

제13장 파일처리

1. 스트림의 개념을 이해한다.
2. 객체 지향적인 방법을 사용하여 파일 입출력을 할 수 있다.
3. 텍스트 파일과 이진 파일의 차이점을 이해한다.
4. 순차 파일과 임의 접근 파일의 차이점을 이해한다.

1

이번 장에서 만들어 볼 프로그램



이진 파일을 복사

이미지 파일을 읽어서 화면에 표시



2

스트림(stream)

- 스트림(stream)은 “순서가 있는 데이터의 연속적인 흐름”이다.
- 스트림은 입출력을 물의 흐름처럼 간주하는 것이다.

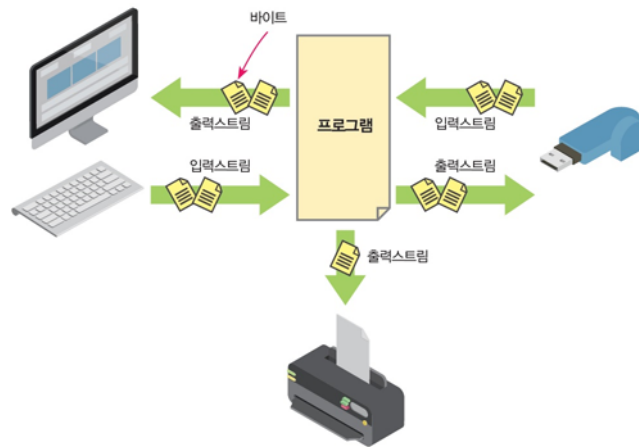


그림 13.1 스트림의 개념

3

입출력 관련 클래스들

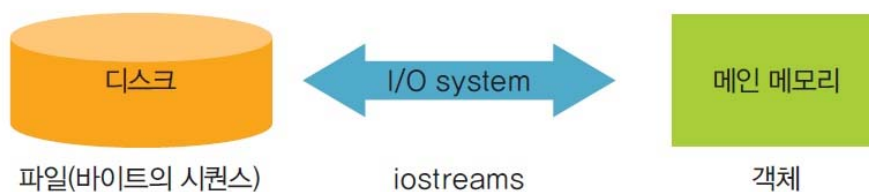


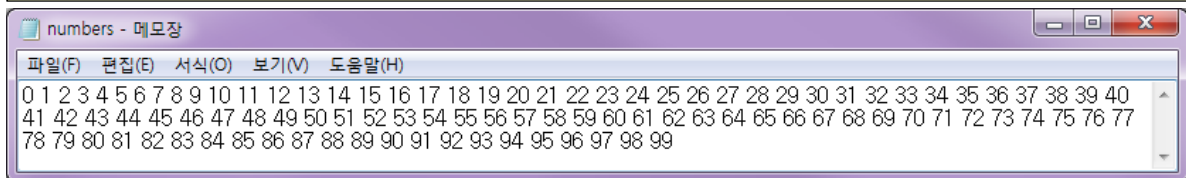
그림 13.2 스트림의 개념

클래스	설명
ofstream	출력 파일 스트림 클래스이다. 출력 파일을 생성하고 파일에 데이터를 쓸 때 사용한다.
ifstream	입력 파일 스트림 클래스이다. 파일에서 데이터를 읽을 때 사용한다.
fstream	일반적인 파일 스트림을 나타낸다.

4

파일쓰기

```
int main()
{
    ofstream os{ "numbers.txt" };
    if (!os) {
        cerr << "파일 오픈에 실패하였습니다" << endl;
        exit(1);
    }
    for(int i=0;i<100; i++)
        os << i << " ";
    return 0;
    // 객체 os가 범위를 벗어나면 ofstream 소멸자가 파일을 닫는다.
}
```



5

파일읽기

```
int main()
{
    ifstream is{ "numbers.txt" };
    if (!is) {
        cerr << "파일 오픈에 실패하였습니다" << endl;
        exit(1);
    }
    int number;
    while (is) {
        is >> number;
        cout << number << " ";
    }
    cout << endl;
    return 0;
    // 객체 is가 범위를 벗어나면 ifstream 소멸자가 파일을 닫는다.
}
```

