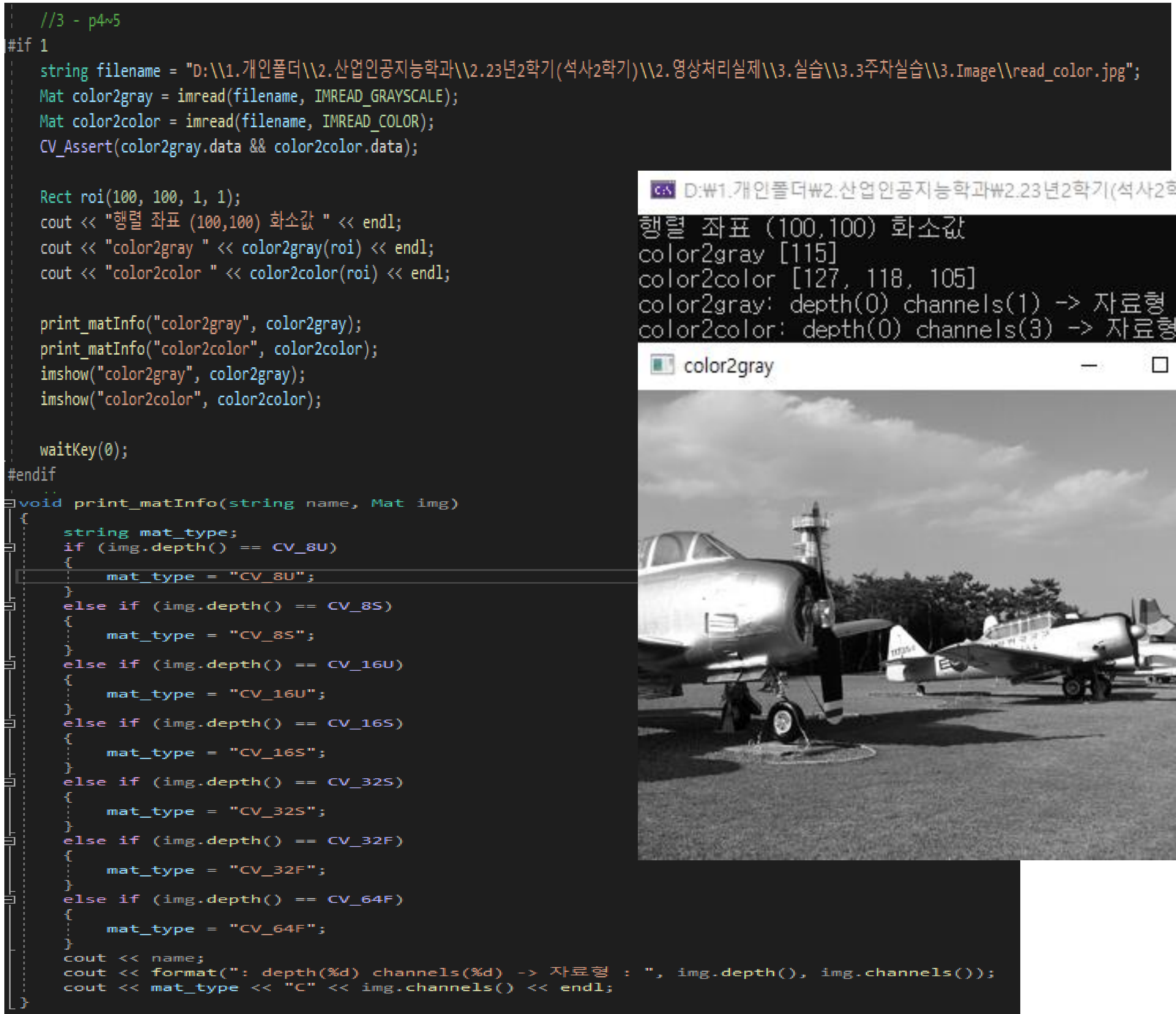
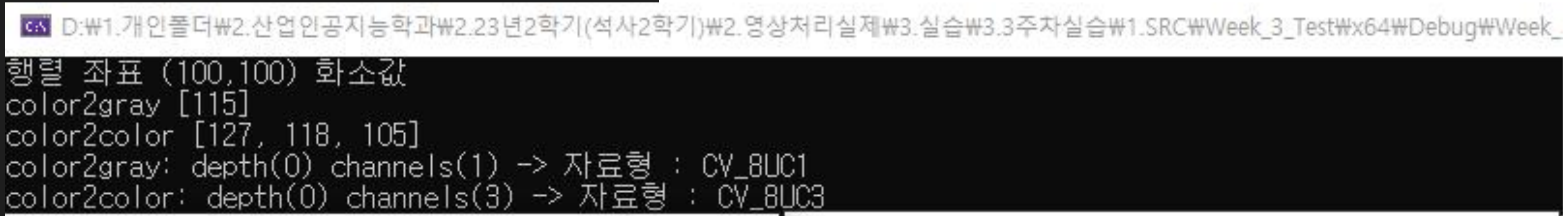
영상처리 실제 - 3주차 실습

2023254009 최현동

(3) - p.4~5

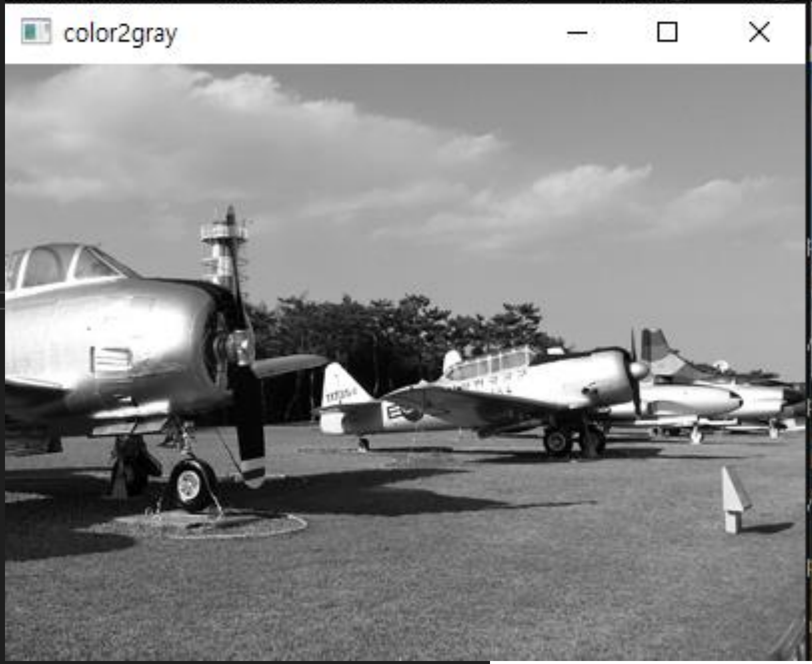


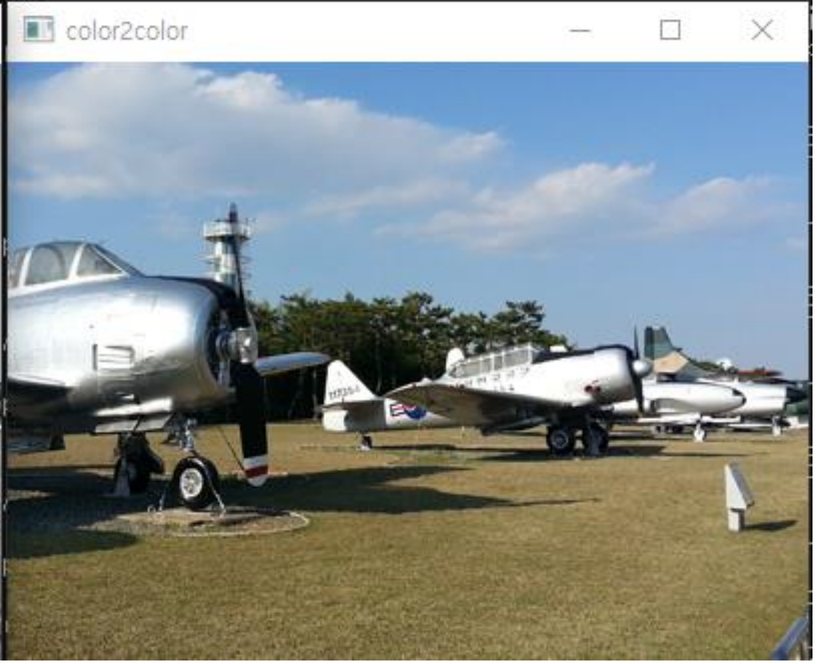
```
    //3 - p4~5
#if 1
    string filename = "D:\\1.개인폴더\\2.산업인공지능학과\\2.23년2학기(석사2학기)\\2.영상처리실제\\3.실습\\3.3주차실습\\3.Image\\read_color.jpg";
    Mat color2gray = imread(filename, IMREAD_GRAYSCALE);
    Mat color2color = imread(filename, IMREAD_COLOR);
    CV_Assert(color2gray.data && color2color.data);

    Rect roi(100, 100, 1, 1);
    cout << "행렬 좌표 (100,100) 화소값 " << endl;
    cout << "color2gray " << color2gray(roi) << endl;
    cout << "color2color " << color2color(roi) << endl;

    print_matInfo("color2gray", color2gray);
    print_matInfo("color2color", color2color);
    imshow("color2gray", color2gray);
    imshow("color2color", color2color);

    waitKey(0);
#endif
void print_matInfo(string name, Mat img)
{
    string mat_type;
    if (img.depth() == CV_8U)
    {
        mat_type = "CV_8U";
    }
    else if (img.depth() == CV_8S)
    {
        mat_type = "CV_8S";
    }
    else if (img.depth() == CV_16U)
    {
        mat_type = "CV_16U";
    }
    else if (img.depth() == CV_16S)
    {
        mat_type = "CV_16S";
    }
    else if (img.depth() == CV_32S)
    {
        mat_type = "CV_32S";
    }
    else if (img.depth() == CV_32F)
    {
        mat_type = "CV_32F";
    }
    else if (img.depth() == CV_64F)
    {
        mat_type = "CV_64F";
    }
    cout << name;
    cout << format(": depth(%d) channels(%d) -> 자료형 : ", img.depth(), img.channels());
    cout << mat_type << "C" << img.channels() << endl;
}
```



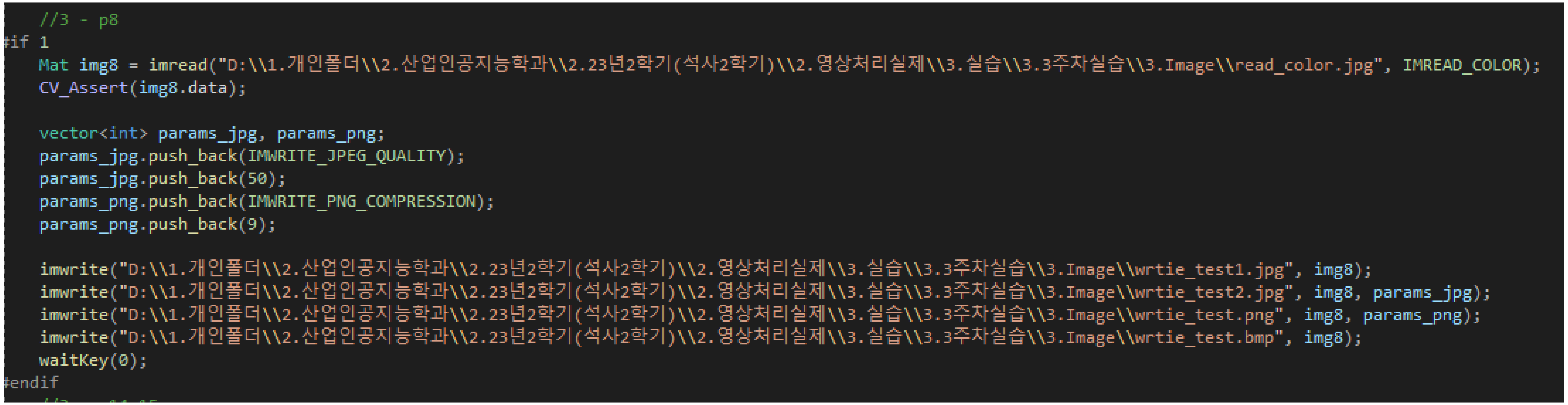
D:₩1.개인폴더₩2.산업인공지능학과₩2.23년2학기(석사2학기)₩2.영상처리실제₩3.실습₩3.3주차실습₩1.SRC₩Week_3_Test₩x64₩Debug₩Week_

```
행렬 좌표 (100,100) 화소값
color2gray [115]
color2color [127, 118, 105]
color2gray: depth(0) channels(1) -> 자료형 : CV_8UC1
color2color: depth(0) channels(3) -> 자료형 : CV_8UC3
```

