



DIP(Digital Image Processing)

Program Exercise

(해당 강의자료의 배포 및 무단 복제를 금함)

김남욱, 김명준, 정지연, 김양우, 이영렬

Sejong University, DMS Lab.

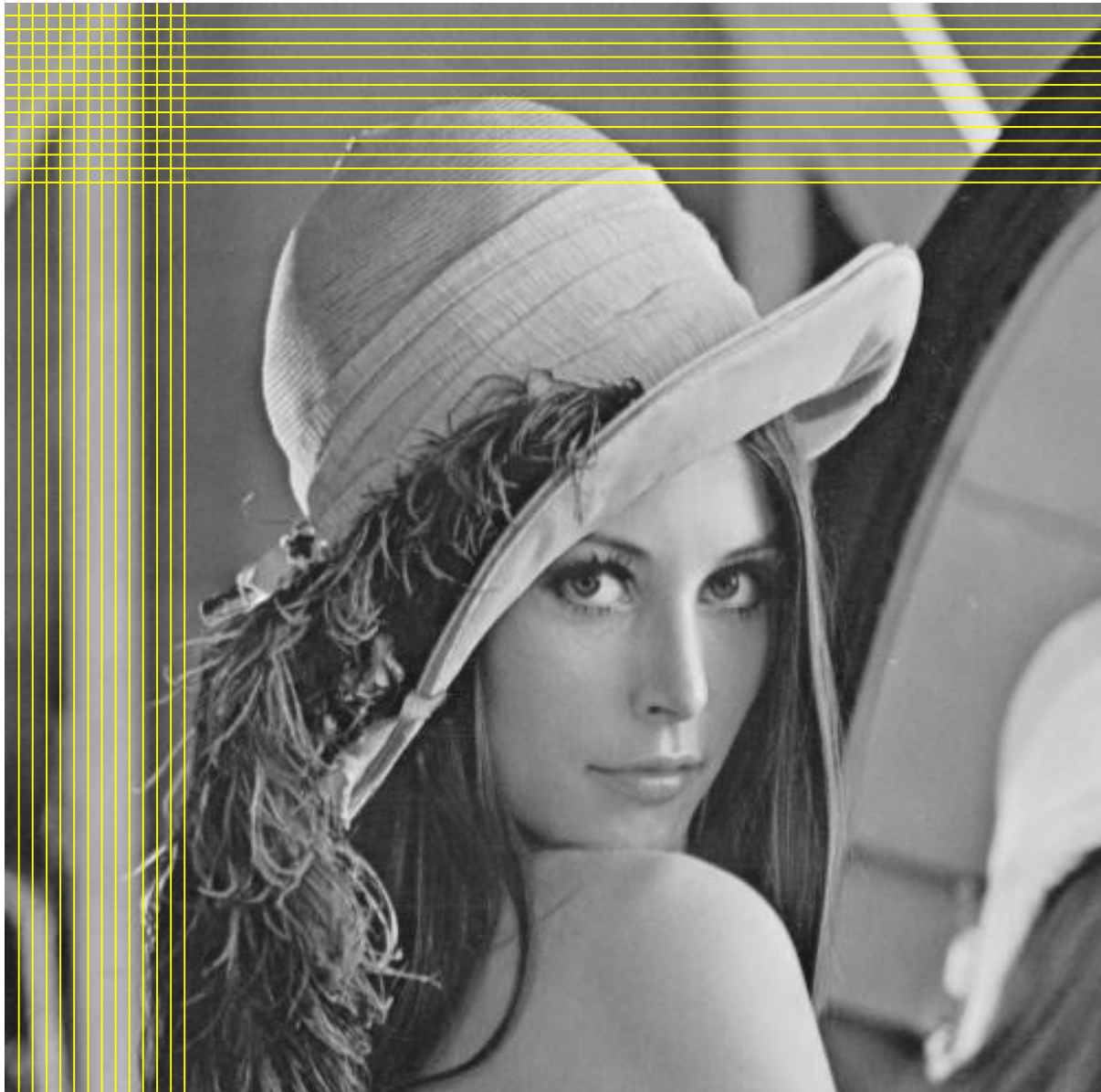
Program Exercise 1

1. Zoom-out
 2. Zoom-in
 3. Low-pass filtering (average filtering)
 4. High-pass filtering (Sobel operator)
 5. Zoom-out and translation
 6. Zoom-out and translation and rotation
 7. histogram equalization(or Histogram slide-mapping & histogram stretch-mapping)
- 총 10점 만점
 - 1~4번 각 1점
 - 5~7번 각 2점
 - 채점 및 검사
 - 장소 : 실습시간 중 또는 대양AI센터 817호
 - 조교 : 김남욱, 김명준, 정지연, 김양우

