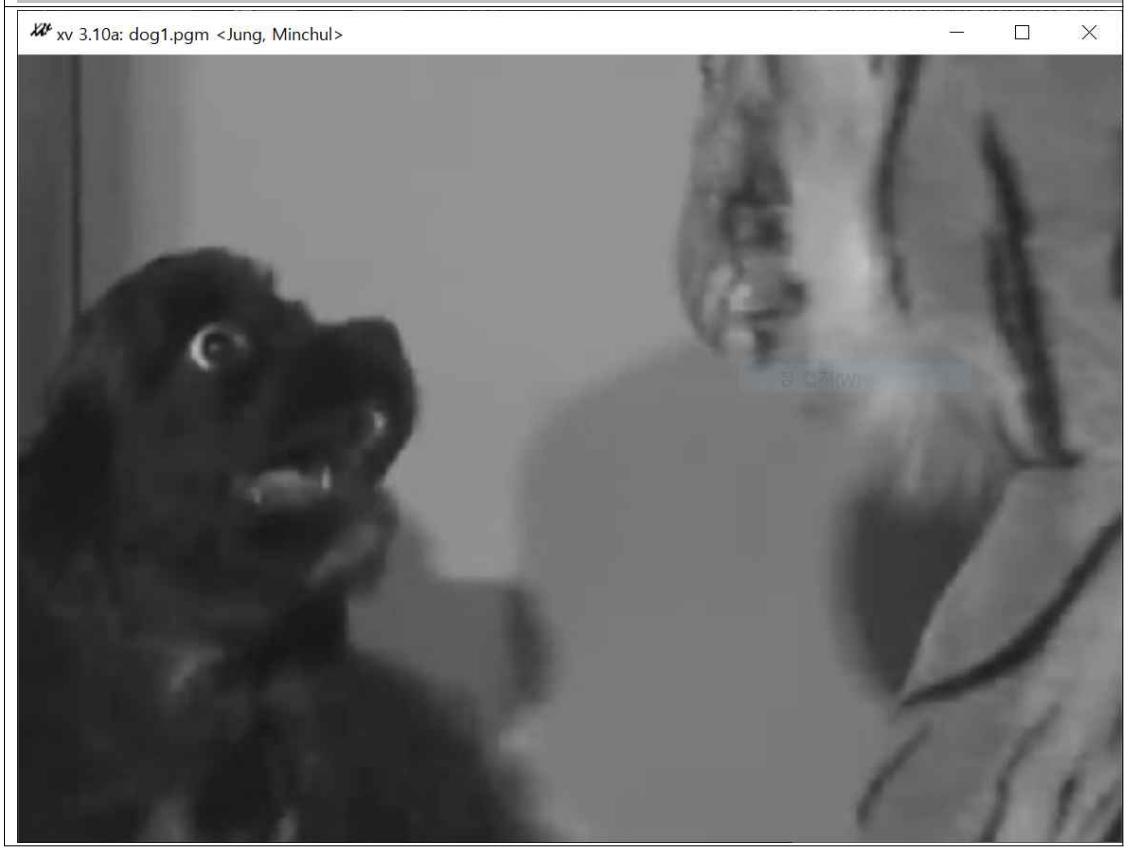
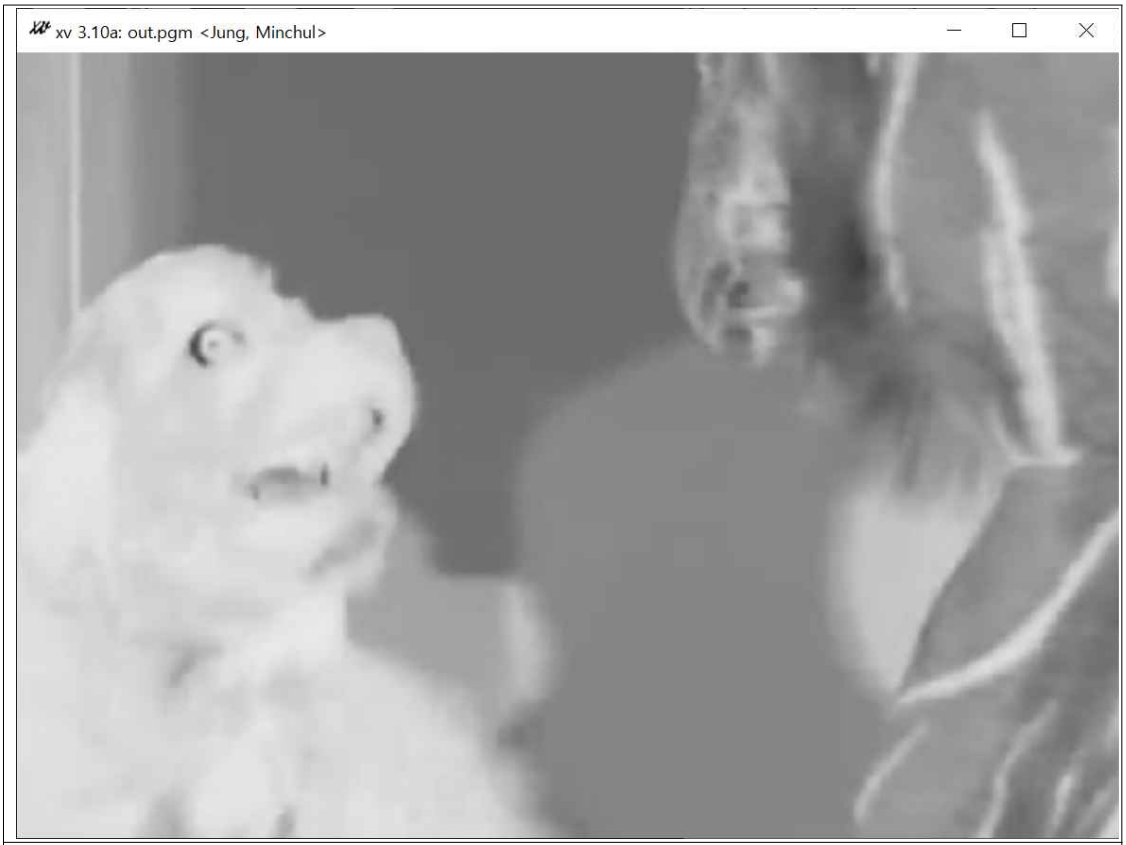
(1) 그레이 스케일 영상을 보수화 처리하는 프로그램을 작성하라
/image/dog1.pgm과 /image/dog2.pgm을 보수화 처리하라.

```
#include <stdio.h>
int main(int argc, char * argv[])  //argc는 외부에서 입력한 문자열 개수,
{                                  //argv[]는 외부에서 입력한 문자열들
 int image[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 염
 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", &image[y][x]);

 for(y = 0; y < YY; y++)           //반복문을 사용하여 영상을 보수화 시킴
  for(x = 0; x < XX; x++)
   image[y][x] = 255 - image[y][x];

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

 for(y = 0; y < YY; y++){
  for(x = 0; x < XX; x++){         //반복문을 이용하여 보수화 시킨 픽셀 값을
   fprintf(file2,"%4d", image[y][x]);  //세 번째 문자열 파일에 입력
  }
  fprintf(file2, "₩n");
 }
 return 0;
}
```