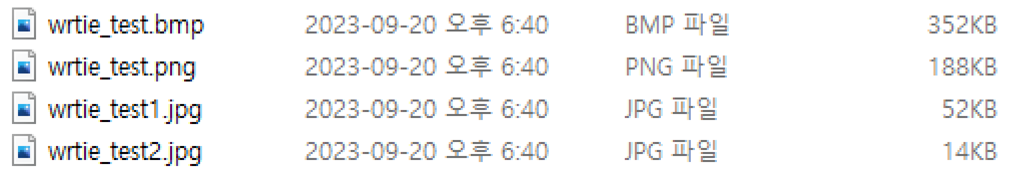




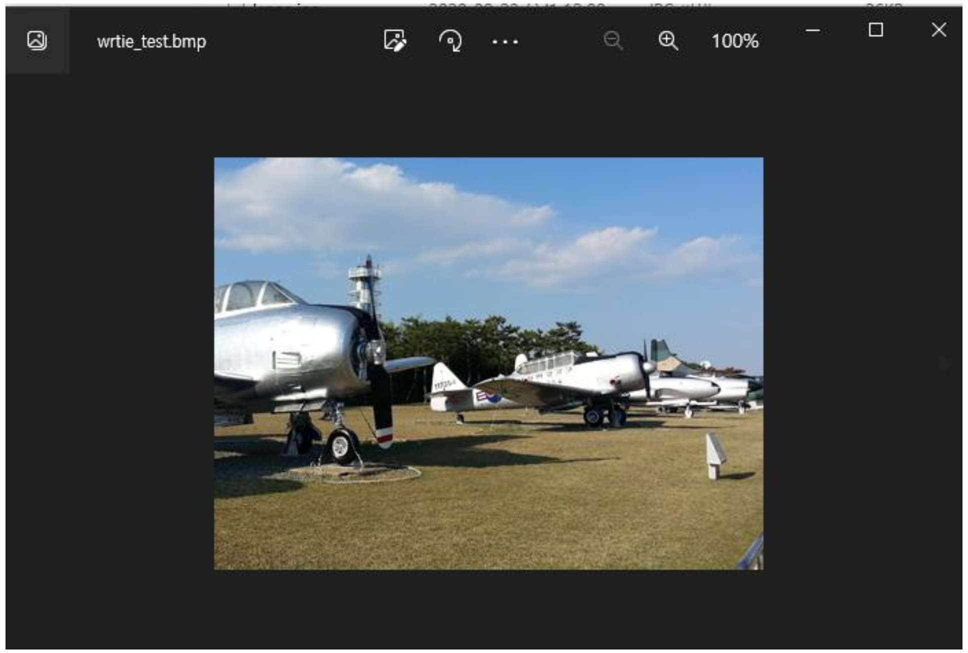
(3) – p.8



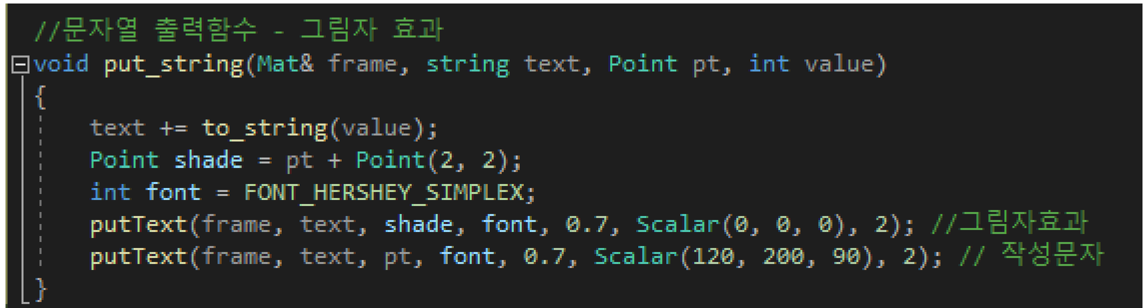
```
    //3 - p8
#if 1
    Mat img8 = imread("D:\\1.개인폴더\\2.산업인공지능학과\\2.23년2학기(석사2학기)\\2.영상처리실제\\3.실습\\3.3주차실습\\3.Image\\read_color.jpg", IMREAD_COLOR);
    CV_Assert(img8.data);

    vector<int> params_jpg, params_png;
    params_jpg.push_back(IMWRITE_JPEG_QUALITY);
    params_jpg.push_back(50);
    params_png.push_back(IMWRITE_PNG_COMPRESSION);
    params_png.push_back(9);

    imwrite("D:\\1.개인폴더\\2.산업인공지능학과\\2.23년2학기(석사2학기)\\2.영상처리실제\\3.실습\\3.3주차실습\\3.Image\\wrtie_test1.jpg", img8);
    imwrite("D:\\1.개인폴더\\2.산업인공지능학과\\2.23년2학기(석사2학기)\\2.영상처리실제\\3.실습\\3.3주차실습\\3.Image\\wrtie_test2.jpg", img8, params_jpg);
    imwrite("D:\\1.개인폴더\\2.산업인공지능학과\\2.23년2학기(석사2학기)\\2.영상처리실제\\3.실습\\3.3주차실습\\3.Image\\wrtie_test.png", img8, params_png);
    imwrite("D:\\1.개인폴더\\2.산업인공지능학과\\2.23년2학기(석사2학기)\\2.영상처리실제\\3.실습\\3.3주차실습\\3.Image\\wrtie_test.bmp", img8);
    waitKey(0);
#endif
```



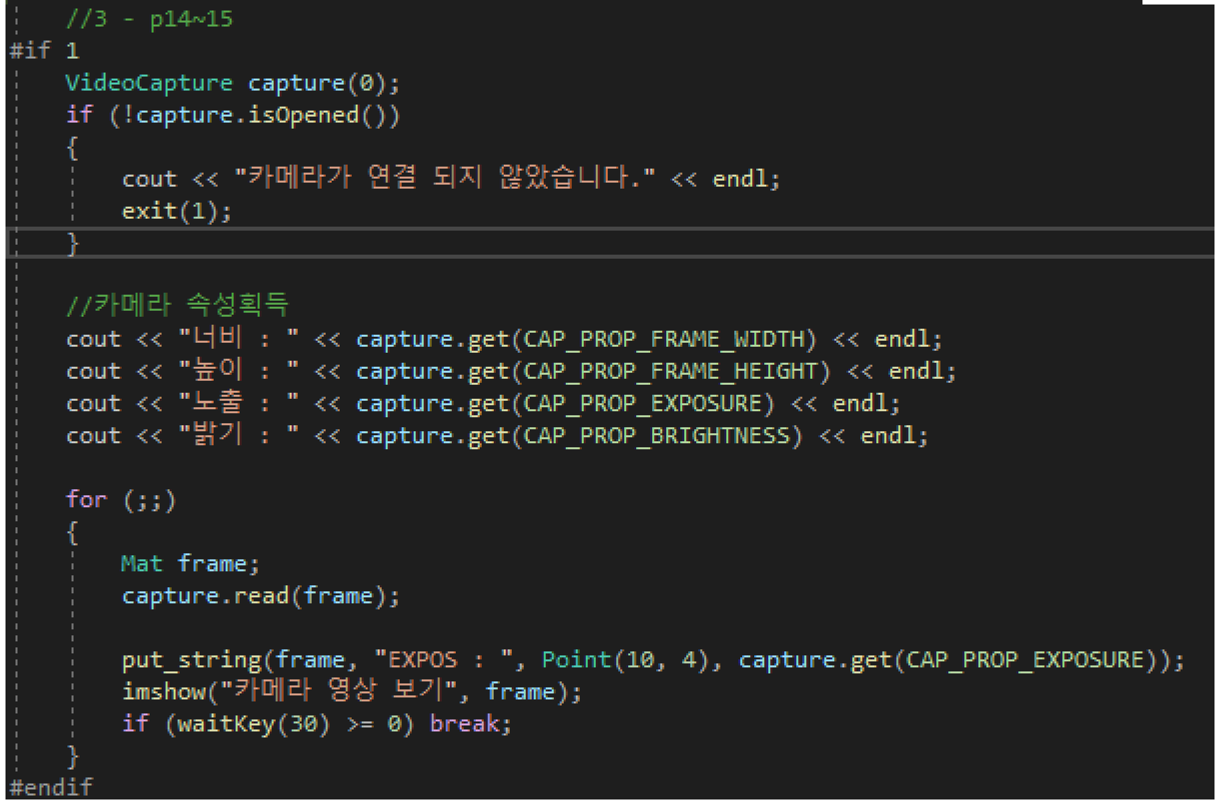
wrtie_test.bmp	2023-09-20 오후 6:40	BMP 파일	352KB
wrtie_test.png	2023-09-20 오후 6:40	PNG 파일	188KB
wrtie_test1.jpg	2023-09-20 오후 6:40	JPG 파일	52KB
wrtie_test2.jpg	2023-09-20 오후 6:40	JPG 파일	14KB



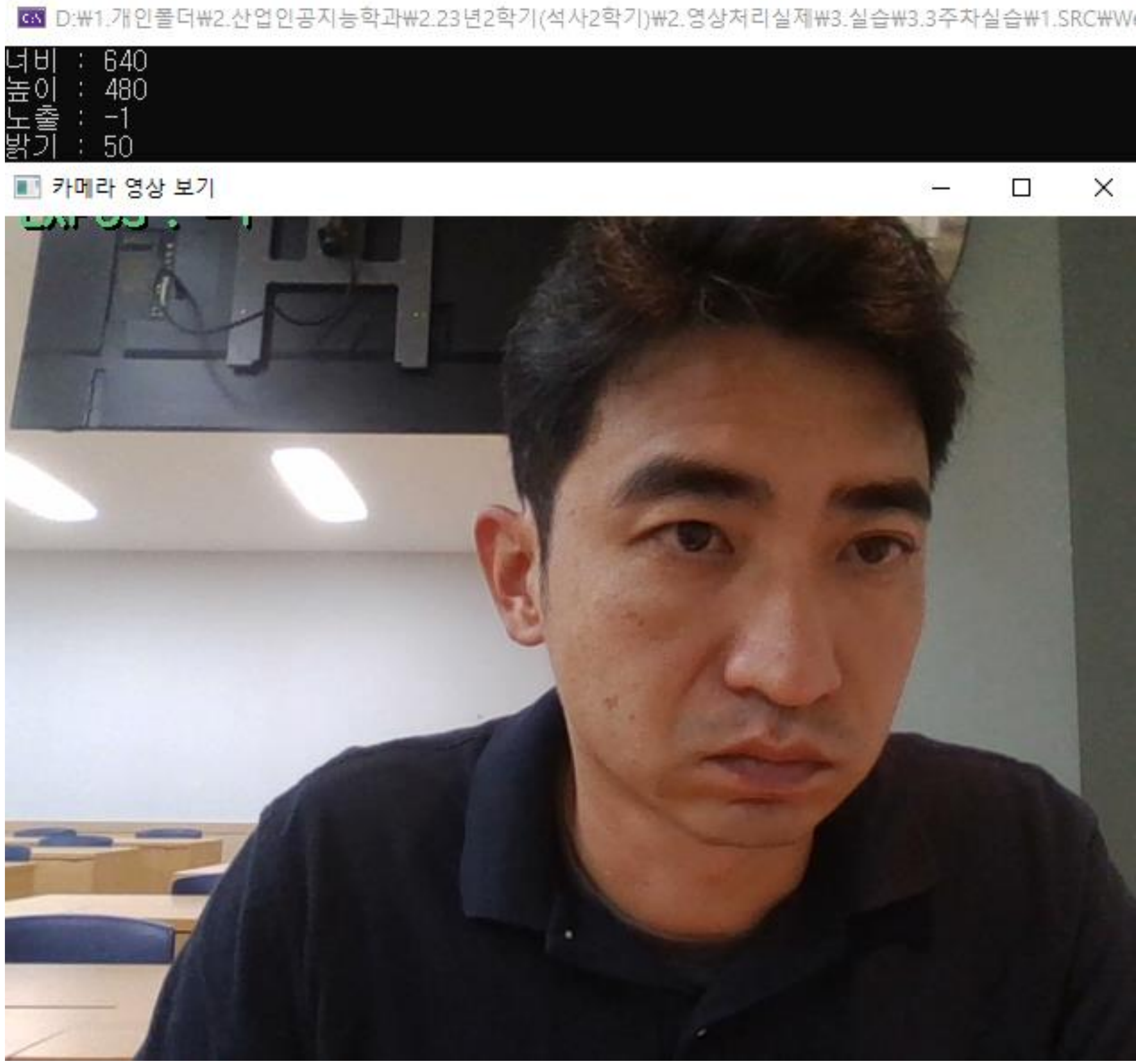
(3) – p.14 ~ 15



```cpp
//문자열 출력함수 - 그림자 효과
void put_string(Mat& frame, string text, Point pt, int value)
{
    text += to_string(value);
    Point shade = pt + Point(2, 2);
    int font = FONT_HERSHEY_SIMPLEX;
    putText(frame, text, shade, font, 0.7, Scalar(0, 0, 0), 2); //그림자효과
    putText(frame, text, pt, font, 0.7, Scalar(120, 200, 90), 2); // 작성문자
}
```