실습이미지 - **lena.img**

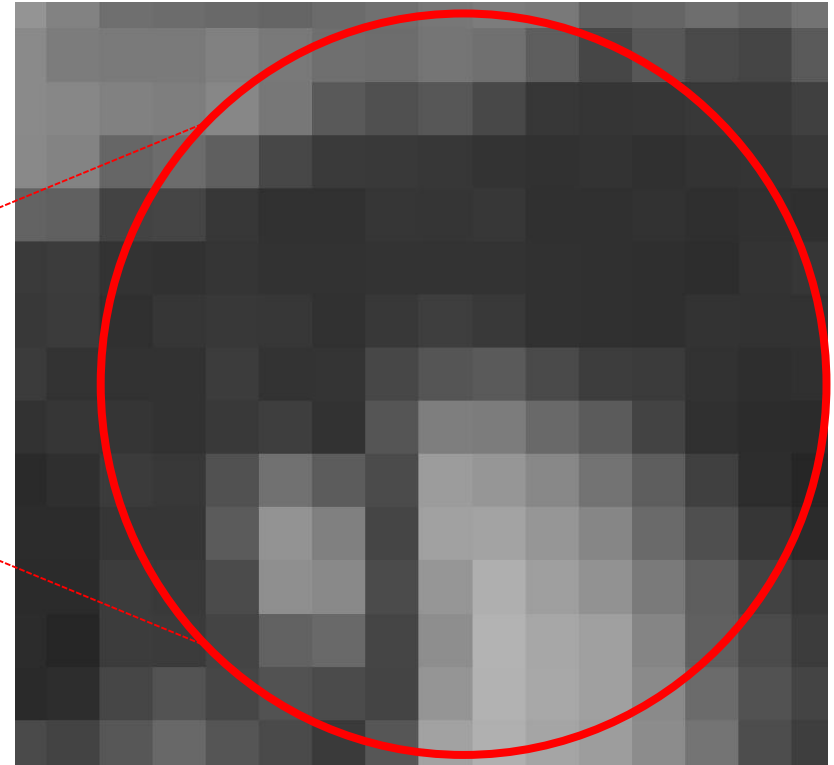


- 512 x 512 (pixel by pixel size)
- 밝기 값만 가지고 있는 이미지
- 8 bits range

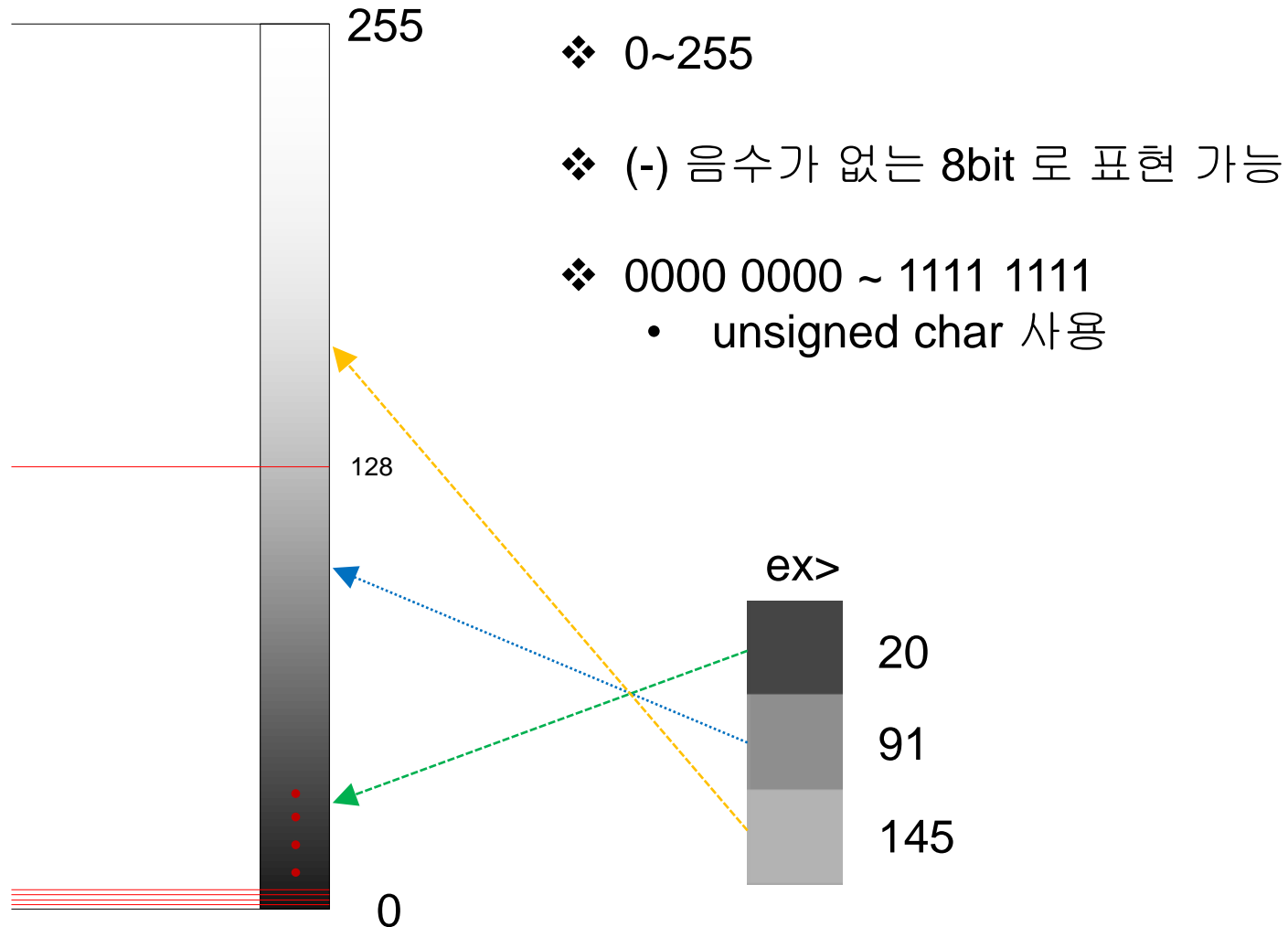


512x512 = 26112개 화소

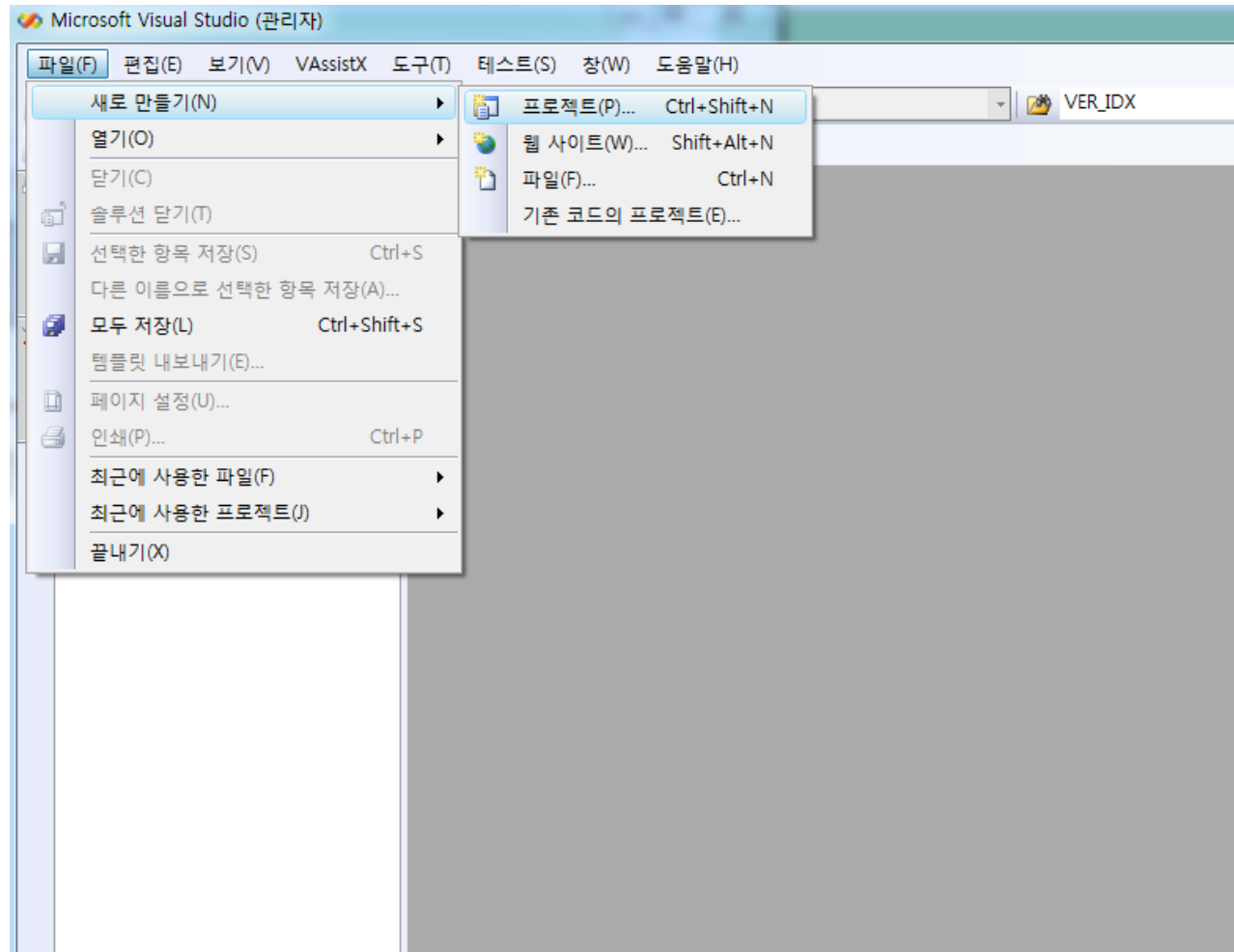
