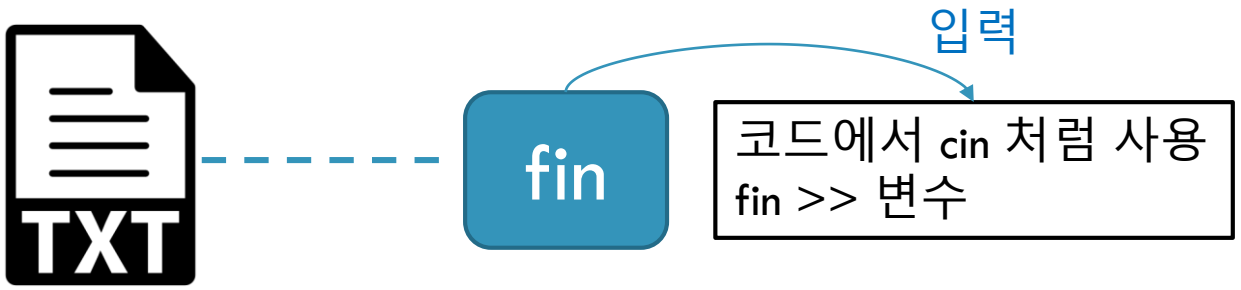
파일에 읽고 쓰기

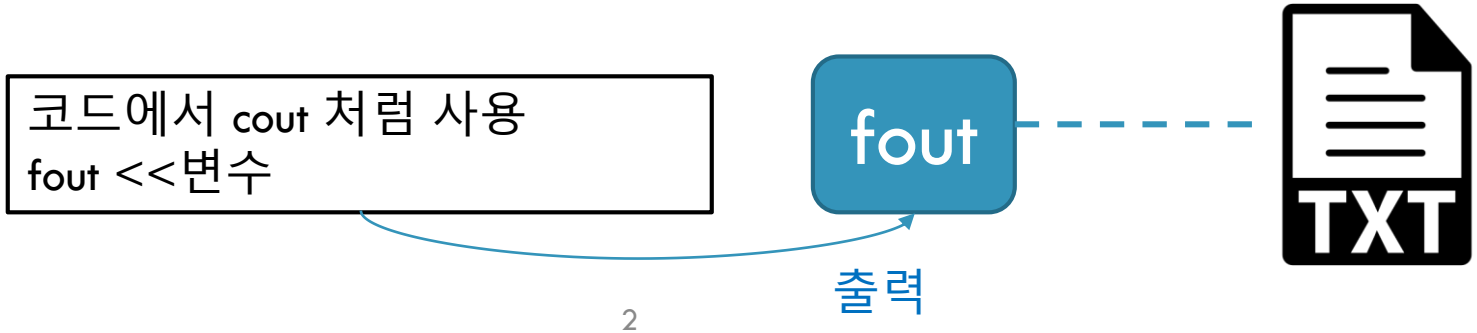
전처리문

이미지 처리 기본

- cin : 키보드와 미리 연결되어 있음 (istream 사용)
- cout: 모니터와 미리 연결되어 있음 (ostream 사용)
- 미리 정의된 변수 cin, cout // 내가 변수를 새로 생성할 필요가 없었음

- 파일을 읽고 쓰는 건 cout과 cin을 사용하는 것과 유사 .
  단, 우리가 원하는 파일로 연결된 변수 선언이 필요
  - **ifstream**: input file stream을 줄인 표현
  - **ofstream**: output file stream을 줄인 표현





파일 읽기 – ifstream 타입

```
#include <fstream>
#include <iostream>

using namespace std;
int main ()
{
      ifstream file_reader( "myfile.txt" );
      if ( ! file_reader.is_open() )
      {
            cout << "Could not open file!" << '\n';
      }
      int number;
      // 여기서 정수를 제대로 읽어 들이는지 아닌지를 확인한다.
      if ( file_reader >> number )
      {
            cout << "The value is: " << number;
      }
}
```