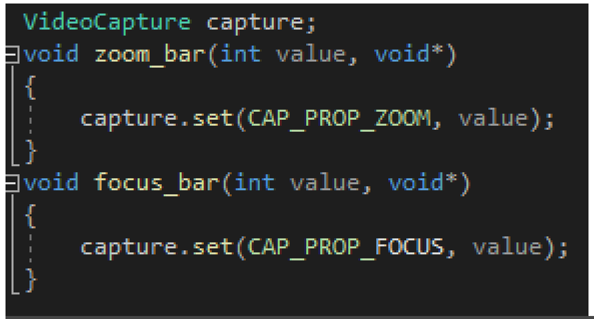
```cpp
    //3 - p14~15
#if 1
    VideoCapture capture(0);
    if (!capture.isOpened())
    {
        cout << "카메라가 연결 되지 않았습니다." << endl;
        exit(1);
    }

    //카메라 속성획득
    cout << "너비 : " << capture.get(CAP_PROP_FRAME_WIDTH) << endl;
    cout << "높이 : " << capture.get(CAP_PROP_FRAME_HEIGHT) << endl;
    cout << "노출 : " << capture.get(CAP_PROP_EXPOSURE) << endl;
    cout << "밝기 : " << capture.get(CAP_PROP_BRIGHTNESS) << endl;

    for (;;)
    {
        Mat frame;
        capture.read(frame);

        put_string(frame, "EXPOS : ", Point(10, 4), capture.get(CAP_PROP_EXPOSURE));
        imshow("카메라 영상 보기", frame);
        if (waitKey(30) >= 0) break;
    }
#endif
```



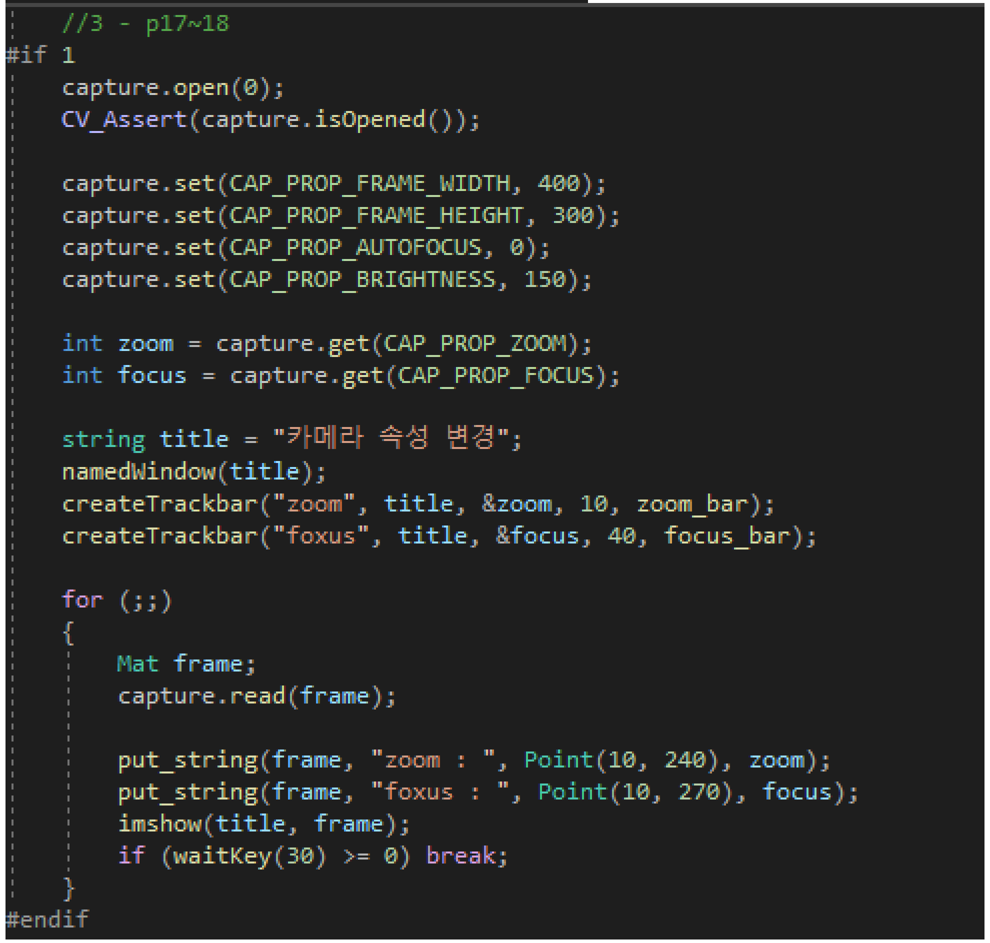
D:₩1.개인폴더₩2.산업인공지능학과₩2.23년2학기(석사2학기)₩2.영상처리실제₩3.실습₩3.3주차실습₩1.SRC₩₩

```
너비 : 640
높이 : 480
노출 : -1
밝기 : 50
```

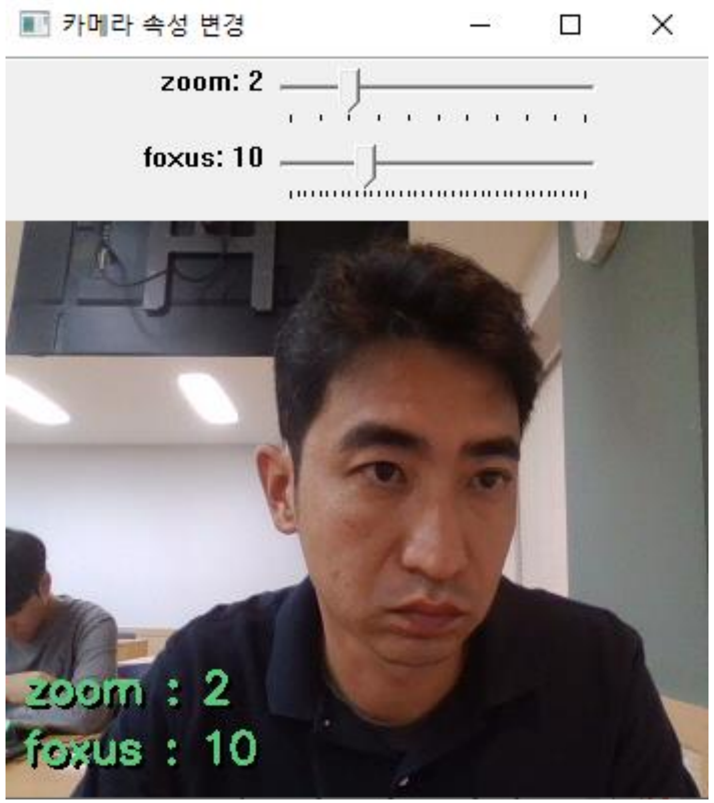
(3) – p.17 ~ 18



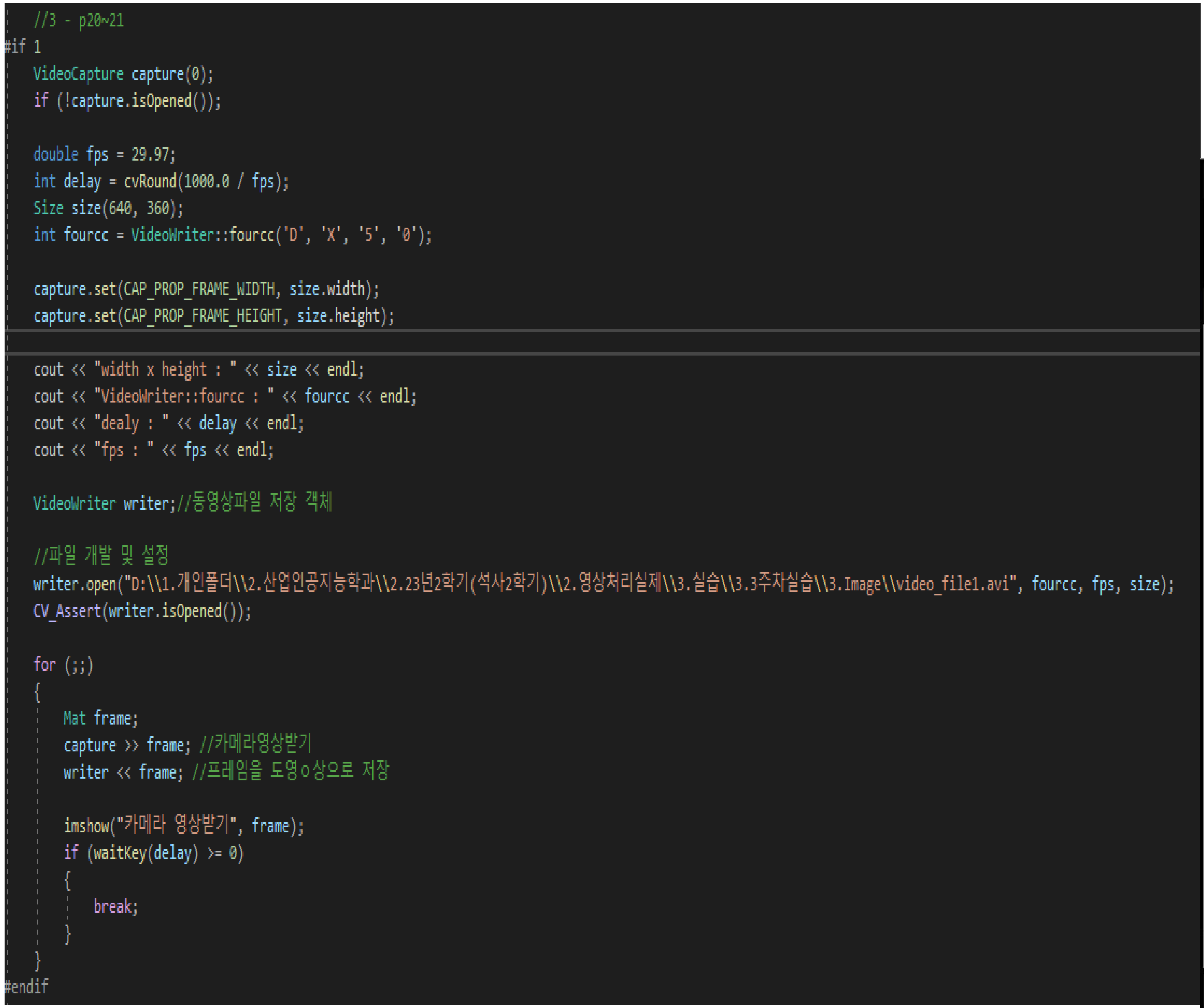
```
VideoCapture capture;
void zoom_bar(int value, void*)
{
    capture.set(CAP_PROP_ZOOM, value);
}
void focus_bar(int value, void*)
{
    capture.set(CAP_PROP_FOCUS, value);
}
```



