입니다. 입력영상에서 읽어 들인 영상의 헤더 부분과 y축 대칭 시킨 영상의 데이터 포맷을 쓰기모드로 연 out.pgm에 fprintf를 사용하여 입력했습니다. 이 후 실행과정에서 -o 옵션을 사용하여 컴파일 한 mirror.c의 실행파일을 mirror으로 만들어주었습니다. 그 후 xv와 후면작업 &을 사용하여 입력영상 /image/puppy.pgm과 mirror로 얻어낸 출력영상 out.pgm을 동시에 출력하였습니다.

(5) 컬러 영상을 mirror 영상으로 바꾸는 프로그램을 작성하라
 입력영상: /image/puppy/ppm
 강아지는 호랑이도 없는데 왜 파랗게 질렸을까요?(분석:20점)
 이를 수정하여 다음과 같이 출력되게 하세요(20점)

```

#include <stdio.h>
int main(int argc, char * argv[]) //argc는 외부에서 입력한 문자열 개수,
{
    //argv[]는 외부에서 입력한 문자열들
    int image1[600][800*3]; //800x600 영상까지 처리가능 컬러영상
    int image2[600][800*3]; //800x600 영상까지 처리가능 컬러영상
    int x, y; //반복문에서 사용할 정수 변수 x, y 선언
    char M, N; //문자 변수 M과 N 선언
    int XX, YY, MAX; //정수 변수 XX, YY, MAX 선언
    FILE *file1, *file2; //파일 변수 file1과 file2 선언
    file1 = fopen(argv[1], "r"); //외부에서 입력한 두 번째 문자열을 읽기모드로 열
    fscanf(file1, "%c", &M); //헤더의 매직넘버를 읽어 들임 'P'
    fscanf(file1, "%c", &N); //헤더의 매직넘버를 읽어 들임 '3'
    fscanf(file1, "%d", &XX); //입력영상의 가로크기 읽어 들임
    fscanf(file1, "%d", &YY); //입력영상의 세로크기 읽어 들임
    fscanf(file1, "%d", &MAX); //입력영상의 최대 명암도 읽어 들임

    for(y = 0; y < YY; y++) //반복문을 사용하여 입력영상의 픽셀 값 읽어 들임
        for(x = 0; x < XX*3; x++) //컬러영상은 가로픽셀 당 값을 3개 가지므로 x3
            fscanf(file1, "%d", &image1[y][x]);

    for(y = 0; y < YY; y++) //반복문을 이용하여 y축 대칭 시킴
        for(x = 0; x < XX*3; x++) //컬러영상이기 때문에 x3
            image2[y][(XX*3-1)-x] = image1[y][x]; //y축 대칭

    file2 = fopen(argv[2], "w"); //외부에서 입력한 세 번째 문자열을 쓰기모드로 열
    fprintf(file2, "%c", M); //읽어 들인 두 번째 문자열 파일의 헤더 매직넘버를
    fprintf(file2, "%cWn", N); //세 번째 문자열 파일에 입력
    fprintf(file2, "%d %dWn", XX, YY); //읽어 들인 가로축과 세로축 크기를 입력
    fprintf(file2, "%dWn", MAX); //읽어 들인 최대 명암도를 입력

    for(y = 0; y < YY; y++){
        for(x = 0; x < XX*3; x++){ //반복문을 이용하여 y축 대칭 시킨 픽셀 값을
            fprintf(file2, "%4d", image2[y][x]); //세 번째 문자열 파일에 입력
        }
        fprintf(file2, "Wn"); //줄 내림 입력
    }

    return 0;
}

```

What are you doing?

```

[kimhyungho@raspberrypi4 ~]gcc cmirror.c -o cmirror
[kimhyungho@raspberrypi4 ~]xv /image/puppy.ppm &
[1] 4130
[kimhyungho@raspberrypi4 ~]cmirror /image/puppy.ppm out.ppm
[kimhyungho@raspberrypi4 ~]xv out.ppm &
[2] 4142
[kimhyungho@raspberrypi4 ~]
    
```