Lana 이미지 확대

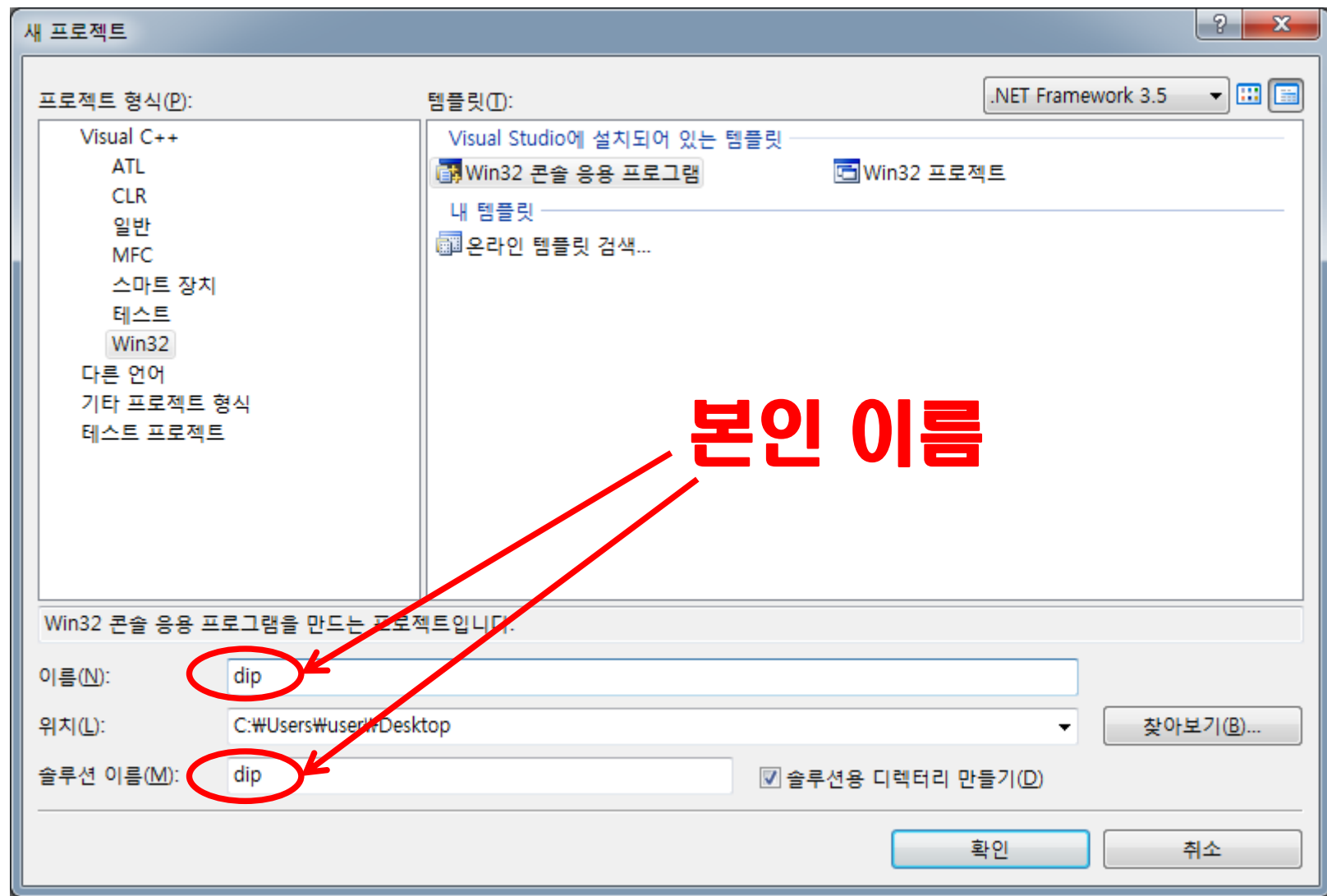


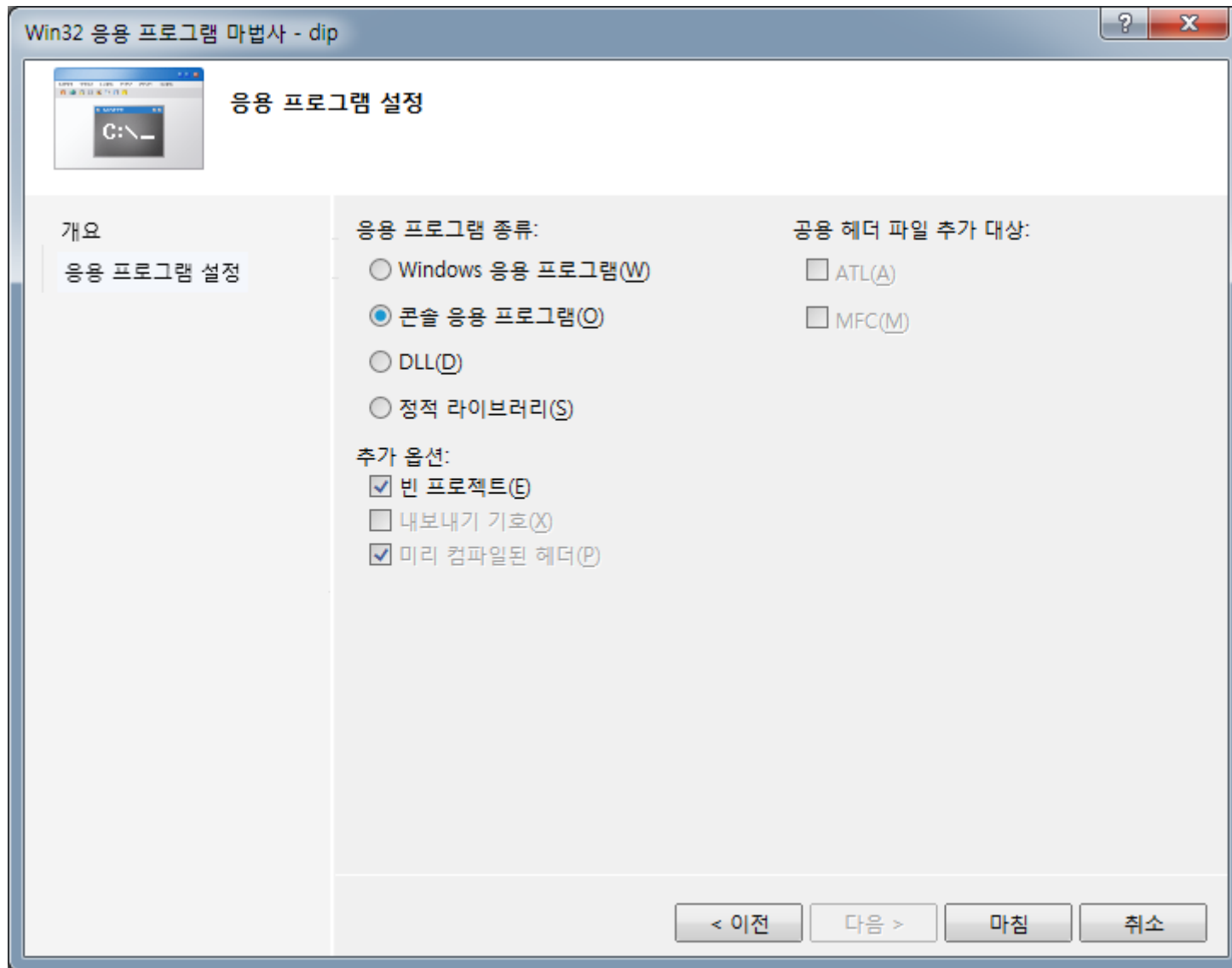
Lena 밝기 표현 방법

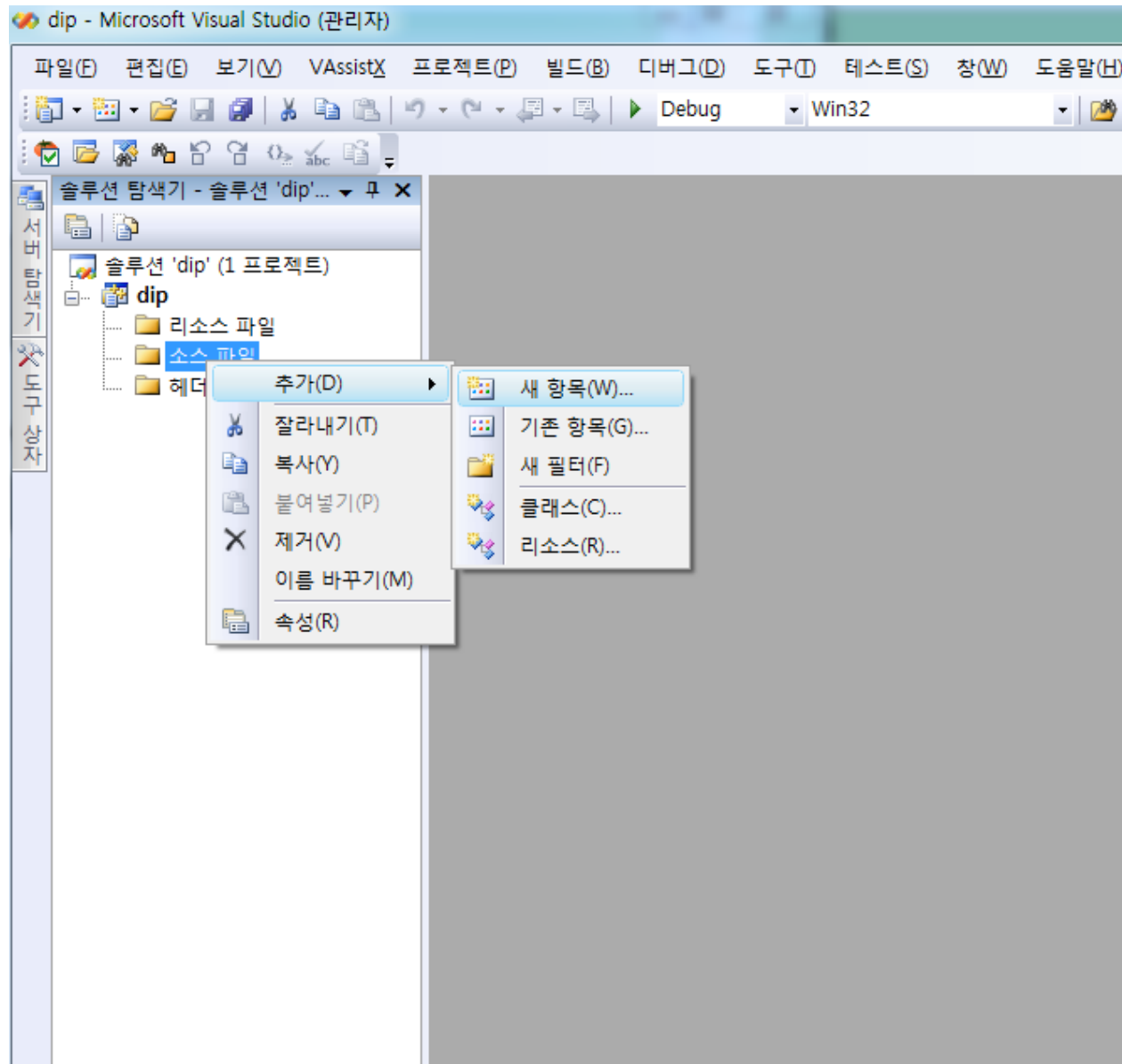


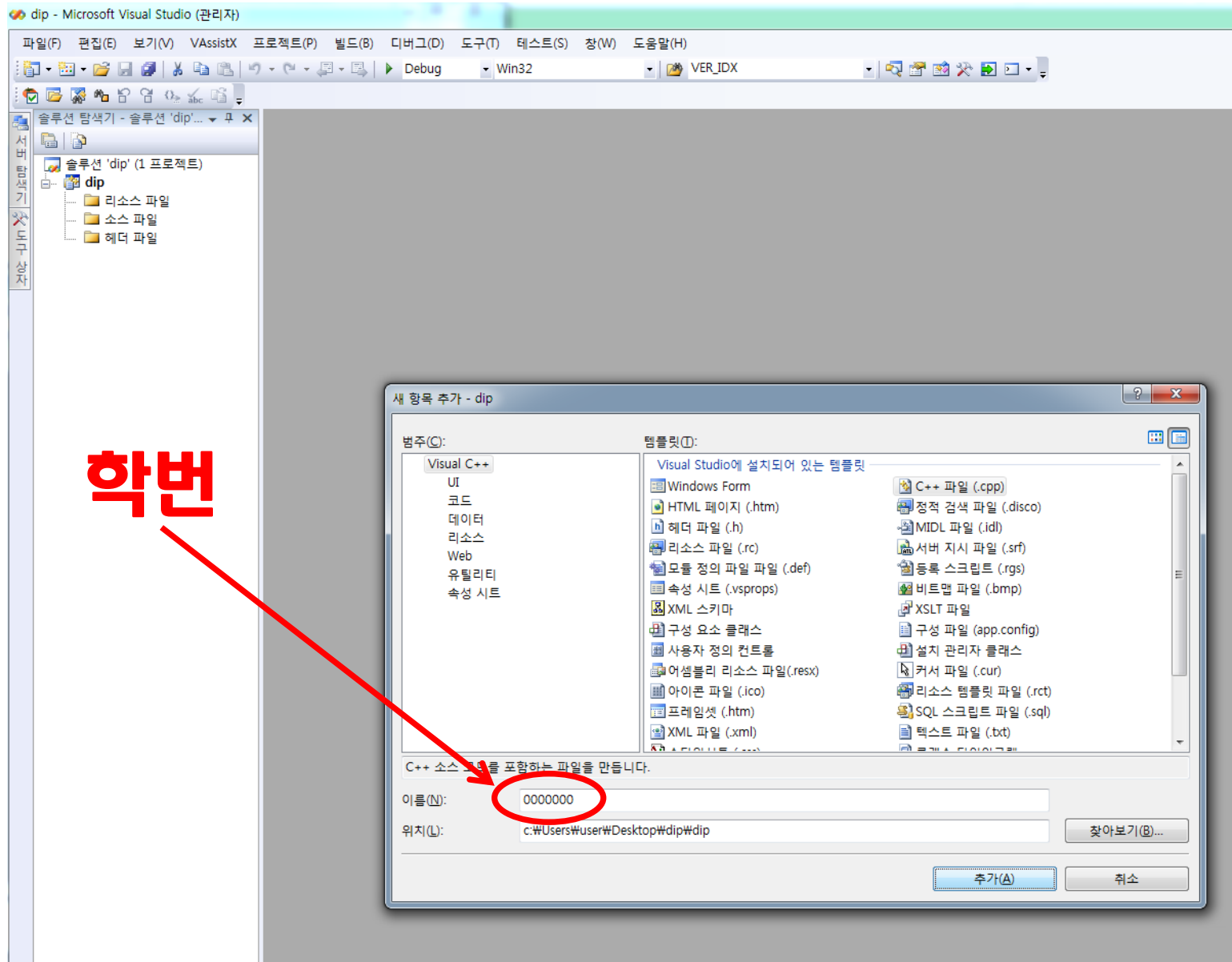
프로젝트 생성 과정 (2010 ver.)