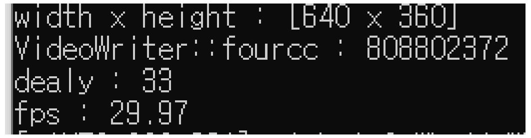
```
    //3 - p17~18
#if 1
    capture.open(0);
    CV_Assert(capture.isOpened());

    capture.set(CAP_PROP_FRAME_WIDTH, 400);
    capture.set(CAP_PROP_FRAME_HEIGHT, 300);
    capture.set(CAP_PROP_AUTOFOCUS, 0);
    capture.set(CAP_PROP_BRIGHTNESS, 150);

    int zoom = capture.get(CAP_PROP_ZOOM);
    int focus = capture.get(CAP_PROP_FOCUS);

    string title = "카메라 속성 변경";
    namedWindow(title);
    createTrackbar("zoom", title, &zoom, 10, zoom_bar);
    createTrackbar("foxus", title, &focus, 40, focus_bar);

    for (;;)
    {
        Mat frame;
        capture.read(frame);

        put_string(frame, "zoom : ", Point(10, 240), zoom);
        put_string(frame, "foxus : ", Point(10, 270), focus);
        imshow(title, frame);
        if (waitKey(30) >= 0) break;
    }
#endif
```



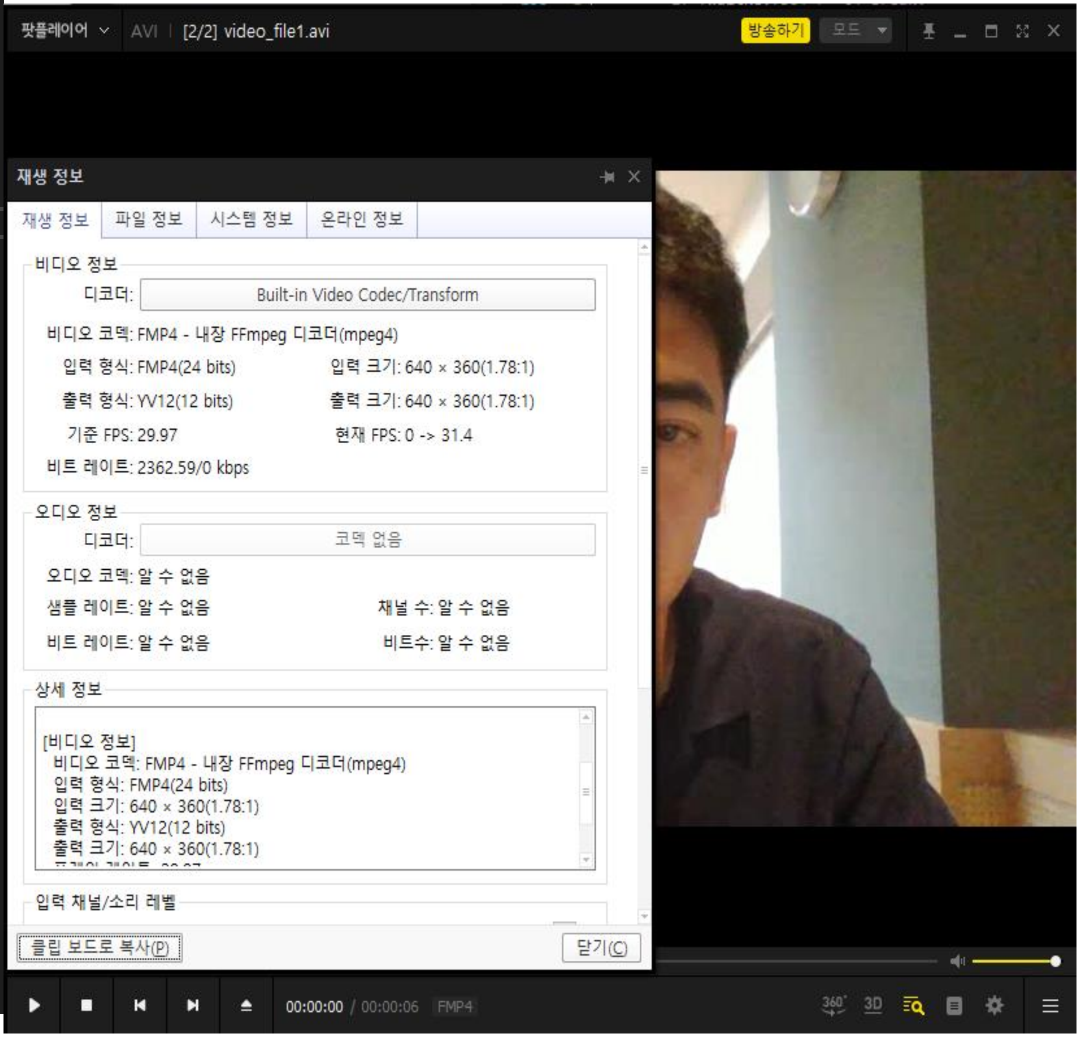
(3) – p.20 ~ 21



```
//3 - p20~21
#if 1
    VideoCapture capture(0);
    if (!capture.isOpened());

    double fps = 29.97;
    int delay = cvRound(1000.0 / fps);
    Size size(640, 360);
    int fourcc = VideoWriter::fourcc('D', 'X', '5', '0');

    capture.set(CAP_PROP_FRAME_WIDTH, size.width);
    capture.set(CAP_PROP_FRAME_HEIGHT, size.height);

    cout << "width x height : " << size << endl;
    cout << "VideoWriter::fourcc : " << fourcc << endl;
    cout << "dealy : " << delay << endl;
    cout << "fps : " << fps << endl;

    VideoWriter writer;//동영상파일 저장 객체

    //파일 개발 및 설정
    writer.open("D:\\1.개인폴더\\2.산업인공지능학과\\2.23년2학기(석사2학기)\\2.영상처리실제\\3.실습\\3.3주차실습\\3.Image\\video_file1.avi", fourcc, fps, size);
    CV_Assert(writer.isOpened());

    for (;;)
    {
        Mat frame;
        capture >> frame; //카메라영상받기
        writer << frame; //프레임을 도영ㅇ상으로 저장

        imshow("카메라 영상받기", frame);
        if (waitKey(delay) >= 0)
        {
            break;
        }
    }
#endif
```



```
width x height : [640 x 360]
VideoWriter::fourcc : 808802372
dealy : 33
fps : 29.97
```



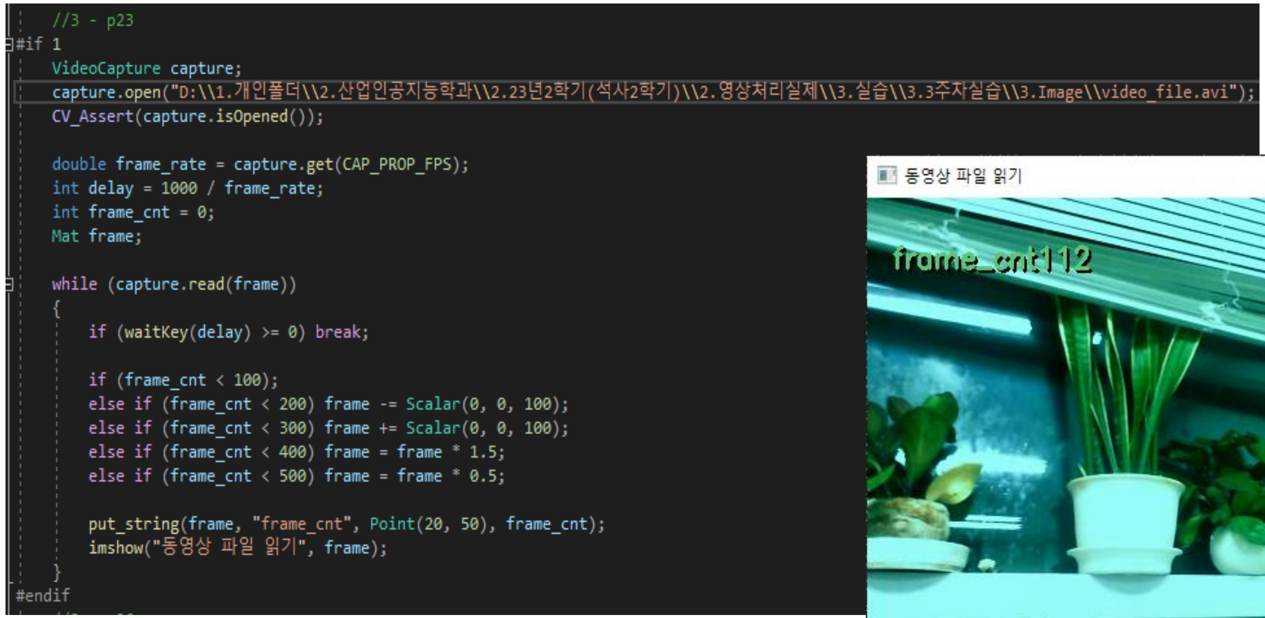
비디오 정보
디코더: Built-in Video Codec/Transform
비디오 코덱: FMP4 - 내장 FFmpeg 디코더(mpeg4)
입력 형식: FMP4(24 bits)　　입력 크기: 640 × 360(1.78:1)
출력 형식: YV12(12 bits)　　출력 크기: 640 × 360(1.78:1)
기준 FPS: 29.97　　현재 FPS: 0 -> 31.4
비트 레이트: 2362.59/0 kbps

오디오 정보
디코더: 코덱 없음
오디오 코덱: 알 수 없음
샘플 레이트: 알 수 없음　　채널 수: 알 수 없음
비트 레이트: 알 수 없음　　비트수: 알 수 없음

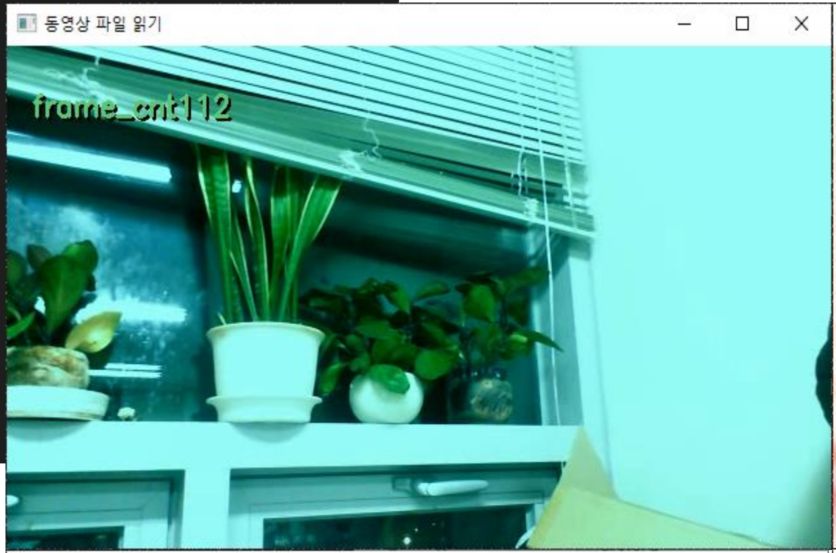
상세 정보
[비디오 정보]
비디오 코덱: FMP4 - 내장 FFmpeg 디코더(mpeg4)
입력 형식: FMP4(24 bits)
입력 크기: 640 × 360(1.78:1)
출력 형식: YV12(12 bits)
출력 크기: 640 × 360(1.78:1)