fscnaf로 읽기모드로 연 입력영상 /image/puppy.ppm의 헤더와 데이터 포맷 부분을 읽어 들였습니다. 읽어 들인 데이터 포맷을 y축 대칭 시켰는데, y축 대칭을 시키려면 픽셀의 위치를 y축은 그대로 x축은 (영상의 가로축 크기-1) -x좌표를 해주시면 됩니다. -1을 해주는 이유는 픽셀의 좌표가 0부터 시작하기 때문입니다. 하지만 이렇게 할 경우 R, G, B 순으로 있는 컬러영상의 픽셀 값이 좌우 대칭된 B, G, R 순이 되어 버립니다. 예를 들어 가로축 크기가 4인 컬러영상의 가로축 값의 위치가 0 1 2 3 4 5 6 7 8 9 10 11 일 경우 (가로축 크기-1)-x 로 좌우 대칭 시킬 경우 11 10 9 8 7 6 5 4 3 2 1이 됩니다. 때문에 영상의 컬러가 달라 질 수 있습니다. 원래 색을 보존한 채 좌우 대칭 시키려면 9 10 11 6 7 8 3 4 5 0 1 2 순이 되어야합니다. 입력영상에서 읽어 들인 영상의 헤더 부분과 y축 대칭 시킨 영상의 데이터 포맷을 쓰기모드로 연 out.ppm에 fprintf를 사용하여 입력했습니다. 이 후 실행과정에서 -o 옵션을 사용하여 컴파일 한 cmirror.c의 실행파일을 cmirror로 만들어주었습니다. 그 후 xv와 후면작업 &을 사용하여 입력영상 /image/puppy.ppm과 cmirror로 얻어낸 출력영상 out.ppm을 동시에 출력하였습니다.

```

#include <stdio.h>
int main(int argc, char * argv[]) //argc는 외부에서 입력한 문자열 개수,
{
    //argv[]는 외부에서 입력한 문자열들
    int image1[600][800*3]; //800x600 영상까지 처리가능 컬러영상
    int image2[600][800*3]; //800x600 영상까지 처리가능 컬러영상
    int x, y; //반복문에서 사용할 정수 변수 x, y 선언
    char M, N; //문자 변수 M과 N 선언
    int XX, YY, MAX; //정수 변수 XX, YY, MAX 선언
    FILE *file1, *file2; //파일 변수 file1과 file2 선언
    file1 = fopen(argv[1], "r"); //외부에서 입력한 두 번째 문자열을 읽기모드로 열
    fscanf(file1, "%c", &M); //헤더의 매직넘버를 읽어 들임 'P'
    fscanf(file1, "%c", &N); //헤더의 매직넘버를 읽어 들임 '3'
    fscanf(file1, "%d", &XX); //입력영상의 가로크기 읽어 들임
    fscanf(file1, "%d", &YY); //입력영상의 세로크기 읽어 들임
    fscanf(file1, "%d", &MAX); //입력영상의 최대 명암도 읽어 들임

    for(y = 0; y < YY; y++) //반복문을 사용하여 입력영상의 픽셀 값 읽어 들임
        for(x = 0; x < XX*3; x++) //컬러영상은 가로픽셀 당 값을 3개 가지므로 x3
            fscanf(file1, "%d", &image1[y][x]);

    for(y = 0; y < YY; y++) //반복문을 이용하여 y축 대칭 시킴
        for(x = 0; x < XX*3; x+=3){ //R, G, B 값 다른 식을 이용할 것이므로 x+=3
            image2[y][-x+(XX-1)*3] = image1[y][x]; //(가로크기*3-1)-x좌표를 할 경우
            image2[y][-x+1)+(XX-1)*3+2] = image1[y][x+1]; //R과 B의 위치가 바뀜
            image2[y][-x+2)+(XX-1)*3+4] = image1[y][x+2]; //그것을 맞춰줌
        }

    file2 = fopen(argv[2], "w"); //외부에서 입력한 세 번째 문자열을 쓰기모드로 열
    fprintf(file2, "%c", M); //읽어 들인 두 번째 문자열 파일의 헤더 매직넘버를
    fprintf(file2, "%cWn", N); //세 번째 문자열 파일에 입력
    fprintf(file2, "%d %dWn", XX, YY); //읽어 들인 가로축과 세로축 크기를 입력
    fprintf(file2, "%dWn", MAX); //읽어 들인 최대 명암도를 입력

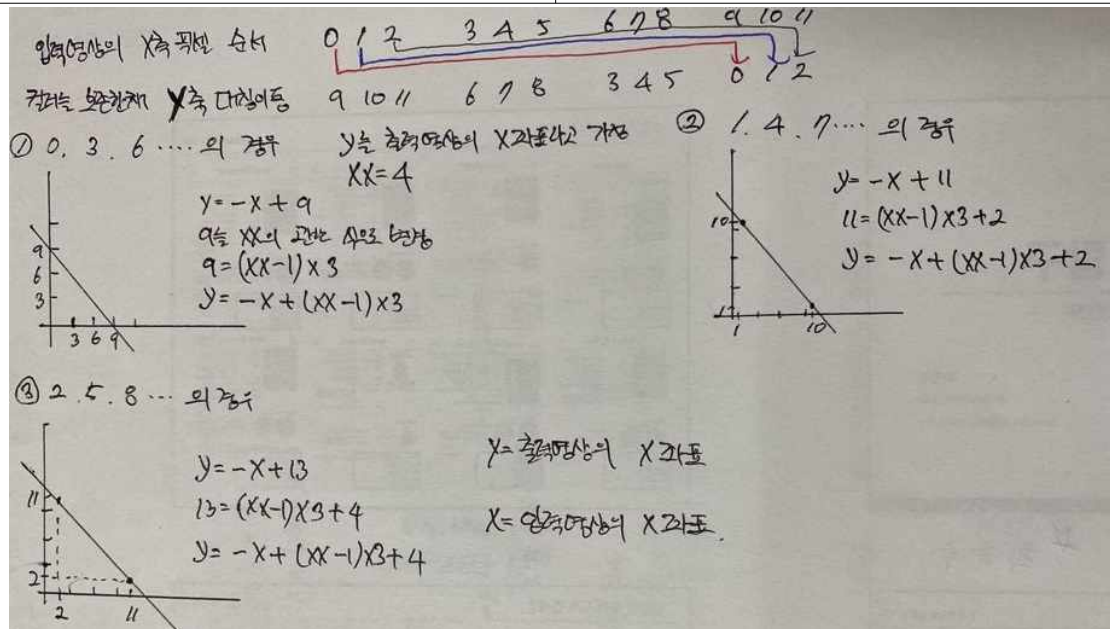
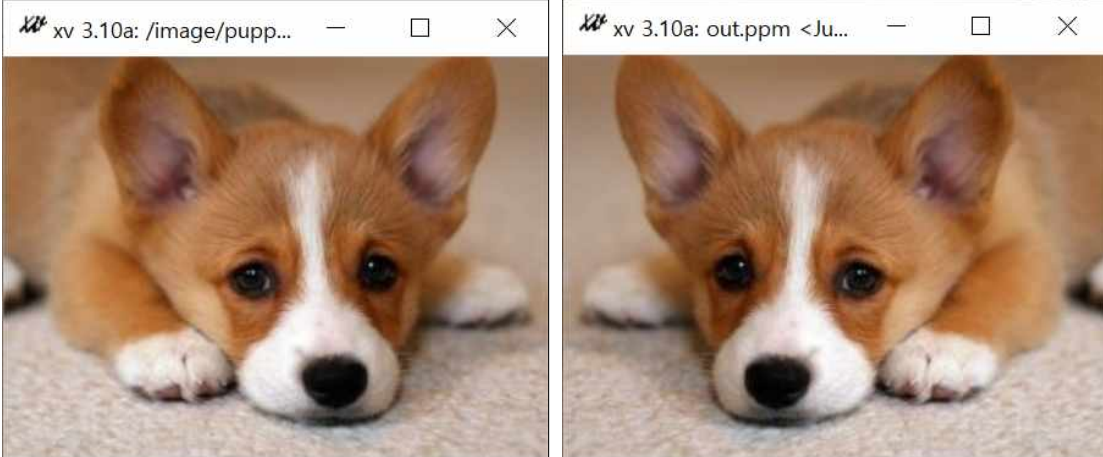
    for(y = 0; y < YY; y++){
        for(x = 0; x < XX*3; x++){ //반복문을 이용하여 y축 대칭 시킨 픽셀 값을
            fprintf(file2, "%4d", image2[y][x]); //세 번째 문자열 파일에 입력
        }
        fprintf(file2, "Wn"); //줄 내림 입력
    }

    return 0;
}