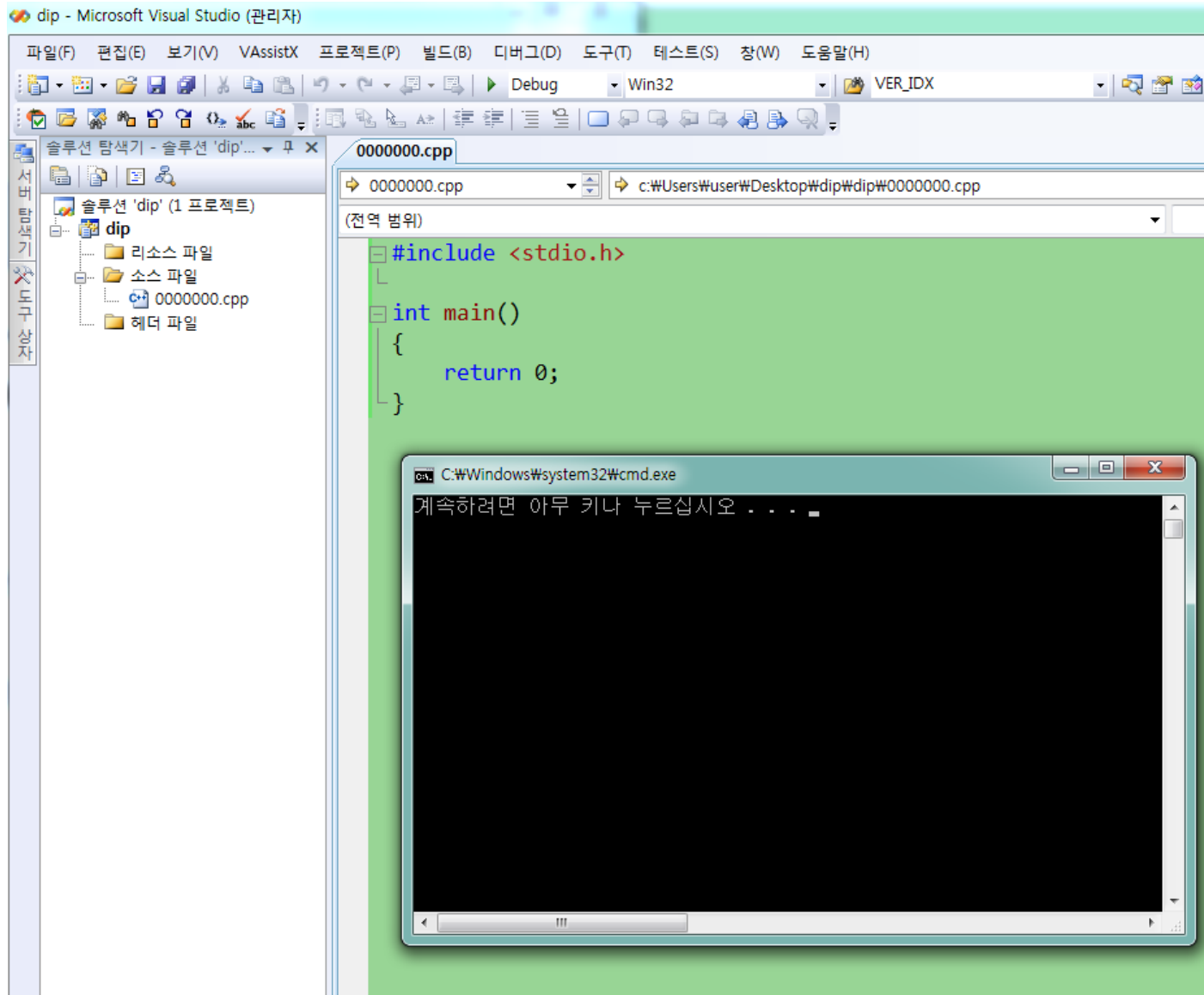




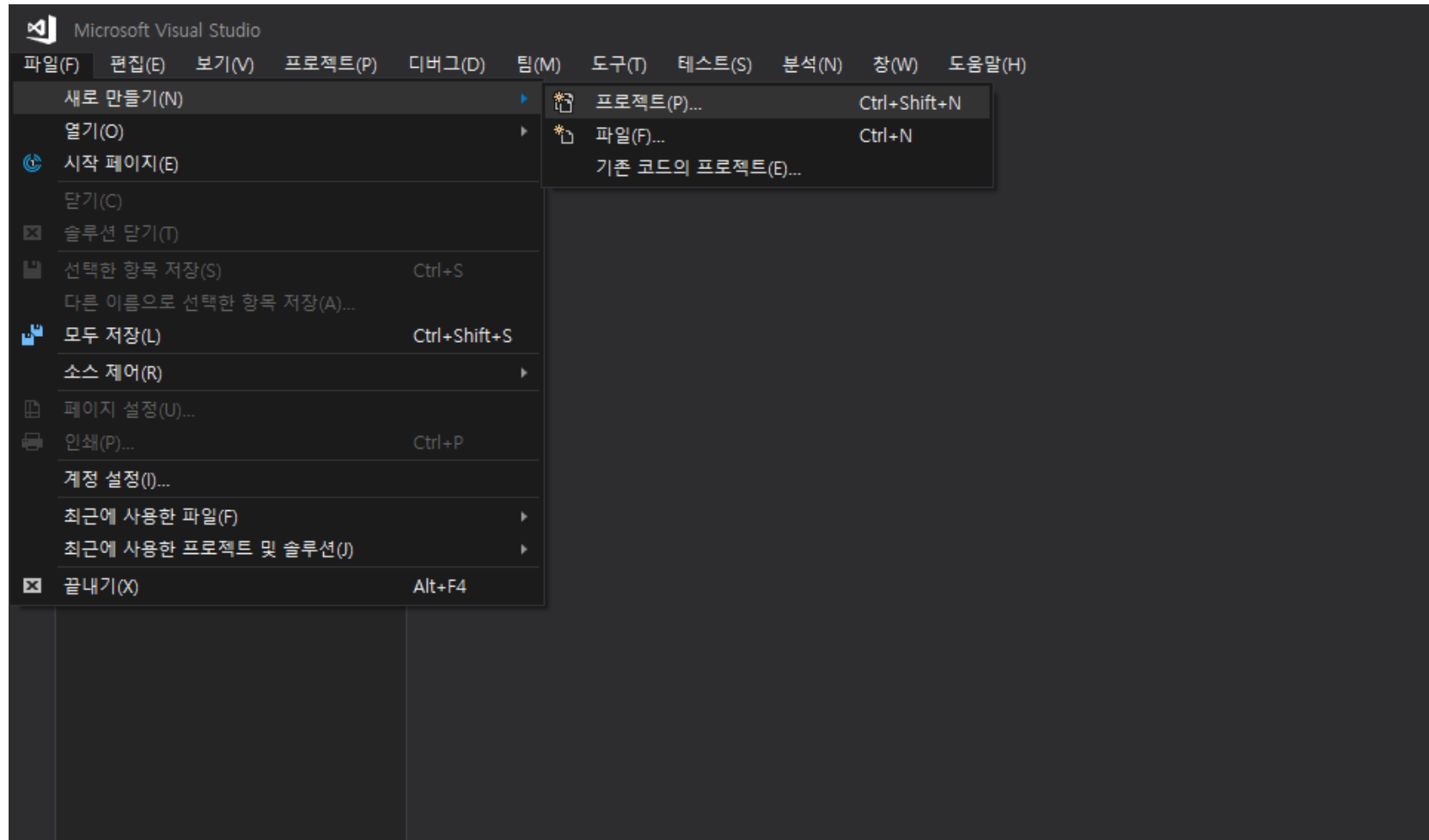


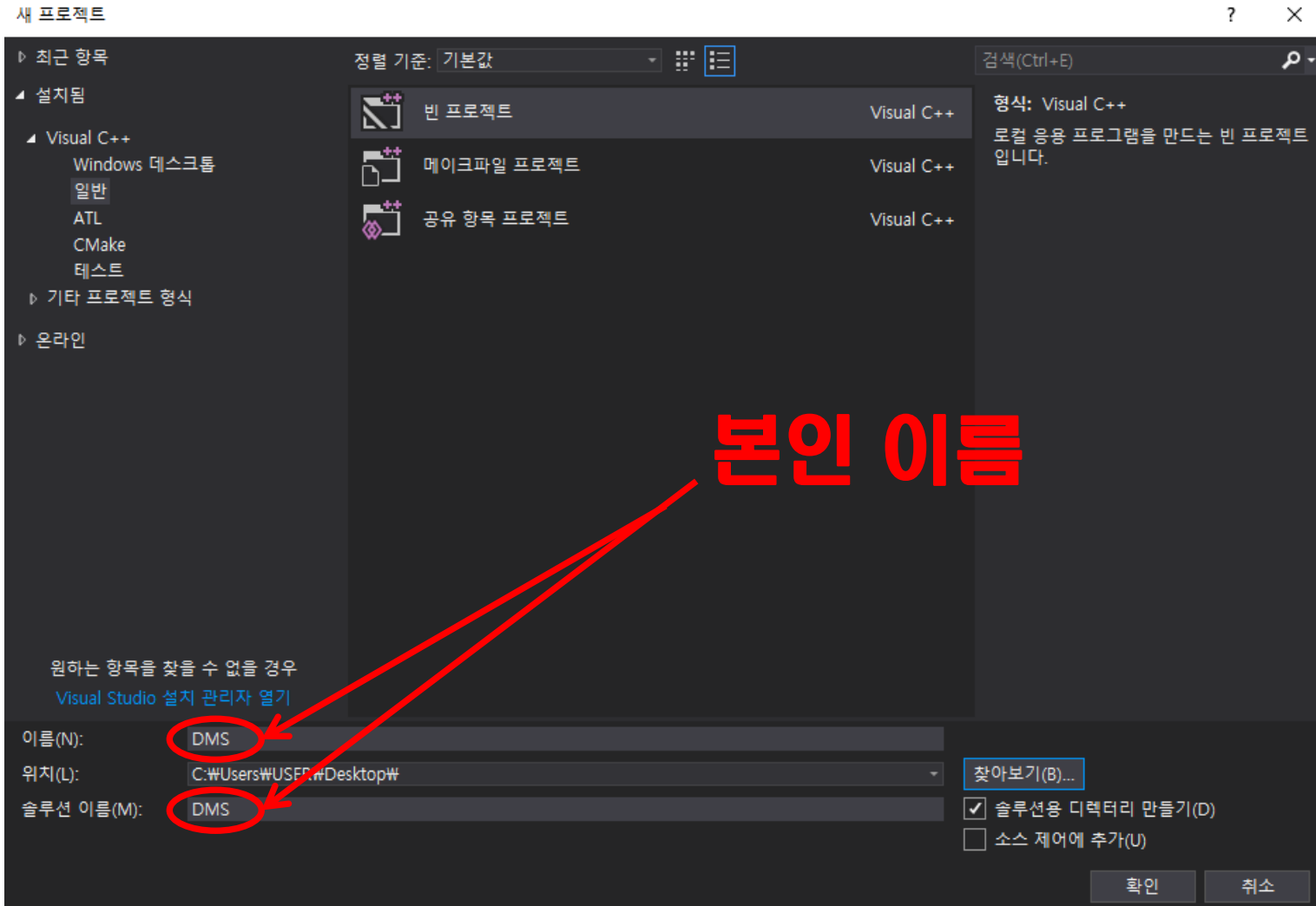


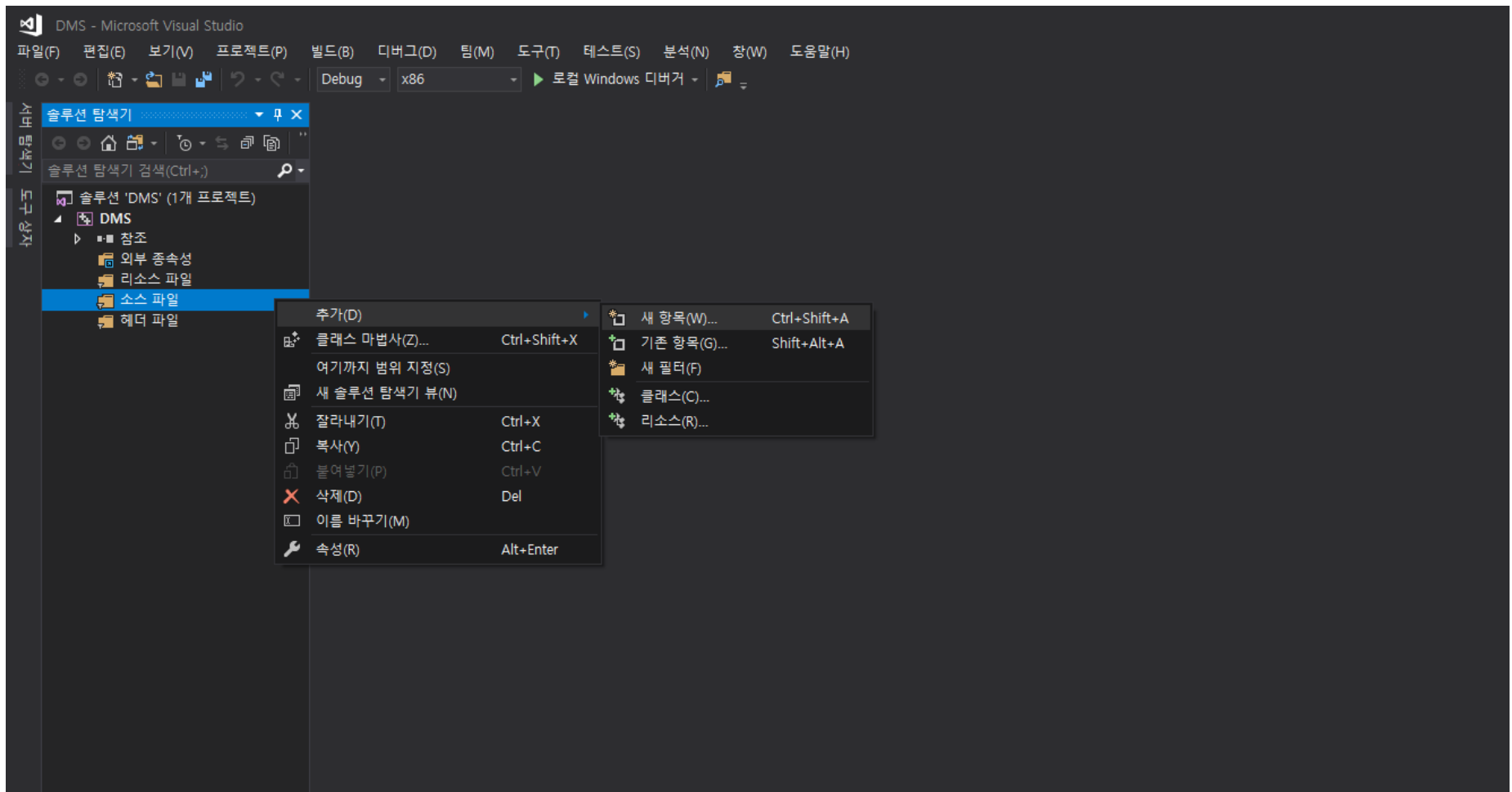


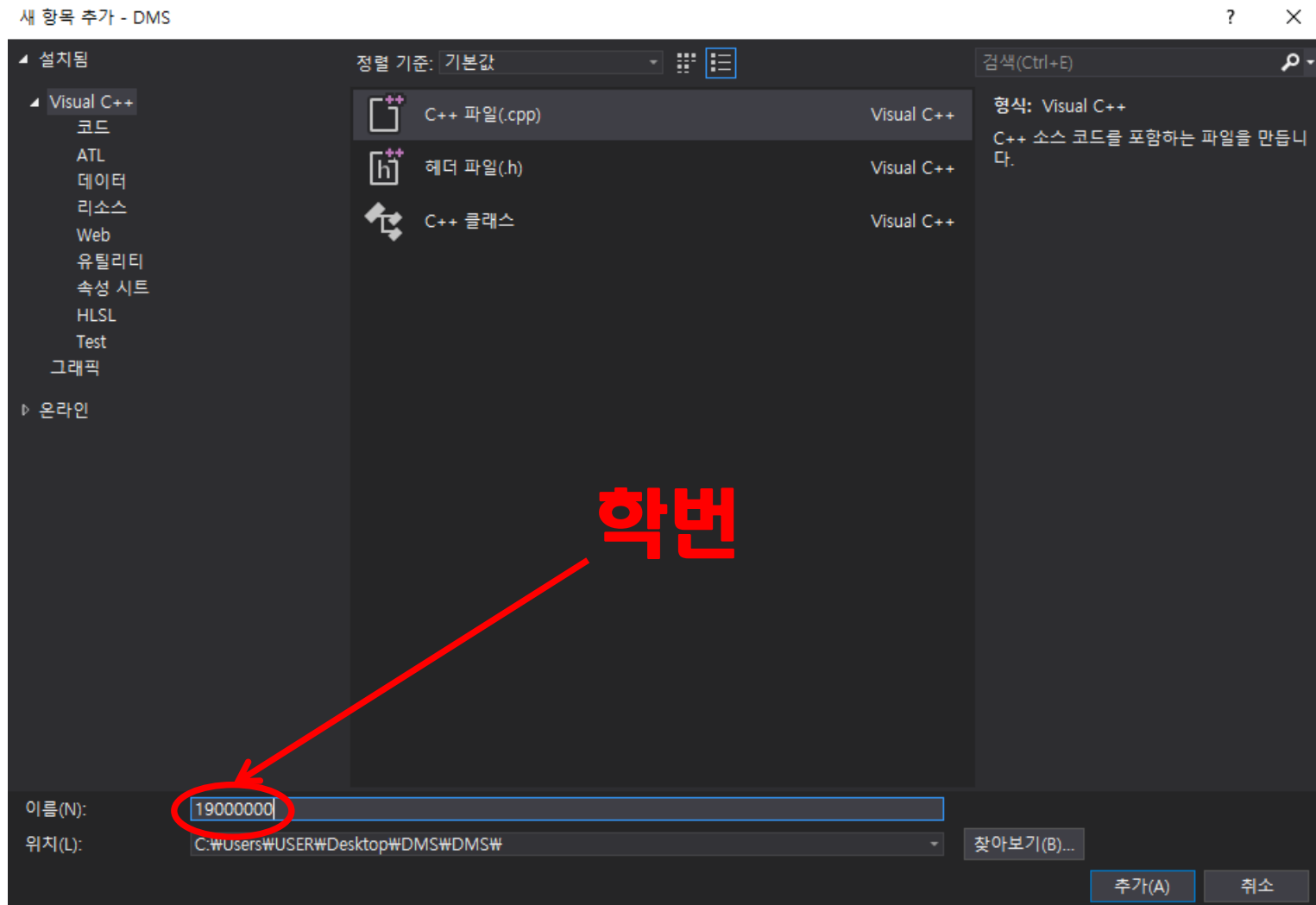


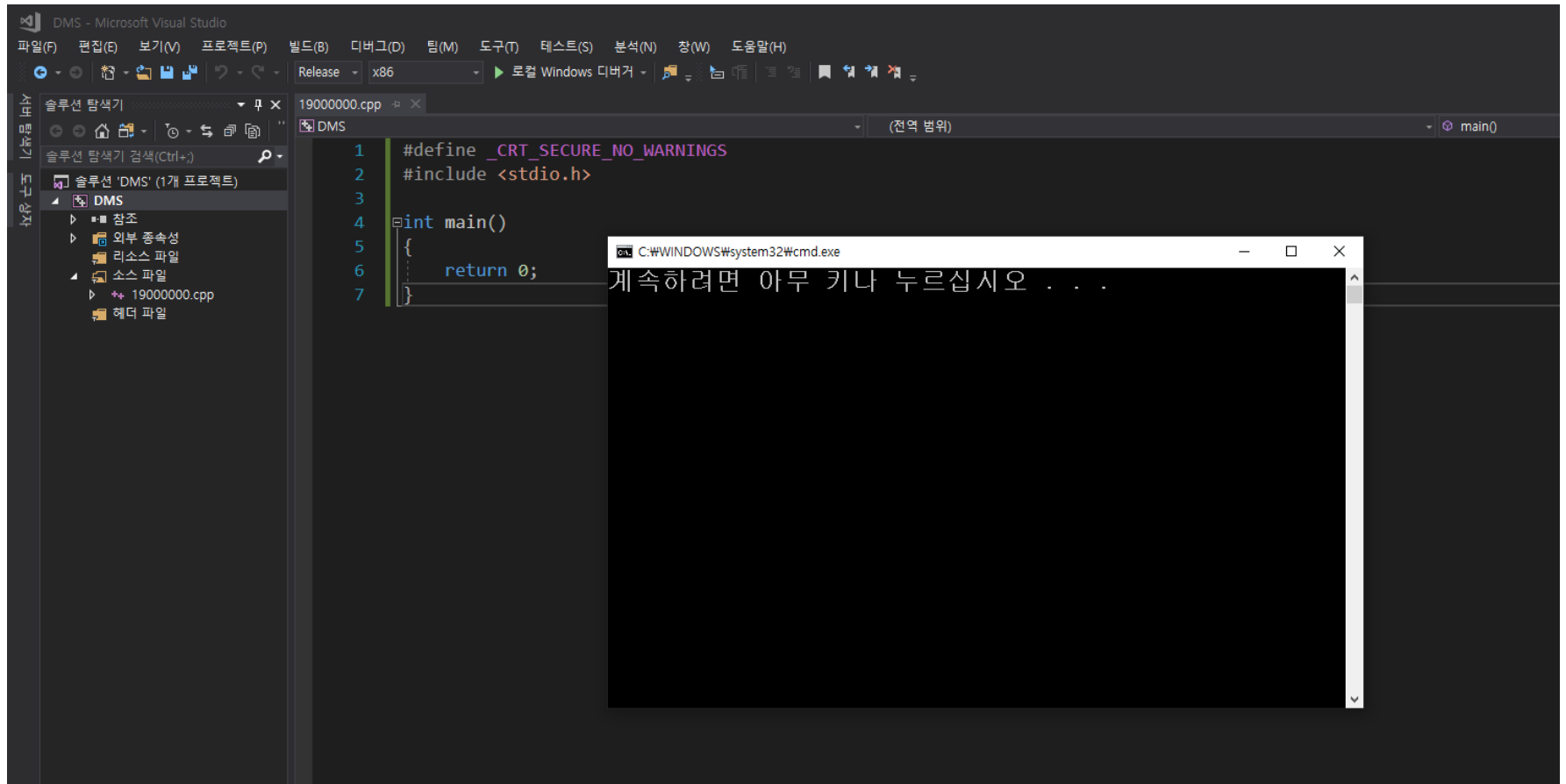
프로젝트 생성 과정 (2017 ver.)











Function

- 메모리 동적 할당 및 해제
 - C : malloc(), free();
 - C++ : new, delete;
- 파일 입출력
 - 파일 열기/생성 : fopen()
 - 파일 닫기 : fclose()
 - 파일 읽기 : fread()
 - 파일에 데이터 쓰기 : fwrite()

File I/O function (ANSI C)

- FILE *fopen(const char *filename, const char *mode);

ex) #include<stdio.h>

FILE *fp;