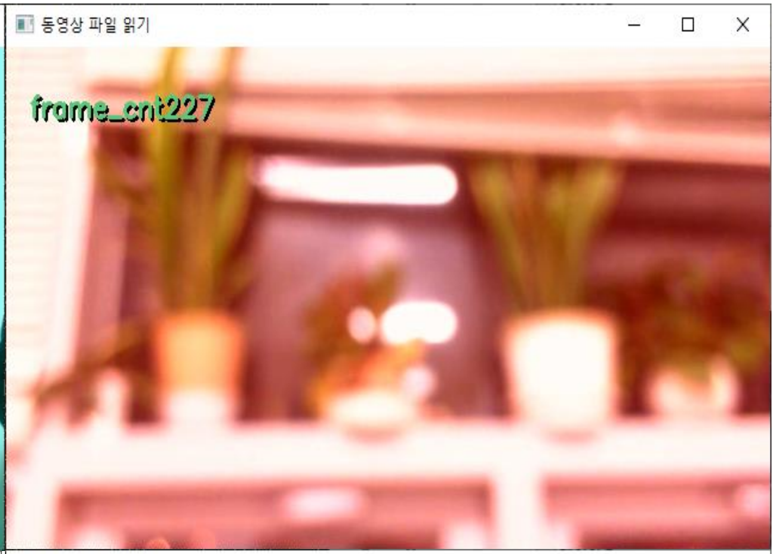
입력 채널/소리 레벨

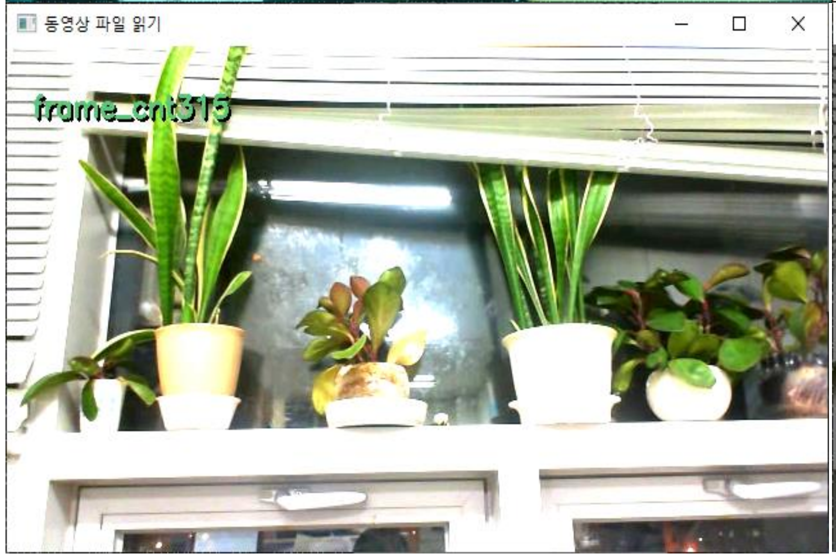
(3) – p.23

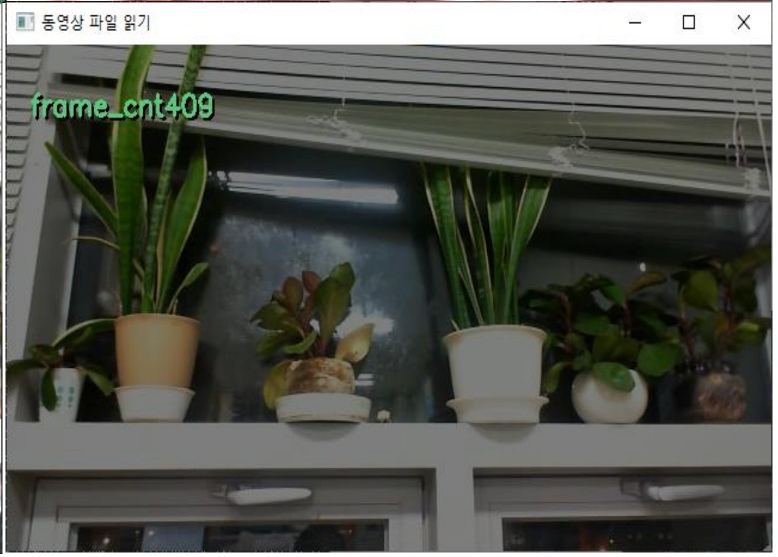


```
    //3 - p23
#if 1
    VideoCapture capture;
    capture.open("D:\\1.개인폴더\\2.산업인공지능학과\\2.23년2학기(석사2학기)\\2.영상처리실제\\3.실습\\3.3주차실습\\3.Image\\video_file.avi");
    CV_Assert(capture.isOpened());

    double frame_rate = capture.get(CAP_PROP_FPS);
    int delay = 1000 / frame_rate;
    int frame_cnt = 0;
    Mat frame;

    while (capture.read(frame))
    {
        if (waitKey(delay) >= 0) break;

        if (frame_cnt < 100);
        else if (frame_cnt < 200) frame -= Scalar(0, 0, 100);
        else if (frame_cnt < 300) frame += Scalar(0, 0, 100);
        else if (frame_cnt < 400) frame = frame * 1.5;
        else if (frame_cnt < 500) frame = frame * 0.5;

        put_string(frame, "frame_cnt", Point(20, 50), frame_cnt);
        imshow("동영상 파일 읽기", frame);
    }
#endif
```

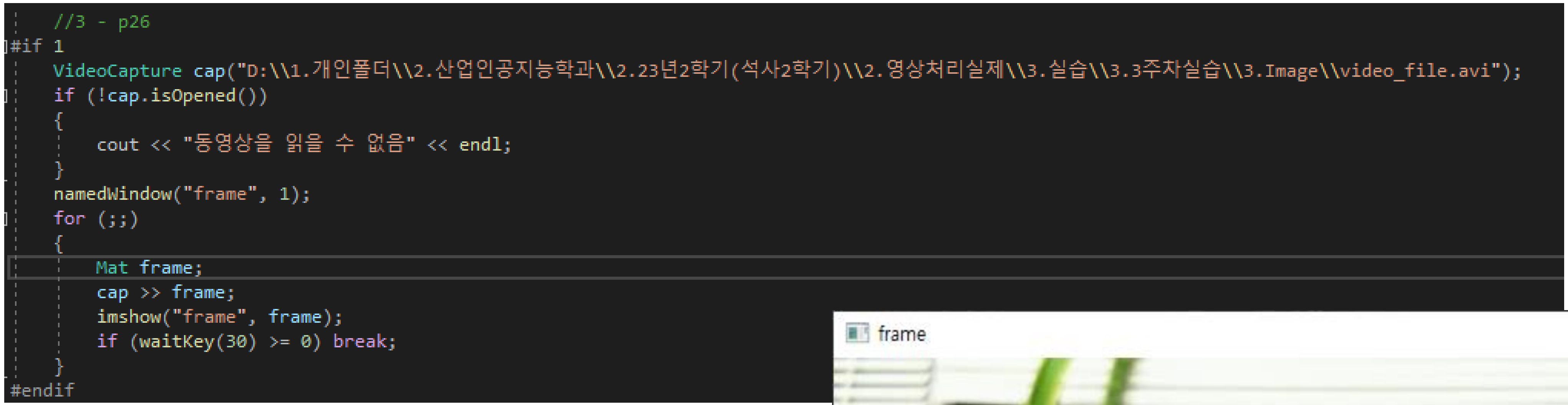




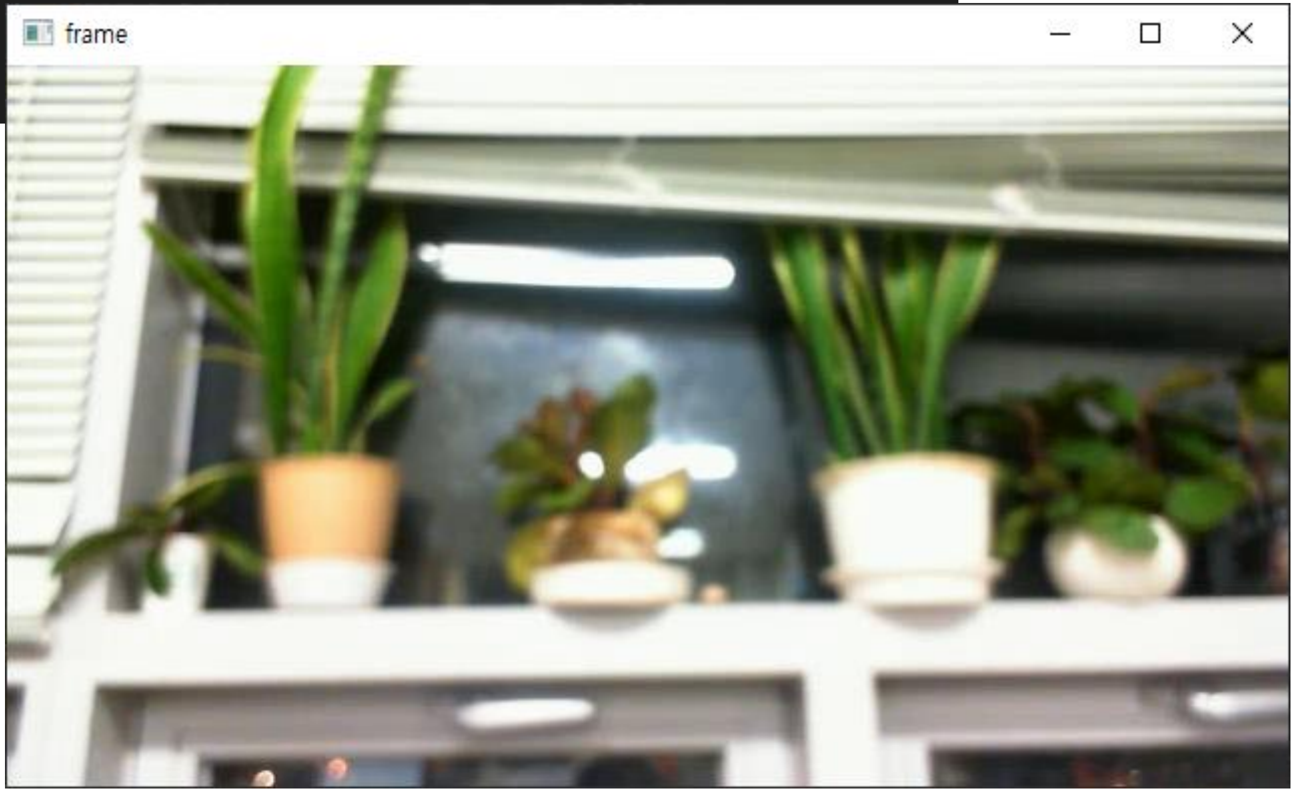




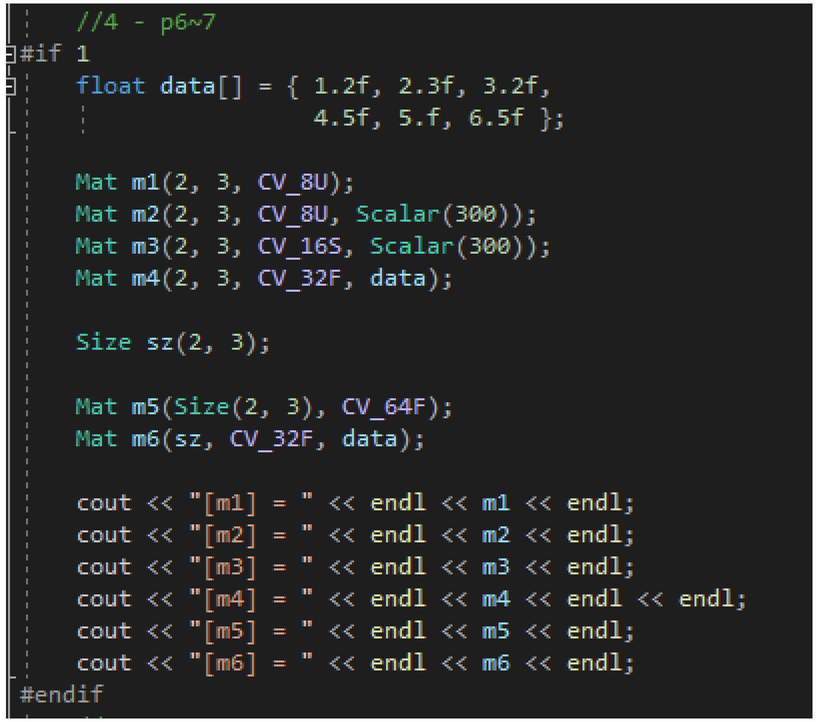
(3) – p.26



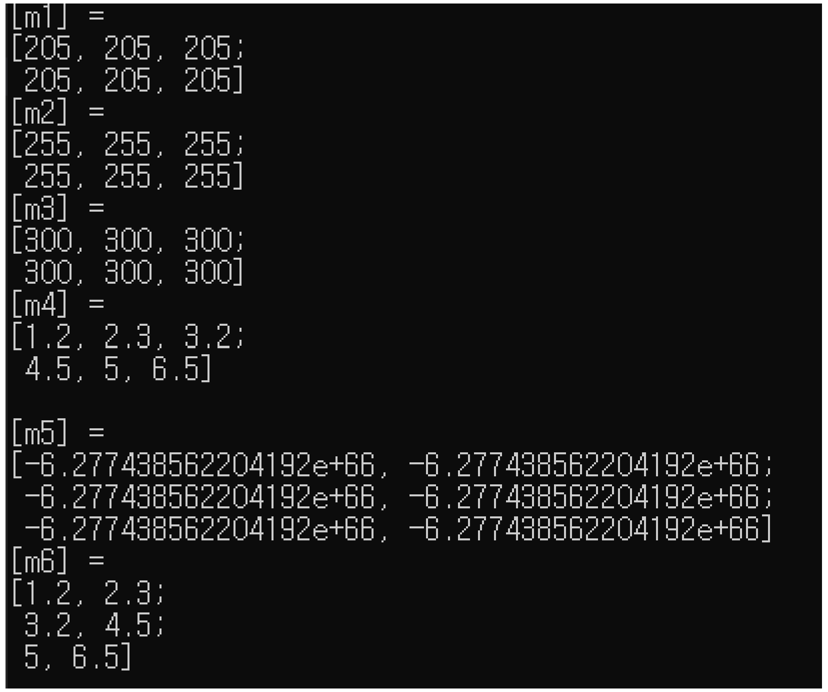
```
    //3 - p26
#if 1
    VideoCapture cap("D:\\1.개인폴더\\2.산업인공지능학과\\2.23년2학기(석사2학기)\\2.영상처리실제\\3.실습\\3.3주차실습\\3.Image\\video_file.avi");
    if (!cap.isOpened())
    {
        cout << "동영상을 읽을 수 없음" << endl;
    }
    namedWindow("frame", 1);
    for (;;)
    {
        Mat frame;
        cap >> frame;
        imshow("frame", frame);
        if (waitKey(30) >= 0) break;
    }
#endif
```