```

What are you doing?

```
[kimhyungho@raspberrypi4 ~] gcc rmirror.c -o rmirror
[kimhyungho@raspberrypi4 ~] xv /image/puppy.ppm &
[1] 4226
[kimhyungho@raspberrypi4 ~] rmirror /image/puppy.ppm out.ppm
[kimhyungho@raspberrypi4 ~] xv out.ppm &
[2] 4228
[kimhyungho@raspberrypi4 ~]
```



위에 좌우대칭 되었지만 색이 이상해진 영상을 색을 보존한 채 좌우대칭 시키려면 B, G, R 순으로 정렬된 픽셀 값을 R, G, B 순으로 바꿔주어야 합니다. 예를 들어 가로축 크기가 4인 컬러영상의 가로축 값의 위치가 0 1 2 3 4 5 6 7 8 9 10 11 일 경우 (가로축 크기-1)-x 로 좌우 대칭 시킬 경우 11 10 9 8 7 6 5 4 3 2 1이 됩니다. 이것을 9 10 11 6 7 8 3 4 5 0 1 2 로 바꿔주어야 본래 색 그대로 좌우 대칭된 영상을 얻을 수 있습니다. 때문에 방정식을 이용하여 R값을 옮기는 방법, G값을 옮기는 방법, B값을 옮기는 방법을 구했고 위에 그 방법을 손으로 풀어 첨부했습니다. 각각 구해본 결과 입력영상의 가로 픽셀 위치를 x라고 하고 출력 영상의 가로 픽셀 위치를 y라고 했을 때 R의 경우는 $y = -x + (\text{가로크기} - 1) * 3$ 이 되고 G의 경우는 $y = -x + (\text{가로크기} - 1) * 3 + 2$, B의 경우는 $y = -x + (\text{가로크기} - 1) * 3 + 4$ 가 됩니다. 따라서 식을 정리하여 C언어로 정리하면

```
image2[y][-x+(XX-1)*3] = image1[y][x]
image2[y][-x+1+(XX-1)*3+2] = image1[y][x+1]
image2[y][-x+2+(XX-1)*3+4] = image1[y][x+2]
```

가 됩니다. 이대로 컴파일해서 출력할 경우 온전하게 y축 대칭 된 컬러 영상을 얻을 수 있습니다.