6

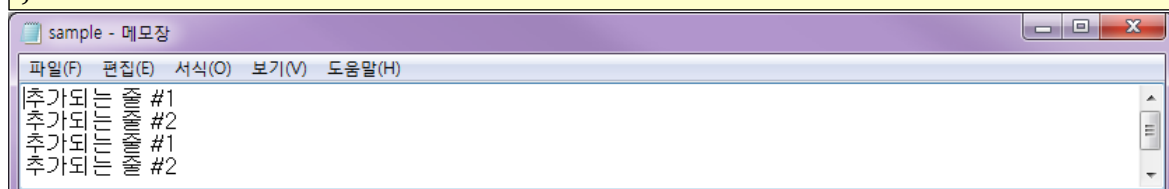
파일모드

ios::in	입력을 위하여 파일을 연다.
ios::out	출력을 위하여 파일을 연다.
ios::binary	이진 파일 입출력을 위하여 파일을 연다.
ios::ate	파일의 끝을 초기 위치로 한다.
ios::app	파일의 끝에 추가된다.
ios::trunc	새로운 내용으로 교체된다.

7

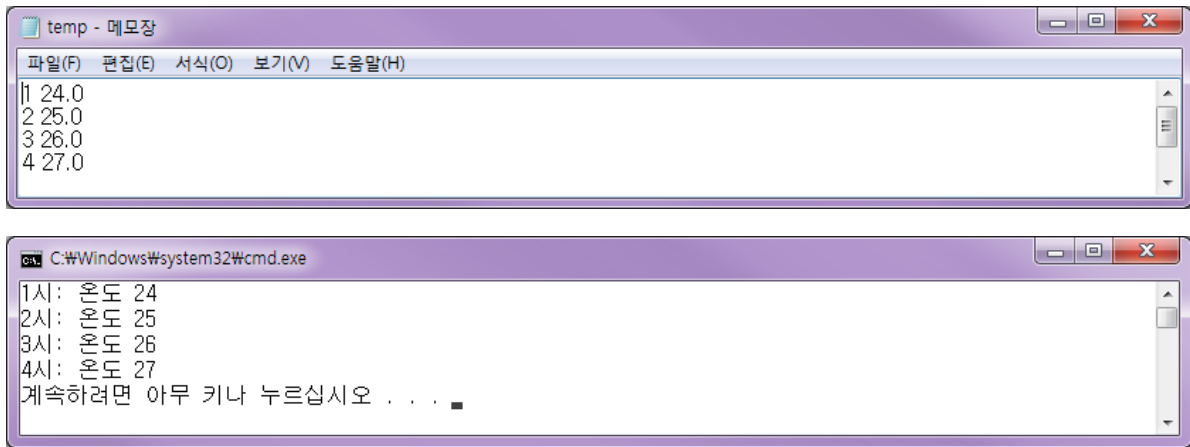
예제

```
int main()
{
    using namespace std;
    ofstream os("sample.txt", ios::app);
    if (!os)
    {
        cerr << "파일 오픈에 실패하였습니다" << endl;
        exit(1);
    }
    os << "추가되는 줄 #1" << endl;
    os << "추가되는 줄 #2" << endl;
    return 0;
}
```



8

Lab: 온도데이터를 처리해보자 #1



9

예제

```
#include <iostream>
#include <fstream> // 파일 입출력을 하려면 헤더 파일을 포함하여야 한다.
using namespace std;
int main()
{
    ifstream is{ "temp.txt" };
    if (!is) { // ! 연산자 오버로딩
        cerr << "파일 오픈에 실패하였습니다" << endl;
        exit(1);
    }
    int hour;
    double temperature;
    while (is >> hour >> temperature) {
        cout << hour << "시: 온도 " << temperature << endl;
    }
    return 0;
}
```

10

Lab: 온도데이터를 처리해보자 #2

- 파일에서 읽은 데이터를 객체로 벡터에 저장했다가 다시 꺼내서 화면에 출력해보자.