(4) – p.6~7

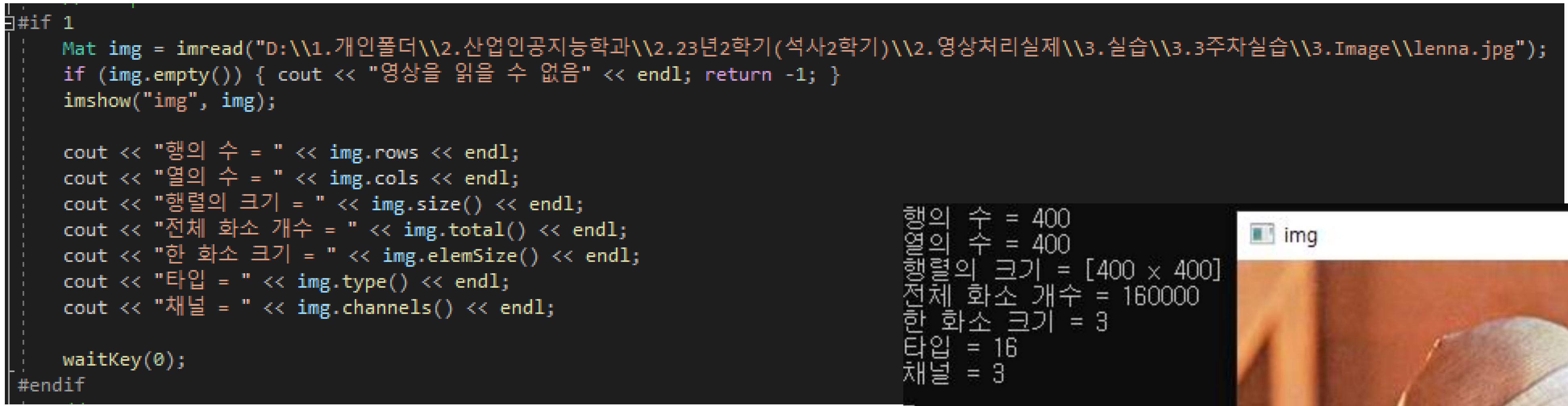


```
    //4 - p6~7
#if 1
    float data[] = { 1.2f, 2.3f, 3.2f,
                     4.5f, 5.f, 6.5f };

    Mat m1(2, 3, CV_8U);
    Mat m2(2, 3, CV_8U, Scalar(300));
    Mat m3(2, 3, CV_16S, Scalar(300));
    Mat m4(2, 3, CV_32F, data);

    Size sz(2, 3);

    Mat m5(Size(2, 3), CV_64F);
    Mat m6(sz, CV_32F, data);

    cout << "[m1] = " << endl << m1 << endl;
    cout << "[m2] = " << endl << m2 << endl;
    cout << "[m3] = " << endl << m3 << endl;
    cout << "[m4] = " << endl << m4 << endl << endl;
    cout << "[m5] = " << endl << m5 << endl;
    cout << "[m6] = " << endl << m6 << endl;
#endif
```

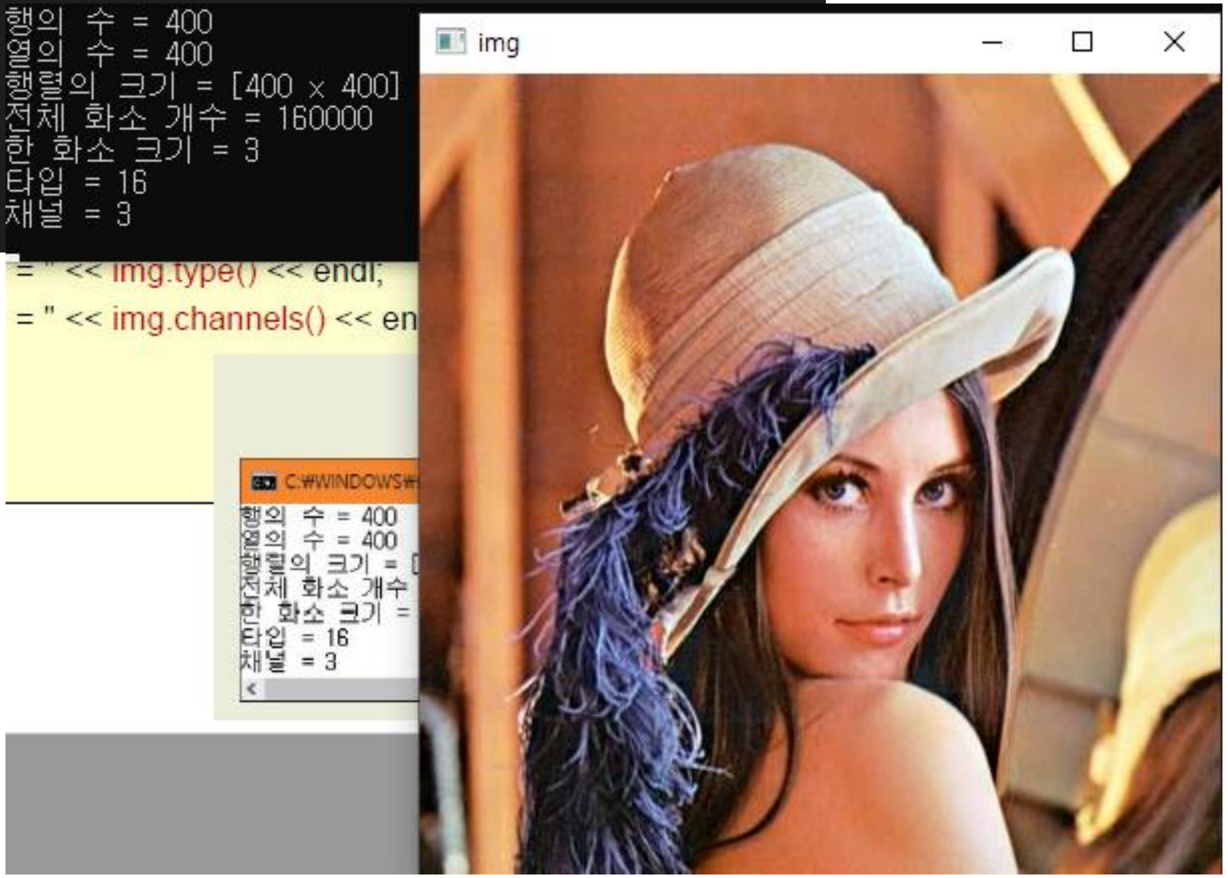


```
[m1] =
[205, 205, 205;
 205, 205, 205]
[m2] =
[255, 255, 255;
 255, 255, 255]
[m3] =
[300, 300, 300;
 300, 300, 300]
[m4] =
[1.2, 2.3, 3.2;
 4.5, 5, 6.5]

[m5] =
[-6.277438562204192e+66, -6.277438562204192e+66;
 -6.277438562204192e+66, -6.277438562204192e+66;
 -6.277438562204192e+66, -6.277438562204192e+66]
[m6] =
[1.2, 2.3;
 3.2, 4.5;
 5, 6.5]
```

(4) – p.17



```cpp
#if 1
    Mat img = imread("D:\\1.개인폴더\\2.산업인공지능학과\\2.23년2학기(석사2학기)\\2.영상처리실제\\3.실습\\3.3주차실습\\3.Image\\lenna.jpg");
    if (img.empty()) { cout << "영상을 읽을 수 없음" << endl; return -1; }
    imshow("img", img);

    cout << "행의 수 = " << img.rows << endl;
    cout << "열의 수 = " << img.cols << endl;
    cout << "행렬의 크기 = " << img.size() << endl;
    cout << "전체 화소 개수 = " << img.total() << endl;
    cout << "한 화소 크기 = " << img.elemSize() << endl;
    cout << "타입 = " << img.type() << endl;
    cout << "채널 = " << img.channels() << endl;

    waitKey(0);
#endif
```

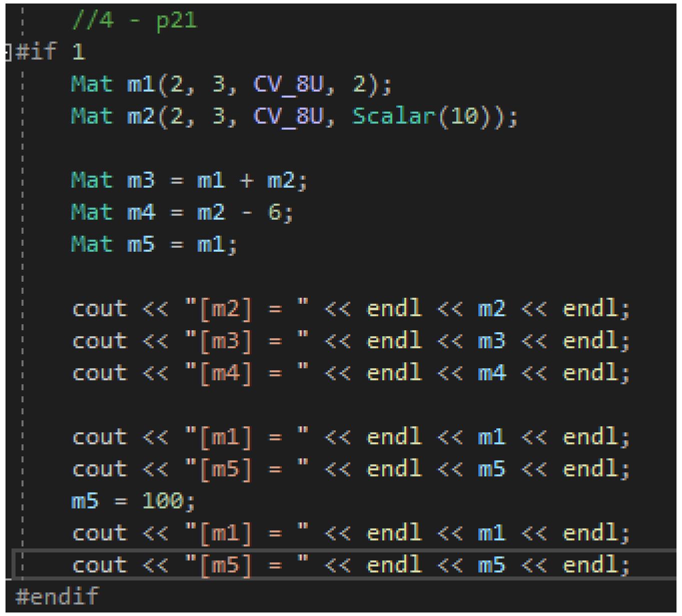


```
행의 수 = 400
열의 수 = 400
행렬의 크기 = [400 x 400]
전체 화소 개수 = 160000
한 화소 크기 = 3
타입 = 16
채널 = 3
```

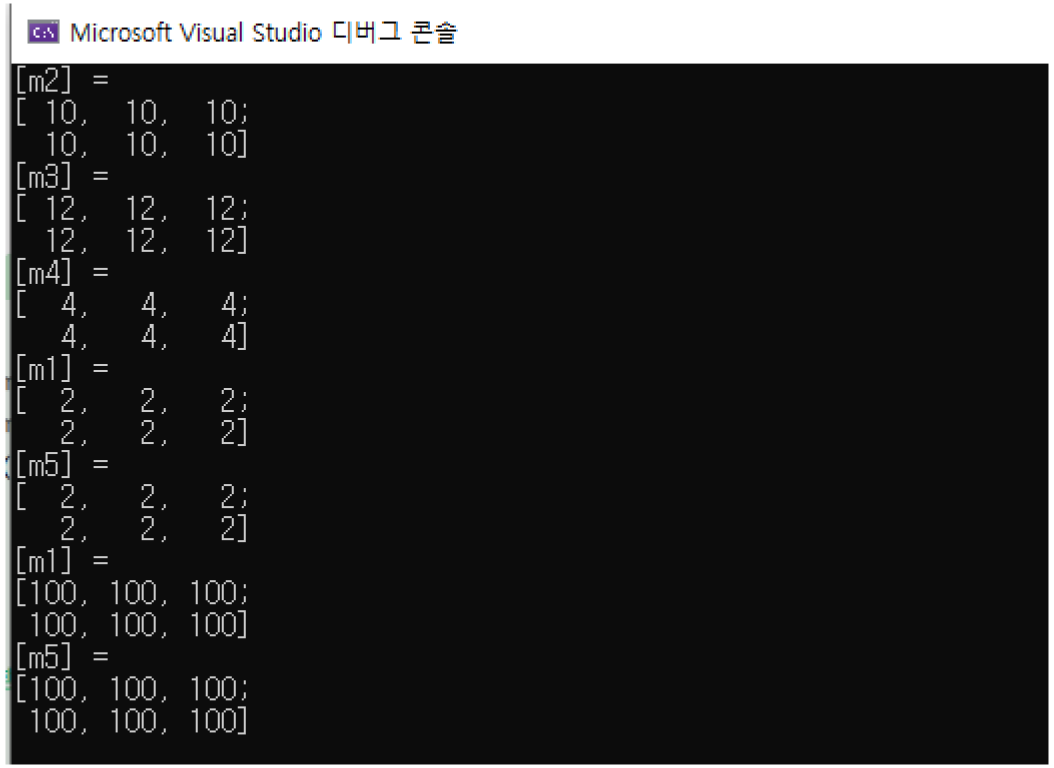
```
= " << img.type() << endl;
= " << img.channels() << en
```

```
행의 수 = 400
열의 수 = 400
행렬의 크기 = [
전체 화소 개수
한 화소 크기 =
타입 = 16
채널 = 3
```