//파일 입력 스트림 변수 선언

현재 디렉토리에서 myfile.txt 찾음
전체 경로를 지정해도 됨(c:\myfile.txt)

cin 을 쓰듯 사용. 파일에서 정수 하나를 읽어들임

파일 읽기 – ifstream 타입

```
#include <fstream>
#include <iostream>
#include <vector>

using namespace std;

int main ()
{
        ifstream file_reader( "myfile.txt" );
        if ( ! file_reader.is_open() )
        {
                cout << "Could not open file!" << '\n';
        }
        while(1){
                int number;
                if ( ! ( file_reader >> number ) )
                {
                        break;
                }

               cout<<"The value is : " << number <<endl;
        }
}
```

//파일 입력 스트림 변수 선언

정수를 계속계속 읽어들임
파일의 끝(EOF)까지
//false로 인식

파일 쓰기 – ofstream 타입

```
#include <fstream>
#include <iostream>
#include <cstdlib>

using namespace std;

int main ()
{                    //파일 출력 스트림 변수 선언
      ofstream file_writer( "highscores.txt" );
      if ( ! file_writer.is_open() )
      {
            cout << "Could not open file!" << '\n';
            return 0;
      }

      // 실제 점수는 아직 없으므로 일단 10에서 1까지 숫자를 출력한다.
      for ( int i = 0; i < 10; i++ )
      {
            file_writer << 10 – i << '\n';
      }
}
```

cout 을 쓰듯 사용

파일에 숫자 쓰기

파일에 읽고 쓰기

전처리문

이미지 처리 기본

- 전처리문
  - 실제 컴파일 전에 미리 처리되는 문장
  - 기존의 방대한 소스를 건드리지 않은 상태에서 부분적인 컴파일 수행
  - 소스의 시작부분에 위치하며, #으로 시작
  - #include, #define, #ifdef, #undef 등

- #define 문
  - 소스 코드에 사용할 숫자나 문자열, 함수의 이름이 너무 길거나 복잡할 때 한 눈에 파악하도록 쉬운 기호로 표현함

```
#define [기호] [숫자 또는 문자열 또는 함수]
```

  - 예) 원주율 표시 - 3.1415926535

```
void main( ){
…
    area = rad * rad * 3.1415926535;
…
}
```

➡

```
#define PI 3.1415926535
void main( ){
…
    area = rad * rad * PI;
…
}
```

전처리 과정

#include나 #define으로 정의된 내용을 main 함수 내부의 값으로 바꿔주는 작업을 '전처리(preprocessing)'라고 하며, 이 작업은 컴파일 전에 수행된다.

파일에 읽고 쓰기

전처리문

이미지 처리 기본

‘가공되지 않은’ 파일 형식

Raw Image Format

"JPEG 파일 형식"

사진 파일의 대표적인 확장자는 jpg입니다
JPEG는 높은 압축률과 함께 뛰어난 화질로
카메라에서 많이 사용되는 파일 형식입니다

용량을 줄이기 위해 압축 과정을 거치므로
적은 용량으로 많은 사진을 저장할 수 있지만
사진의 화질이 저하된다는 단점이 있습니다

카메라를 구입하고 별도의 설정을 하지 않으면
기본으로 JPEG 형식이 설정되어 있을 것입니다
JPEG 형식도 별도로 Fine, Normal, Basic 등
압축 비율에 따른 화질 설정이 가능합니다

"Raw 파일 형식"