fp=fopen("lena.img","rb");

.....

fclose(fp);

fp=fopen("out.img","wb");

.....

fclose(fp);

- fread(void *ptr, size_t size, size_t n, FILE *stream);

ex) fread(a,sizeof(unsigned char),512*512,fp);

- fwrite(const void *ptr, size_t size, size_t n, FILE *stream);

ex) fwrite(a,sizeof(unsigned char),512*512,fp);

1-D Memory Allocation (C)

```
#include<stdlib.h>
void *malloc(size_t size);
free(void * block);
```

Ex) unsigned char *a;

a=(unsigned char*)malloc(sizeof(unsigned char)*512*512);

.....

.....

free(a);

1-D memory allocation (C++)

- 예제 코드

```
#include <stdio.h>
int main(void)
{
    // 이미지 읽기 및 메모리(pLenaImg)에 저장
    FILE* hLena = fopen("lena.img", "rb");
    unsigned char *pLenaImg = new unsigned char[512*512];

    fread(pLenaImg, 1, 512*512, hLena);
    fclose(hLena);

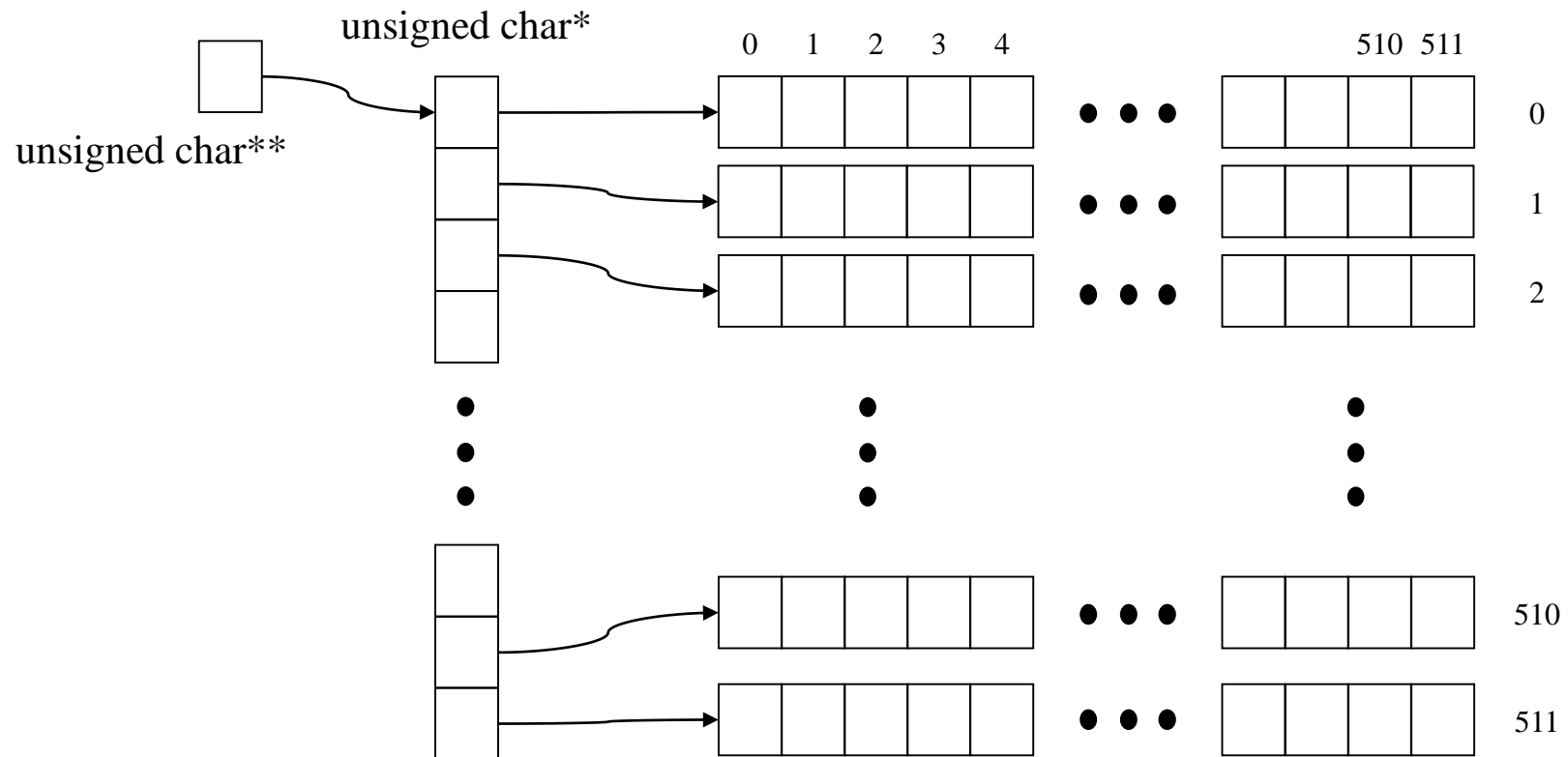
    // 메모리에 저장된 이미지를 이용하여 영상처리 코드 구현
    unsigned char *pOutputImg = new unsigned char[512*512];
    ...

    // 파일에 결과를 출력, 동적할당 해제
    FILE* hOutput = fopen("output.img", "wb");
    fwrite(pOutputImg, 1, 512*512, hOutput);
    fclose(hOutput);
    delete[] pLenaImg;
    delete[] pOutputImg;

    return 0;
}
```