(4) – p.21



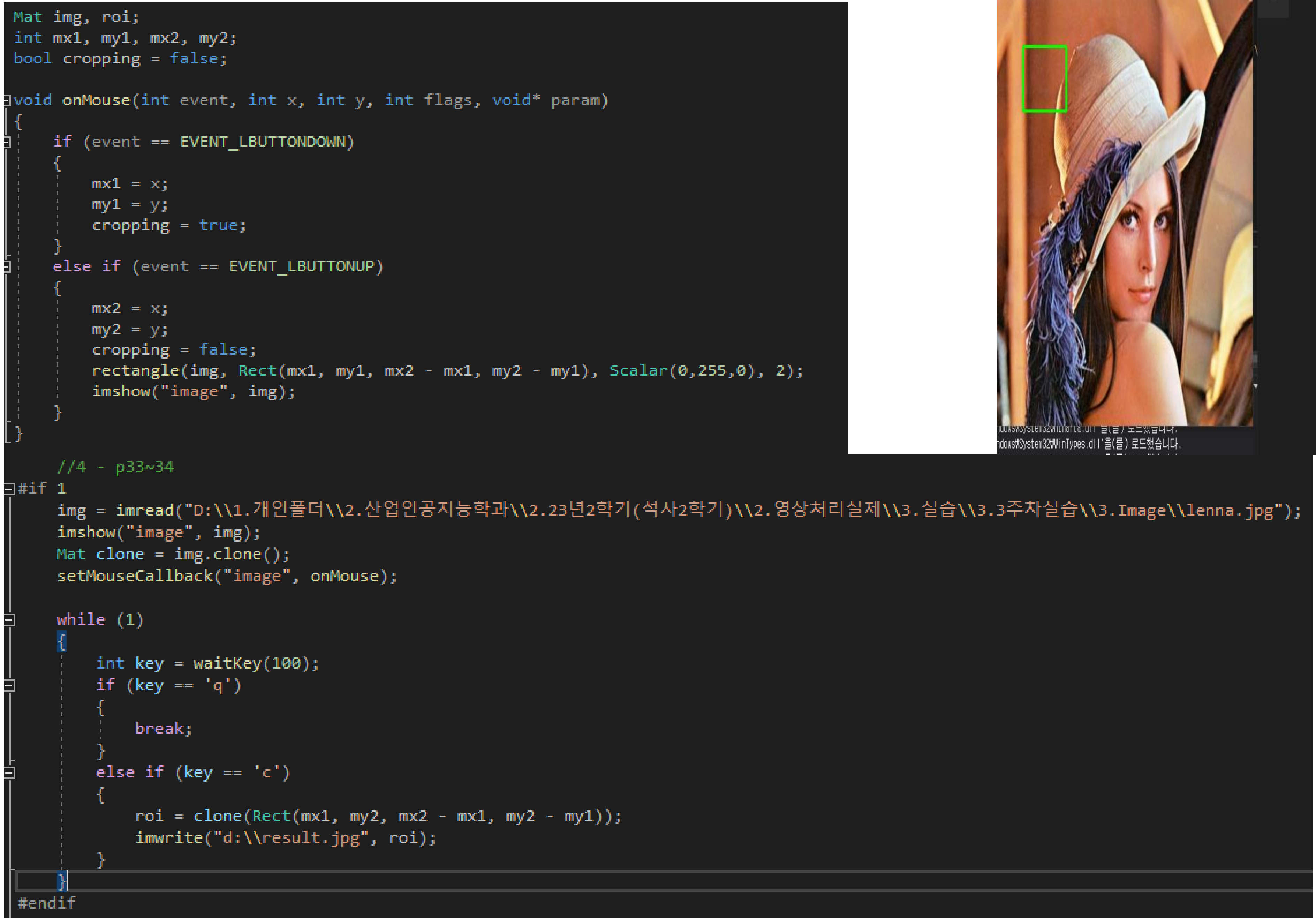
```
    //4 - p21
#if 1
    Mat m1(2, 3, CV_8U, 2);
    Mat m2(2, 3, CV_8U, Scalar(10));

    Mat m3 = m1 + m2;
    Mat m4 = m2 - 6;
    Mat m5 = m1;

    cout << "[m2] = " << endl << m2 << endl;
    cout << "[m3] = " << endl << m3 << endl;
    cout << "[m4] = " << endl << m4 << endl;

    cout << "[m1] = " << endl << m1 << endl;
    cout << "[m5] = " << endl << m5 << endl;
    m5 = 100;
    cout << "[m1] = " << endl << m1 << endl;
    cout << "[m5] = " << endl << m5 << endl;
#endif
```



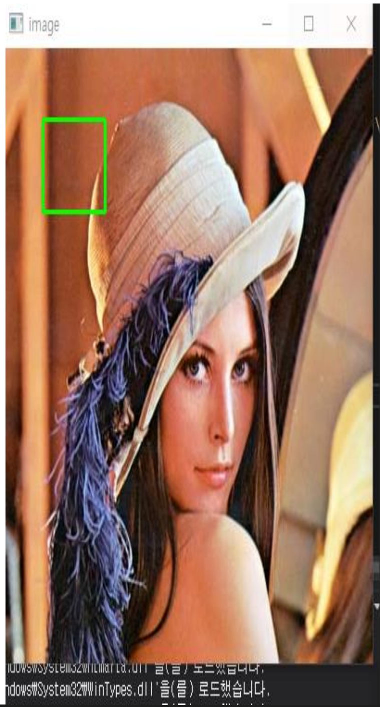
Microsoft Visual Studio 디버그 콘솔

```
[m2] =
[ 10,  10,  10;
  10,  10,  10]
[m3] =
[ 12,  12,  12;
  12,  12,  12]
[m4] =
[  4,   4,   4;
   4,   4,   4]
[m1] =
[  2,   2,   2;
   2,   2,   2]
[m5] =
[  2,   2,   2;
   2,   2,   2]
[m1] =
[100, 100, 100;
 100, 100, 100]
[m5] =
[100, 100, 100;
 100, 100, 100]
```

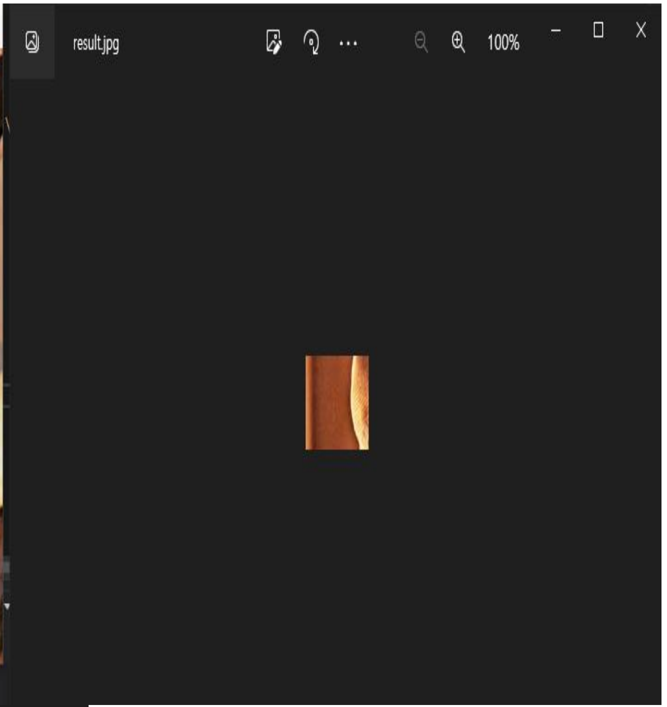
(4) – p.33 ~ 34



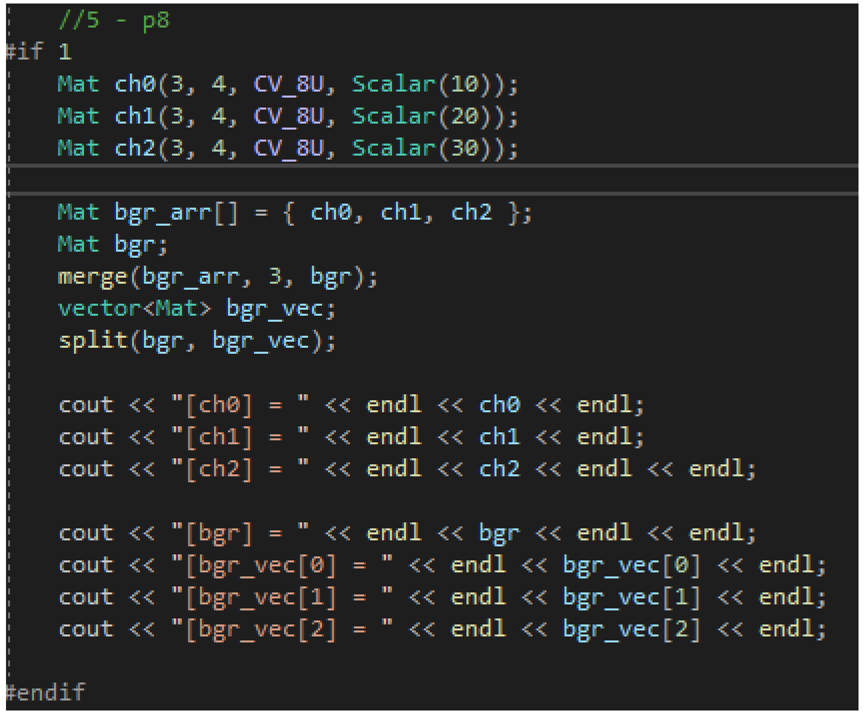
```
Mat img, roi;
int mx1, my1, mx2, my2;
bool cropping = false;

void onMouse(int event, int x, int y, int flags, void* param)
{
    if (event == EVENT_LBUTTONDOWN)
    {
        mx1 = x;
        my1 = y;
        cropping = true;
    }
    else if (event == EVENT_LBUTTONUP)
    {
        mx2 = x;
        my2 = y;
        cropping = false;
        rectangle(img, Rect(mx1, my1, mx2 - mx1, my2 - my1), Scalar(0,255,0), 2);
        imshow("image", img);
    }
}

    //4 - p33~34
#if 1
    img = imread("D:\\1.개인폴더\\2.산업인공지능학과\\2.23년2학기(석사2학기)\\2.영상처리실제\\3.실습\\3.3주차실습\\3.Image\\lenna.jpg");
    imshow("image", img);
    Mat clone = img.clone();
    setMouseCallback("image", onMouse);

    while (1)
    {
        int key = waitKey(100);
        if (key == 'q')
        {
            break;
        }
        else if (key == 'c')
        {
            roi = clone(Rect(mx1, my2, mx2 - mx1, my2 - my1));
            imwrite("d:\\result.jpg", roi);
        }
    }
#endif
```