Computer Engineering - Prof. Jung, Minchul

```
[kimhyungho@raspberry4 ~]nano complement.c
[kimhyungho@raspberry4 ~]gcc complement.c
[kimhyungho@raspberry4 ~]cp /image/dog1.pgm .
[kimhyungho@raspberry4 ~]xv dog1.pgm &
[1] 4504
[kimhyungho@raspberry4 ~]a.out dog1.pgm out.pgm
[kimhyungho@raspberry4 ~]xv out.pgm &
[2] 4507
[kimhyungho@raspberry4 ~]
```

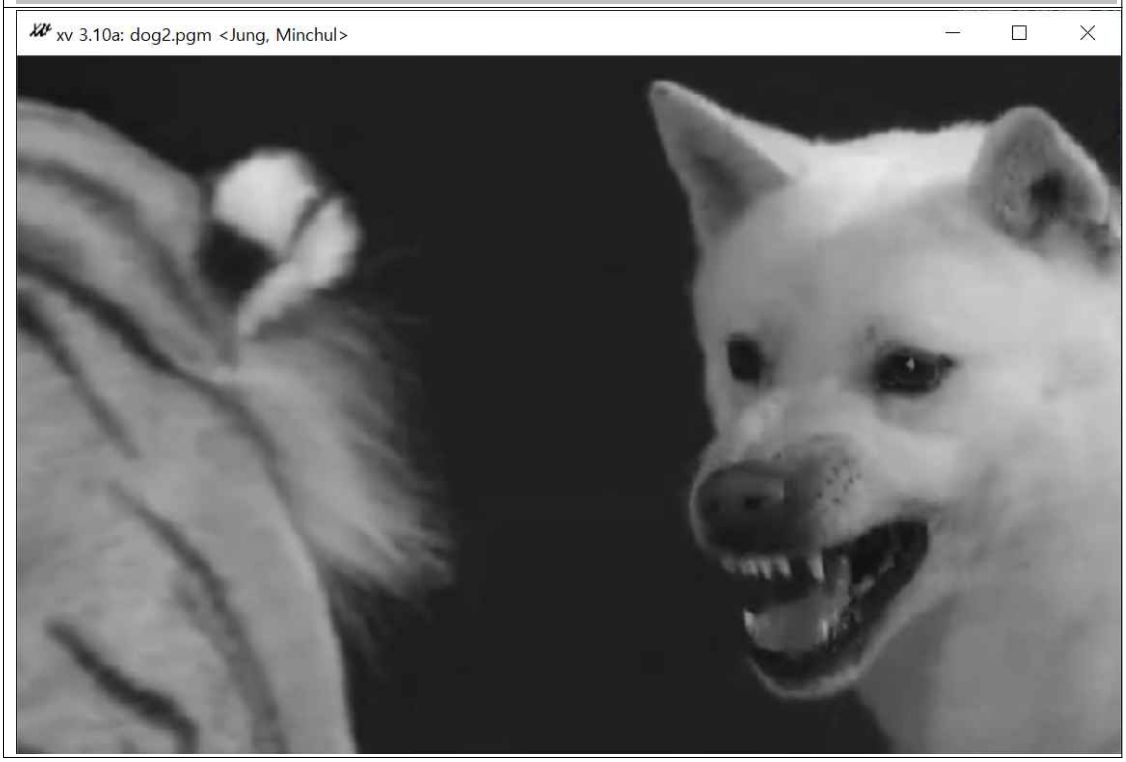




이전 과제까지는 입력 이미지와 출력이미지의 이름을 받아서 사용했습니다. 하지만 포인터 배열을 사용함으로써 사용자가 원하는 이름을 가진 입력이미지와 출력이미지를 자유롭게 키보드로 입력해서 컴파일 할 수 있습니다. argc는 외부에서 입력받는 문자열의 개수이고 argv[0], argv[1], argv[2]는 순서대로 외부에서 입력한 문자열입니다. 여기서는 argv[0]은 a.out, argv[1]은 dog1.pmg, argv[2]는 out.pgm을 외부에서 입력받아서 argv[1]인 dog1.pgm이 입력영상이되고 argv[2]인 out.pgm이 출력영상이 되었습니다. argc는 외부에서 입력받은 문자열의 개수이기 때문에 3이 됩니다. fscanf로 읽기모드로 연 dog1.pgm의 헤더와 데이터 포맷 부분을 읽어 들였고 읽어 들인 데이터 포맷을 반복문과 image[y][x] = 255 - image[y][x]를 사용하여 보수화 시켰습니다. fprintf로 쓰기모드로 연 out.pgm에 읽어 들인 영상 헤더와 보수화 된 데이터 포맷을 입력하였습니다. 그 후 실행과정에서 gcc를 사용하여 complement.c를 컴파일하였고, cp /image/dog1.pgm . 으로 현재 디렉터리에 dog1.pgm을 복사하였습니다. (.은 같은 이름으로 복사한다는 말이므로 반드시 적어주어야 합니다. 만약 다른 이름으로 저장하고 싶다면 .대신에 저장하고 싶은 파일명을 적어주면 됩니다.) 그 다음a.out으로 실행시켰으며 argv[1]에 dog1.pgm을 argv[2]에 out.pgm을 입력하였고 xv와 후면 작업 &을 이용하여 화면에 동시에 입력영상 dog1.pgm과 출력영상 out.pgm을 띄었습니다.