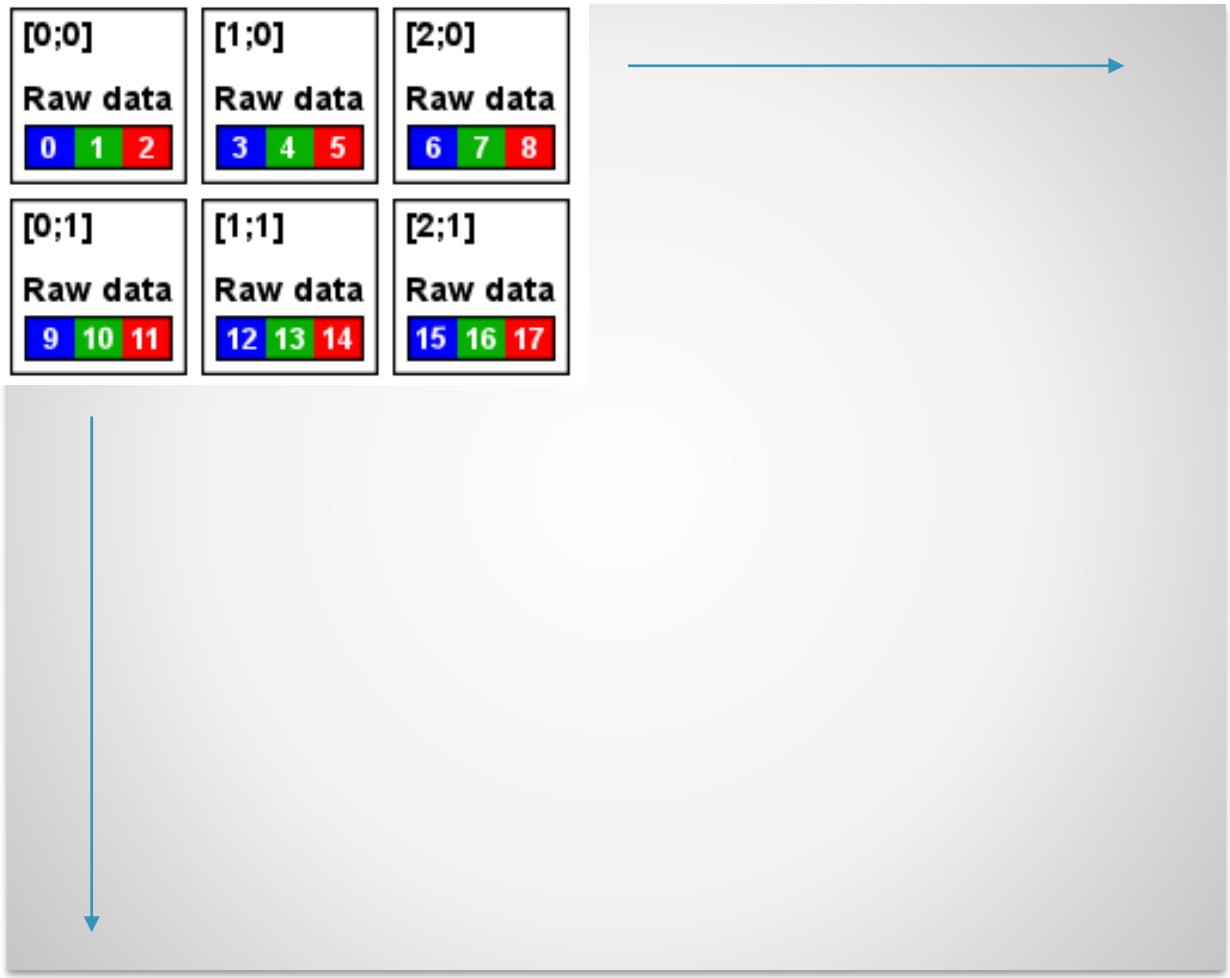
한편, Raw 파일은 뜻 그대로 '날 것'의 의미로
최소한으로 가공된 데이터만 포함하기 때문에
편집 및 출력할 준비가 되지 않은 상태입니다