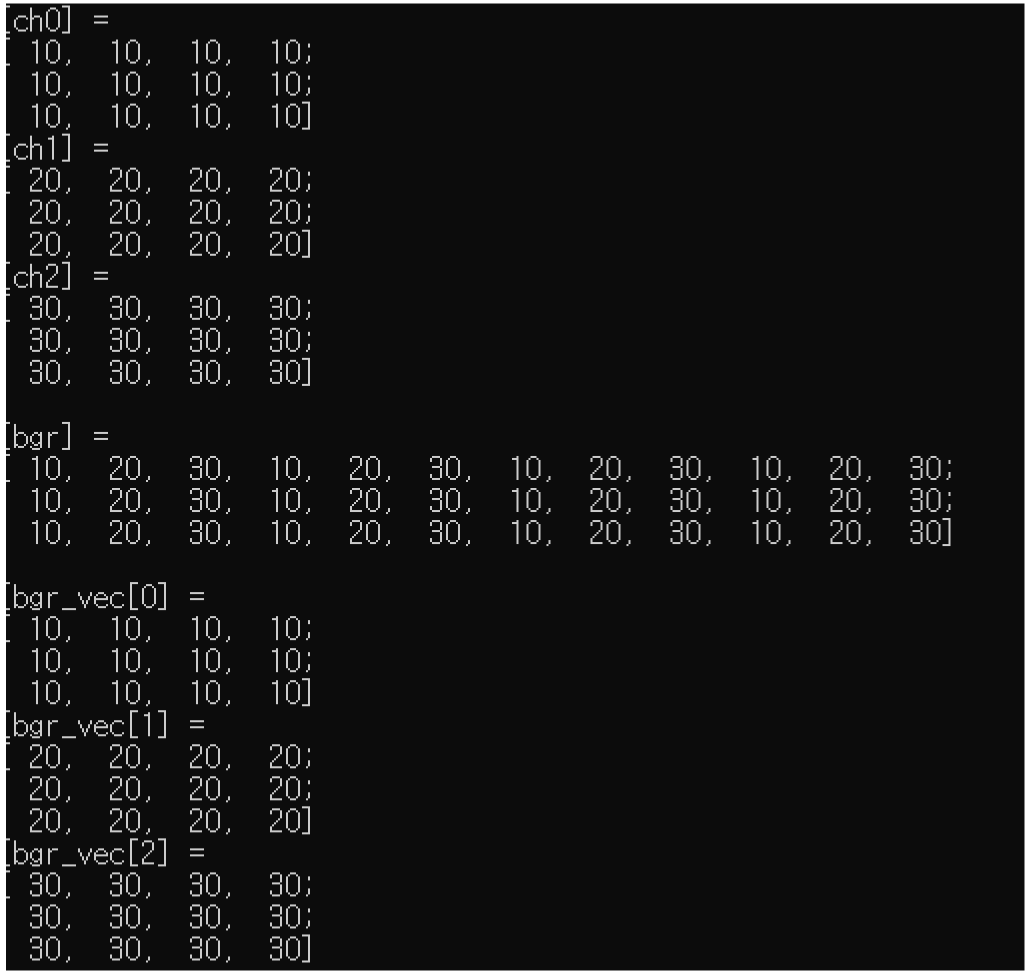
```
ndows\System32\WinTypes.dll'을(를) 로드했습니다.
```



(5) – p.8

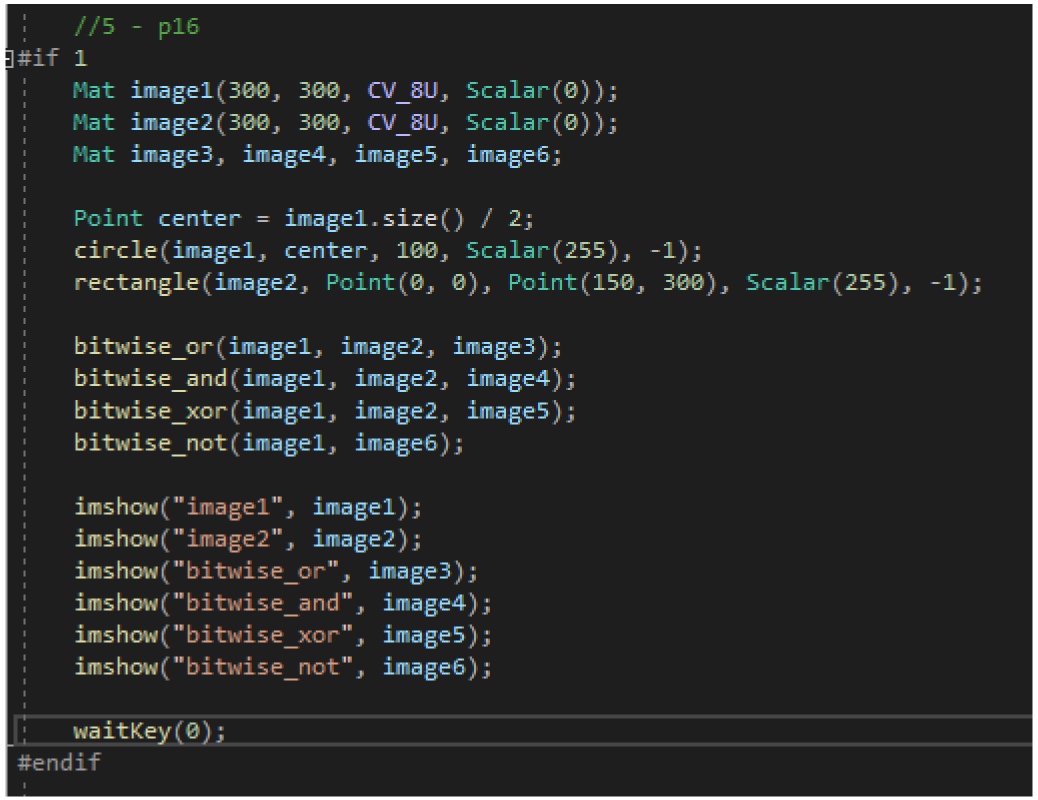


```
    //5 - p8
#if 1
    Mat ch0(3, 4, CV_8U, Scalar(10));
    Mat ch1(3, 4, CV_8U, Scalar(20));
    Mat ch2(3, 4, CV_8U, Scalar(30));

    Mat bgr_arr[] = { ch0, ch1, ch2 };
    Mat bgr;
    merge(bgr_arr, 3, bgr);
    vector<Mat> bgr_vec;
    split(bgr, bgr_vec);

    cout << "[ch0] = " << endl << ch0 << endl;
    cout << "[ch1] = " << endl << ch1 << endl;
    cout << "[ch2] = " << endl << ch2 << endl << endl;

    cout << "[bgr] = " << endl << bgr << endl << endl;
    cout << "[bgr_vec[0] = " << endl << bgr_vec[0] << endl;
    cout << "[bgr_vec[1] = " << endl << bgr_vec[1] << endl;
    cout << "[bgr_vec[2] = " << endl << bgr_vec[2] << endl;

#endif
```



```
[ch0] =
[ 10,  10,  10,  10;
  10,  10,  10,  10;
  10,  10,  10,  10]
[ch1] =
[ 20,  20,  20,  20;
  20,  20,  20,  20;
  20,  20,  20,  20]
[ch2] =
[ 30,  30,  30,  30;
  30,  30,  30,  30;
  30,  30,  30,  30]

[bgr] =
[ 10,  20,  30,  10,  20,  30,  10,  20,  30,  10,  20,  30;
  10,  20,  30,  10,  20,  30,  10,  20,  30,  10,  20,  30;
  10,  20,  30,  10,  20,  30,  10,  20,  30,  10,  20,  30]

[bgr_vec[0] =
[ 10,  10,  10,  10;
  10,  10,  10,  10;
  10,  10,  10,  10]
[bgr_vec[1] =
[ 20,  20,  20,  20;
  20,  20,  20,  20;
  20,  20,  20,  20]
[bgr_vec[2] =
[ 30,  30,  30,  30;
  30,  30,  30,  30;
  30,  30,  30,  30]
```

(5) – p.16



```
    //5 - p16
#if 1
    Mat image1(300, 300, CV_8U, Scalar(0));
    Mat image2(300, 300, CV_8U, Scalar(0));
    Mat image3, image4, image5, image6;

    Point center = image1.size() / 2;
    circle(image1, center, 100, Scalar(255), -1);
    rectangle(image2, Point(0, 0), Point(150, 300), Scalar(255), -1);

    bitwise_or(image1, image2, image3);
    bitwise_and(image1, image2, image4);
    bitwise_xor(image1, image2, image5);
    bitwise_not(image1, image6);

    imshow("image1", image1);
    imshow("image2", image2);
    imshow("bitwise_or", image3);
    imshow("bitwise_and", image4);
    imshow("bitwise_xor", image5);
    imshow("bitwise_not", image6);

    waitKey(0);
#endif
```

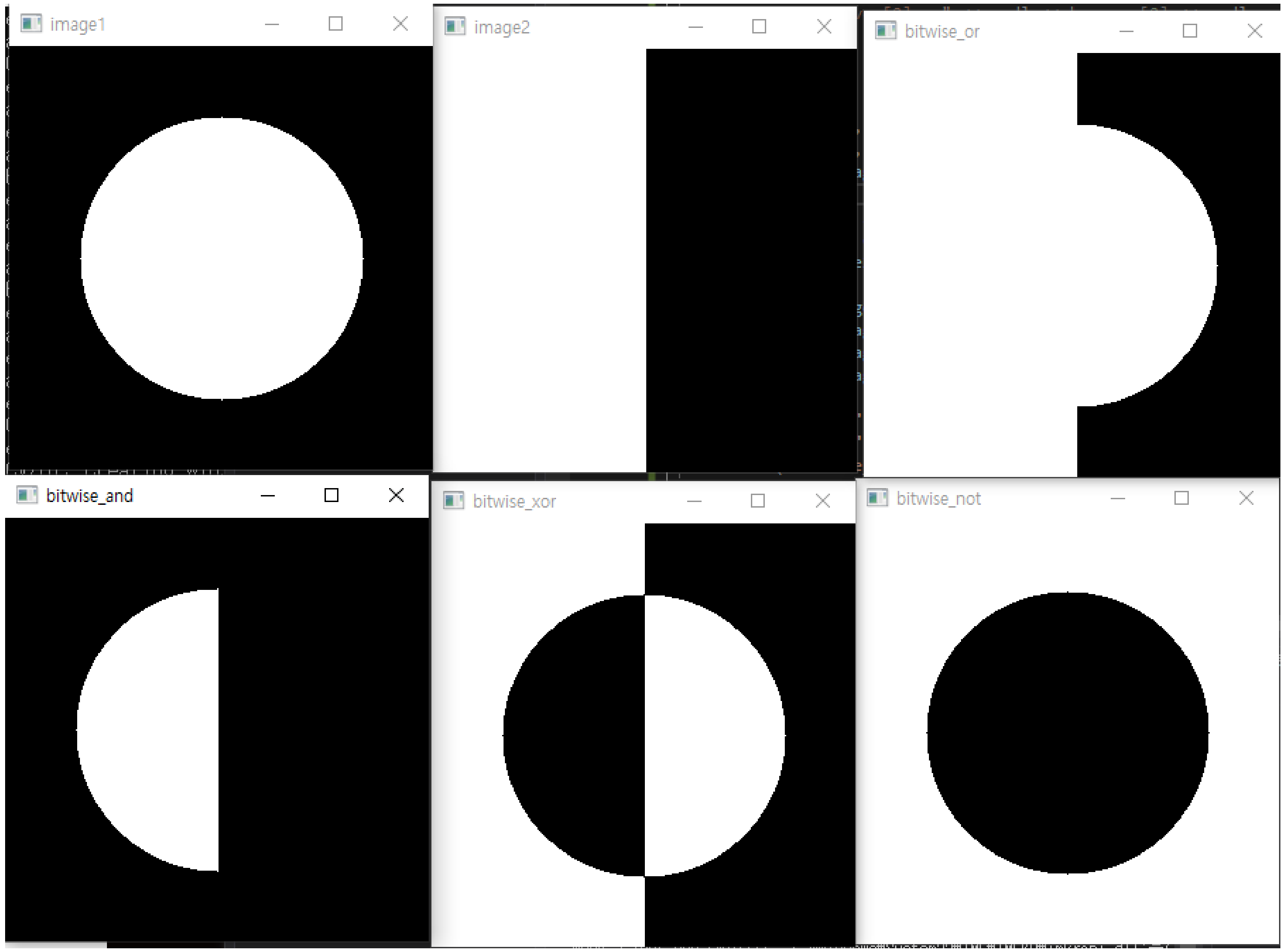


image1
image2
bitwise_or
bitwise_and
bitwise_xor
bitwise_not