Basic 2-D Memory allocation

a[512][512]



Basic 2D Memory Allocation

2-Dimensional malloc() and image read

```
#include<stdio.h>
```

```
#include<math.h>
```

```
ex) unsigned char **a; //since image is 2D data
```

```
a=(unsigned char**)malloc(sizeof(unsigned char*)*512); //Height
```

```
for(i=0;i<512;i++)
```

```
    a[i]=(unsigned char*)malloc(sizeof(unsigned char)*512);
```

```
fp=fopen("lena.img","rb");
```

```
for(i=0;i<512;i++)
```

```
    fread(a[i],sizeof(unsigned char),512,fp);
```

```
// Do operation like Convolution, DFT,.....
```

```
.....
```

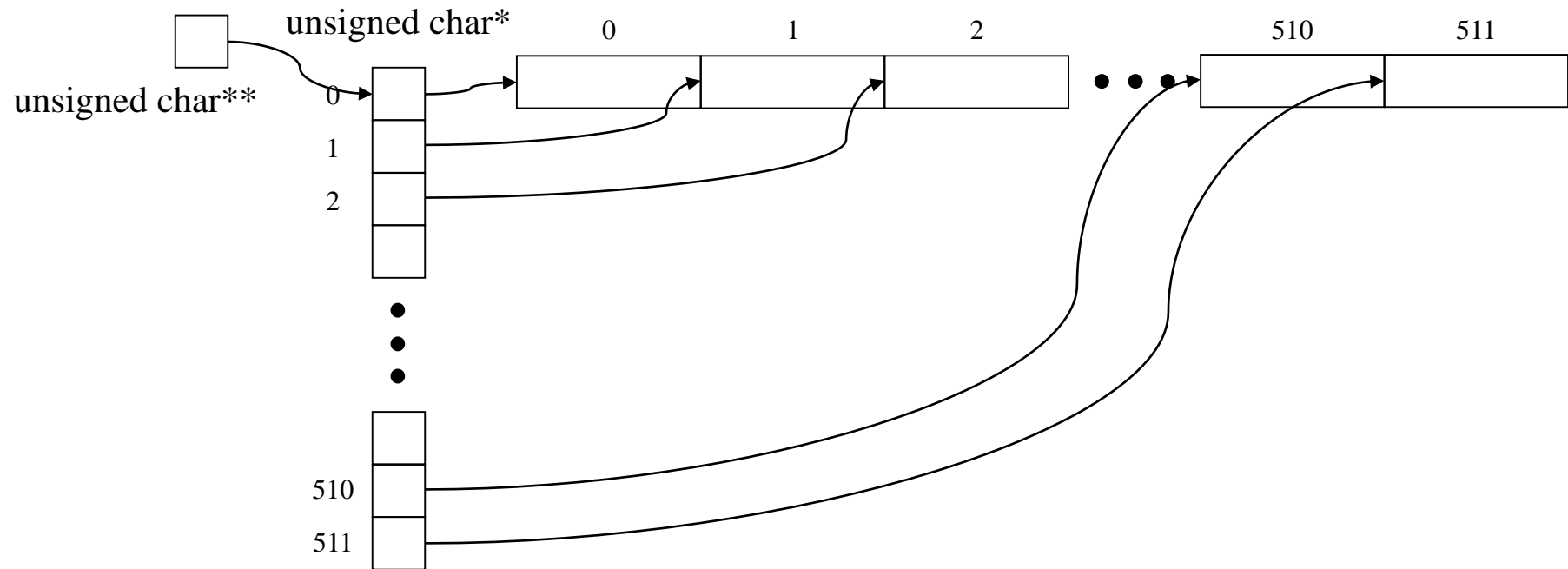
```
for(i=0;i<512;i++)
```

```
    free(a[i]);
```

```
free(a);
```

C

Advanced 2-D memory allocation



Advanced 2-D memory allocation

- 2차원 배열 할당 예제 (<http://codeng.tistory.com/8> 참조)

```
unsigned char** _2dAlloc(int width, int height)
{
    int i;
    unsigned char** ppA = new unsigned char*[height];
    ppA[0] = new unsigned char[width*height];
    for(i=1;i<height;i++) ppA[i] = ppA[i-1]+width;

    return ppA;
}
```

C

```
unsigned char** memory_alloc2D(int height, int width)
{
    unsigned char** ppMem2D = 0;
    int j, i;

    // array of pointer
    ppMem2D = (unsigned char **)malloc(sizeof(unsigned char *) * height);

    *ppMem2D = (unsigned char *)malloc(sizeof(unsigned char) * (width*height));

    for (j=1; j< height; j++ )
    {
        ppMem2D[j] = ppMem2D[j-1] + width;
    }
    return ppMem2D;
}
```

C++

typedef uint8 unsigned char
typedef uint32 unsigned int

Memory Allocation

- 2차원 배열을 이용한 파일 입출력 예제

```
#include <stdio.h>
int main(void)
{
    FILE* hLena = fopen("lena.img", "rb");
    unsigned char** ppLena = _2dAlloc(512,512);
    fread(ppLena[0], 1, 512*512, hLena);

    ...

    fclose(hLena);
    delete[] ppLena[0];
    delete[] ppLena;
    return 0;
}
```

C++

example

```
uint8** memory_alloc2D(uint32 height, uint32 width)
```

```
{
    uint8**    ppMem2D = 0;
    uint32     j, i;

    // array of pointer
    ppMem2D = (uint8**)calloc(sizeof(uint8*), height);
    if ( ppMem2D == 0 )
    {
        return 0;
    }

    *ppMem2D = (uint8*)calloc(sizeof(uint8), width * height );
    if( (*ppMem2D) == 0 )
    {
        // free the memory of array of pointer
        free( ppMem2D );
        return 0;
    }

    for (j=1; j< height; j++ )
    {
        ppMem2D[j] = ppMem2D[j-1] + width;
    }

    return ppMem2D;
}
```

```
typedef    uint8    unsigned char
typedef    uint32   unsigned int
```

C

```
int memory_free2D(uint8** ppMemAllocated)
{
    if (ppMemAllocated == 0 )
    {
        return -1;
    }

    free( ppMemAllocated[0] );
    free( ppMemAllocated );

    return 0;
}
```

Image read/write function (ANSI C)

```
int main(void)
{
    FILE*      fpInputImage = 0;
    FILE*      fpOutputImage = 0;
    uint8**    ppInputImageBuffer= 0;

    // input file open
    fpInputImage = fopen(IMG_NAME, "rb");

    // memory allocaiton
    ppInputImageBuffer = memory_alloc2D(IMG_HEIGHT, IMG_WIDTH);

    // input file read to memory from the file
    fread( &ppInputImageBuffer[0][0], sizeof(uint8), IMG_WIDTH*IMG_HEIGHT, fpInputImage);

    // output fileopen
    fpOutputImage = fopen("result.raw", "wb");

    // write the file
    fwrite( &ppInputImageBuffer[0][0], sizeof(uint8), IMG_WIDTH*IMG_HEIGHT, fpOutputImage);

    memory_free2D(ppInputImageBuffer);
    fclose(fpInputImage);
    fclose(fpOutputImage);

    return 0;
}
```