Computer Engineering - Prof. Jung, Minchul

```
[kimhyungho@raspberry4 ~]cp /image/dog2.pgm .
[kimhyungho@raspberry4 ~]a.out dog2.pgm out.pgm
[kimhyungho@raspberry4 ~]xv dog2.pgm &
[1] 4520
[kimhyungho@raspberry4 ~]xv out.pgm &
[2] 4521
[kimhyungho@raspberry4 ~]
```



dog1.pgm을 보수화 처리 한 것과 같은 소스를 이용하였습니다. 여기서 argc는 3, argv[0]는 a.out, argv[1]은 입력영상으로 dog2.pgm, argv[2]는 출력영상으로 out.pgm이 됩니다. 실행과정에서는 dog1.pgm을 작업할 때 이미 앞에서 complement.c를 컴파일 했으므로 gcc로 따로 컴파일 하지 않았습니다. cp /image/dog2.pgm .을 이용해서 image 디렉터리에 있는 dog2.pgm을 같은 이름으로 현재 디렉터리에 저장했습니다. 그 다음 a.out으로 실행시켰고 argv[1]에는 dog2.pgm, argv[2]에는 out.pgm을 입력하였고 xv와 후면작업 &을 이용해 화면에 동시에 dog2.pgm과 out.pgm을 띄었습니다.

(2) 컬러 영상을 보수화 처리하는 프로그램을 작성하라
/image/conan.ppm을 보수화 처리하라.

```
#include <stdio.h>
int main(int argc, char * argv[])  //argc는 외부에서 입력한 문자열 개수,
{                                  //argv[]는 외부에서 입력한 문자열들
 int image[600][800*3];            //800*3x600 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 염
 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 픽셀 당 값 3개 가지므로 *3
     fscanf(file1, "%d", &image[y][x]);

 for(y = 0; y < YY; y++)           //반복문을 사용하여 영상을 보수화 시킴
  for(x = 0; x < XX*3; x++)        //컬러영상은 픽셀 당 값 3개 가지므로 *3
   image[y][x] = 255 - image[y][x];

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

 for(y = 0; y < YY; y++){
  for(x = 0; x < XX*3; x++){       //컬러영상은 픽셀 당 값 3개 가지므로 *3
   fprintf(file2,"%4d", image[y][x]);  //반복문을 이용하여 보수화 시킨 픽셀 값을
  }                                //세 번째 문자열 파일에 입력
  fprintf(file2, "₩n");
 }
 return 0;
}
```

Computer Engineering - Prof. Jung, Minchul

```
[kimhyungho@raspberry4 ~]nano complement.c
[kimhyungho@raspberry4 ~]gcc complement.c
[kimhyungho@raspberry4 ~]cp /image/conan.ppm .
[kimhyungho@raspberry4 ~]a.out conan.ppm out.ppm
[kimhyungho@raspberry4 ~]xv conan.ppm &
[1] 4586
[kimhyungho@raspberry4 ~]xv out.ppm &
[2] 4587
[kimhyungho@raspberry4 ~]
```

여기서 argc는 3, argv[0]는 a.out, argv[1]은 conan.ppm, argv[2]는 out.pgm이 됩니다. 앞에 pgm 영상들과는 다르게 ppm 확장자를 갖는 컬러 영상이기 때문에 읽어 들인 영상의 종류를 나타내는 헤더의 매직 넘버는 P3가 됩니다. 컬러영상의 가로크기는 한 픽셀에 R, G, B 세 개의 값을 가지므로 반복문을 써서 픽셀 값을 입력영상에서 읽어 들일 때와 출력영상에 입력할 때 가로크기에 곱하기 3을 해주었습니다. 또 800x600으로 영상을 처리하지 못해서 image[600][800*3] 부분에 *3을 해주었습니다. image[600][800]으로 영상을 작업해 봤더니 중간 중간 영상이 잘린 이상한 이미지가 출력되었습니다. fscanf로 읽기모드로 연 conan.ppm의 헤더와 데이터 포맷 부분을 읽어 들였고 읽어 들인 데이터 포맷을 반복문과 image[y][x] = 255 - image[y][x]를 사용하여 보수화 시켰습니다. fprintf로 쓰기모드로 연 out.pgm에 읽어 들인 영상 헤더와 보수화 된 데이터 포맷을 입력하였습니다. 그 후 실행과정에서 gcc를 사용하여 complement.c를 컴파일하였고, cp /image/conan.ppm .을 이용해서 image 디렉터리에 있는 conan.ppm을 같은 이름으로 현재 디렉터리에 저장했습니다. 그 다음a.out으로 실행시켰으며 argv[1]에 conan.ppm을 argv[2]에 out.pgm을 입력하였고 xv와 후면 작업 &을 이용하여 화면에 동시에 입력영상 conan.ppm과 출력영상 out.pgm을 띄었습니다.