필름의 경우 네거티브 상태로 존재하기 때문에
사진으로 바로 활용할 수 없는 점에 비유하여
Raw 파일을 디지털 네거티브라고 부릅니다

Raw 파일의 이미지는 최종 변환된 이미지보다
더 넓은 다이내믹 레인지, 색 공간을 가지고
촬영된 이미지의 거의 모든 정보를 보존합니다

이러한 Raw 파일의 형식은 표준이 있지만
CR2 (캐논), NEF (니콘), ARW (소니) 등
제조사에 따라 다른 파일 형식을 사용합니다

http://kong-story.tistory.com/74



칼라
1 픽셀: 8x3 = 24 bit

그레이
1 픽셀: 8x1 = 8 bit

EX1 define 사용하기

```
01 #include <stdio.h>
02
03 #define PI 3.1415926535                          ---상수를 PI로 정의한다.
04 #define STR "원의 면적을 계산했습니다.₩n"          ---문자열을 STR로 정의한다.
05 #define END_MSG printf("프로그램이 종료되었습니다. ₩n₩n")   ---함수를 END_MSG로 정의한다.
06
07 void main( )
08 {
09      printf("반지름이 10인 원의 면적은 = = > %10.5f ₩n", 10*10*PI);   ---정의한 PI를 사용한다.
10
11      printf(STR);                                 ---정의한 STR을 사용한다.
12
13      END_MSG;                                     ---정의한 END_MSG를 사용한다.
14 }
```

EX2 파일 복사하기 (input.txt → output.txt)

```
#include <iostream>
#include <fstream>
 using namespace std;

 int main() {
 ifstream fin("input.txt") ;
 ofstream fout("output.txt");
 char ch;

 while(fin.get(ch))   // char 를 하나 파일로부터 읽기
 fout.put(ch);        // char 를 하나 파일에 쓰기

 fin.close();
 fout.close();
}
```

EX3 파일에서 읽은 것을 모니터에 출력하기

```
1  // Fig. 14.6: Fig14_06.cpp
2  // Reading and printing a sequential file.
3  #include <iostream>
4  #include <fstream> // file stream
5  #include <iomanip>
6  #include <string>
7  #include <cstdlib>
8  using namespace std;
9
10 void outputLine( int, const string &, double ); // prototype
11
12 int main()
13 {
14    // ifstream constructor opens the file
15    ifstream inClientFile( "clients.txt", ios::in );
16
17    // exit program if ifstream could not open file
18    if ( !inClientFile )
19    {
20       cerr << "File could not be opened" << endl;
21       exit( EXIT_FAILURE );
22    } // end if
23
```

exit: 프로그램 종료 시키는 시스템 함수

Fig. 14.6 | Reading and printing a sequential file. (Part 1 of 3.)

```
24      int account; // the account number
25      string name; // the account owner's name
26      double balance; // the account balance
27
28      cout << left << setw( 10 ) << "Account" << setw( 13 )
29         << "Name" << "Balance" << endl << fixed << showpoint;
30
31      // display each record in file
32      while ( inClientFile >> account >> name >> balance )// cin처럼 연속해서 읽기
33         outputLine( account, name, balance );
34   } // end main
35
36   // display single record from file
37   void outputLine( int account, const string &name, double balance )
38   {
39      cout << left << setw( 10 ) << account << setw( 13 ) << name
40         << setw( 7 ) << setprecision( 2 ) << right << balance << endl;
41   } // end function outputLine
```

Fig. 14.6 | Reading and printing a sequential file. (Part 2 of 3.)

```
Account   Name          Balance
100       Jones         24.98
200       Doe           345.67
300       White          0.00
400       Stone        -42.16
500       Rich          224.62
```

Fig. 14.6 | Reading and printing a sequential file. (Part 3 of 3.)

참고 256x256 크기의 raw image 파일 읽기

```
#define MAX_X 256
#define MAX_Y 256

char in_data[MAX_Y][MAX_X];
char ch;

ifstream in("GIRL.gray", ios::in | ios::binary);
ofstream out("GIRL_half.gray", ios::out | ios::binary);

int i, j;
for (i = 0; i < MAX_Y; i++) {
   for (j = 0; j < MAX_X; j++) {
   in.get(ch);
   in_data[i][j] = ch;
  }
}

for (i = 0; i < MAX_Y / 2; i++) {
   for (j = 0; j < MAX_X; j++) {
   out.put(in_data[i][j]);
   }
}
```

1 pixel 8 bit(=1 byte)
char 변수 사용