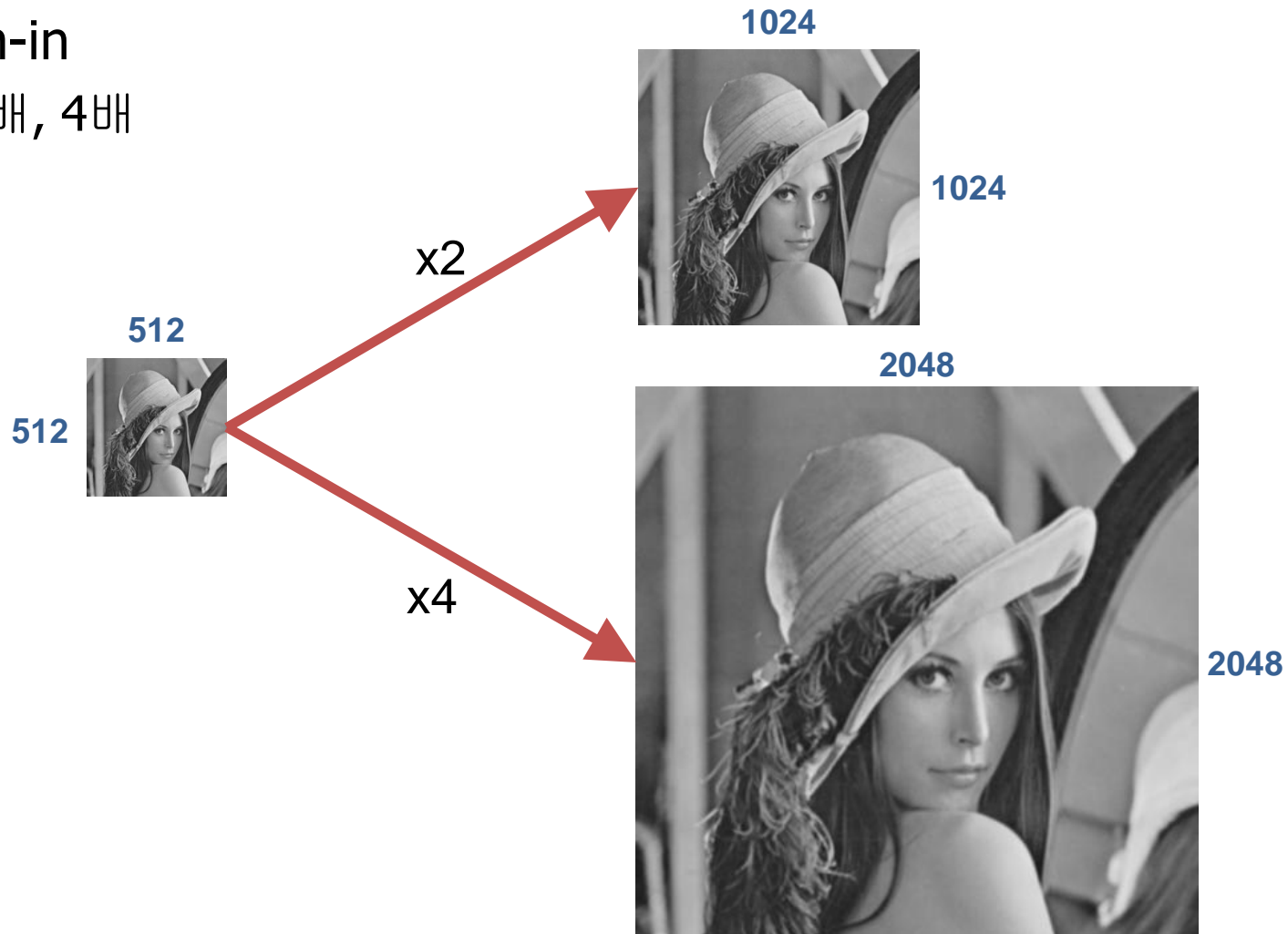
1. Zoom-out

- Zoom-out
 - $1/2$ 배, $1/4$ 배



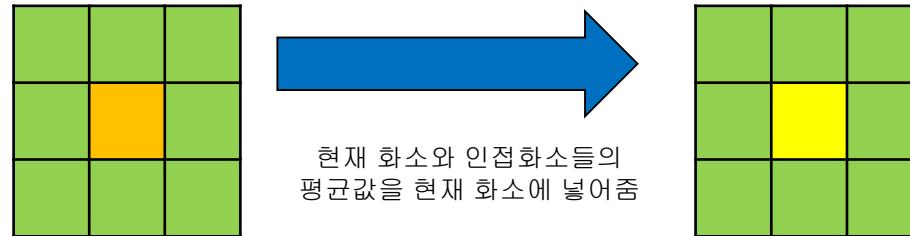
2. Zoom-in

- Zoom-in
 - 2배, 4배



3. Low-pass filtering

- Average filter 이용



■ : 현재 화소

■ : 인접 화소

■ : 필터링 된 화소



4. High-pass filtering

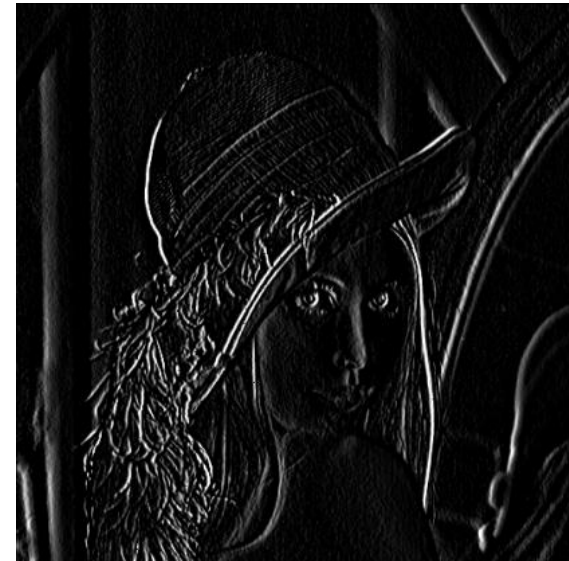
- Sobel operator 이용

+1	+2	+1
0	0	0
-1	-2	-1



< 수 평 >

+1	0	-1
+2	0	-2
+1	0	-1



< 수 직 >





여러 가지 **High-pass filtering**의 예

Roberts
operator

-1	0	0
0	1	0
0	0	0

0	0	-1
0	1	0
0	0	0

Prewitt
operator

-1	-1	-1
0	0	0
1	1	1

-1	0	1
-1	0	1
-1	0	1

Sobel
operator

-1	-2	-1
0	0	0
1	2	1

-1	0	1
-2	0	2
-1	0	1

Frei-chen
operator

-1	$-\sqrt{2}$	-1
0	0	0
1	$\sqrt{2}$	1

-1	0	1
$-\sqrt{2}$	0	$\sqrt{2}$
-1	0	1

5. Zoom-out and translation

- Zoom-out and translation
 - Lena 영상 크기 256 x 256
 - 검정 배경 크기 512x512

$$x' = x + dx$$

$$y' = y + dy$$



(50,50)



(100,80)

6. Zoom-out and translation and rotation

- Translation - (200,20)

$$x' = x + dx$$

$$y' = y + dy$$

- Rotation - ($\theta = 0.2, 0.5$): radian

$$x'' = x' * \cos(\theta) - y' * \sin(\theta)$$

$$y'' = x' * \sin(\theta) + y' * \cos(\theta)$$

6. Zoom-out and translation and rotation

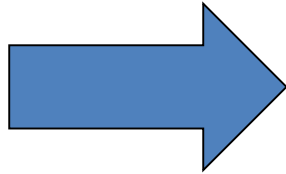
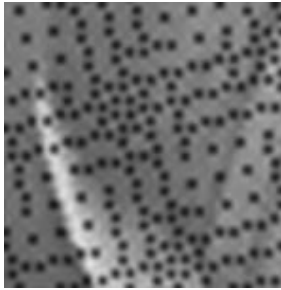


(200,20)
 $\theta = 0.2$



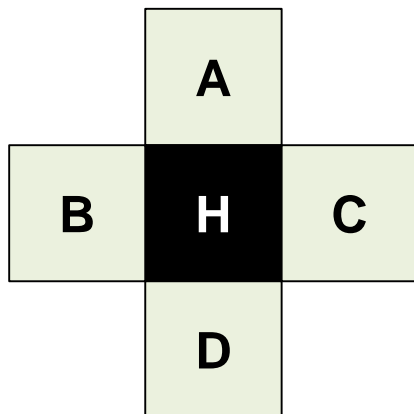
(200,20)
 $\theta = 0.5$

Hole 채우기



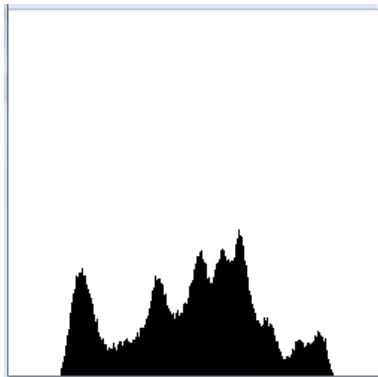
ex) 홀(Hole) 채우기

$$H = (A+B+C+D)/4$$

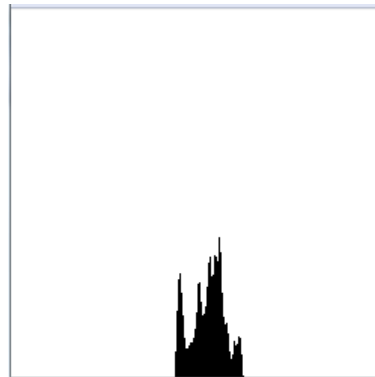
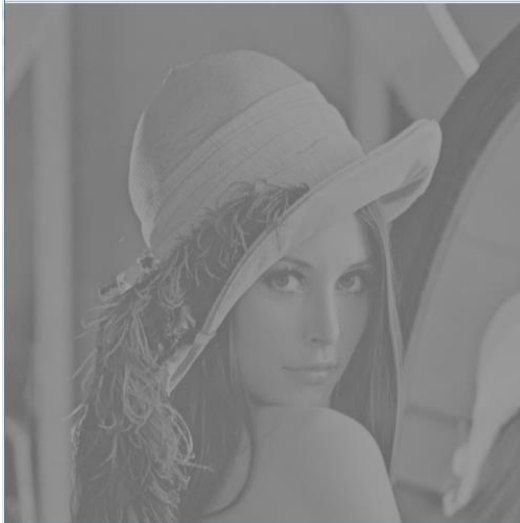


7. Histogram Slide-mapping & Stretch-mapping

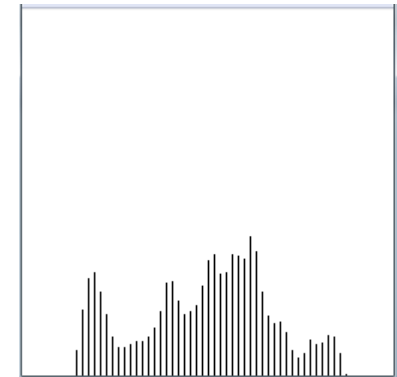
Lena



Slide mapping

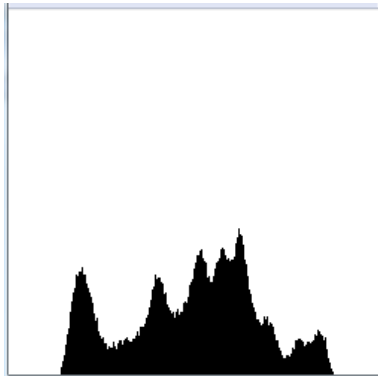


Stretch mapping



7. Histogram Equalization

Lena



주의사항

- Image Boundary
 - Don't touch
 - Image filtering
 - 이미지 파일 출력 시 pixel range 를 0~255로 맞춰줘야 함.
 - 절대값
 - ✓ DPCM, Closing-Opening 등
 - clipping
 - ✓ Image filtering 등
- ```
if(img[i][j] < 0)
 clip = 0;
else if(img[i][j] > 255)
 clip = 255;
else
 clip = img[i][j];
```