11

Solution

```
class TempData {
public:
    int hour;
    double temperature;
};
int main()
{
    ifstream is{ "temp.txt" };
    if (!is) {                // ! 연산자 오버로딩
        cerr << "파일 오픈에 실패하였습니다" << endl;
        exit(1);
    }
    vector<TempData> temps;
```

12

Solution

```
int hour;
double temperature;
while (is >> hour >> temperature) {
    temps.push_back(TempData{ hour, temperature });
}
for ( TempData t : temps) {
    cout << t.hour << "시: 온도 " << t.temperature << endl;
}
return 0;
}
```

13

13.3 멤버함수를 이용한 파일 입출력

□ get(), put()

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

int main()
{
    ifstream is{ "scores.txt" };
    if (!is) {          // ! 연산자 오버로딩
        cerr << "파일 오픈에 실패하였습니다" << endl;
        exit(1);
    }
}
```

14

멤버함수를 이용한 파일 입출력

□ get(), put()

```
char c;
is.get(c);      // 하나의 문자를 읽는다.
while (!is.eof()) // 파일의 끝이 아니면
{
    cout << c;
    is.get(c);
}
cout << endl;
return 0;
}
```

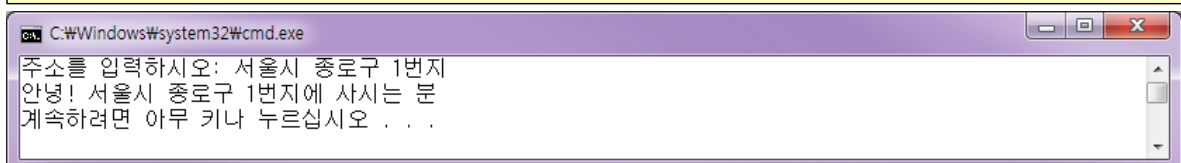


```
C:\Windows\system32\cmd.exe
20180001 홍길동 100
20180002 김유신 90
20180003 강감찬 80
```

15

예제

```
#include <string>
#include <iostream>
using namespace std;
int main() {
    string address;
    cout << "주소를 입력하시오: ";
    getline(cin, address);
    cout << "안녕! " << address << "에 사시는 분" << endl;
    return 0;
}
```



```
C:\Windows\system32\cmd.exe
주소 입력하시오: 서울시 종로구 1번지
안녕! 서울시 종로구 1번지에 사시는 분
계속하려면 아무 키나 누르십시오 . . .
```

16

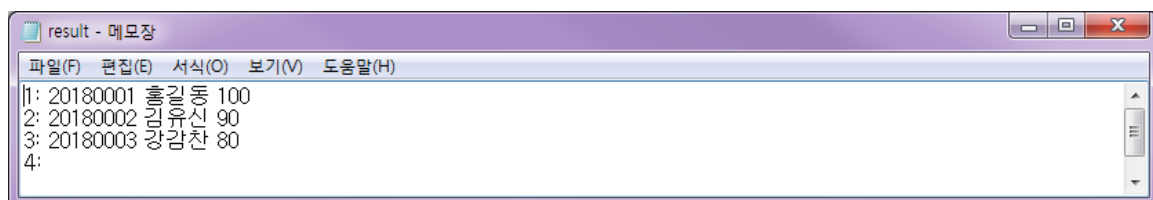
출력 형식 지정

플래그	설명
ios::fixed	고정 소수점 표기법 사용
ios::scientific	과학적 표기법 사용(지수를 이용하여 표기)
ios::showpoint	소수점을 반드시 표시한다.
ios::showpos	양수 부호를 반드시 출력한다.
ios::right	값을 출력할 때 오른쪽 정렬을 사용한다.
ios::left	값을 출력할 때 왼쪽 정렬을 사용한다.
ios::dec	값을 출력할 때 10진법을 사용한다.
ios::oct	값을 출력할 때 8진법을 사용한다.
ios::hex	값을 출력할 때 16진법을 사용한다.
ios::uppercase	지수나 16진법으로 표시할 때 대문자를 사용한다.
ios::show	8진수이면 앞에 0을 붙이고 16진수이면 앞에 0x를 붙인다.

17

Lab: 줄번호를 붙여보자.

- 소스가 저장된 텍스트 파일을 읽어서 각 줄의 앞에 숫자를 붙인 후에 출력 파일에 기록하여 보자.



18

Solution

```
int main()
{
    ifstream is("scores.txt");
    ofstream os("result.txt");
    if (is.fail()) {
        cerr << "파일 오픈 실패" << endl;
        exit(1);
    }
    if (os.fail()) {
        cerr << "파일 오픈 실패" << endl;
        exit(1);
    }
}
```

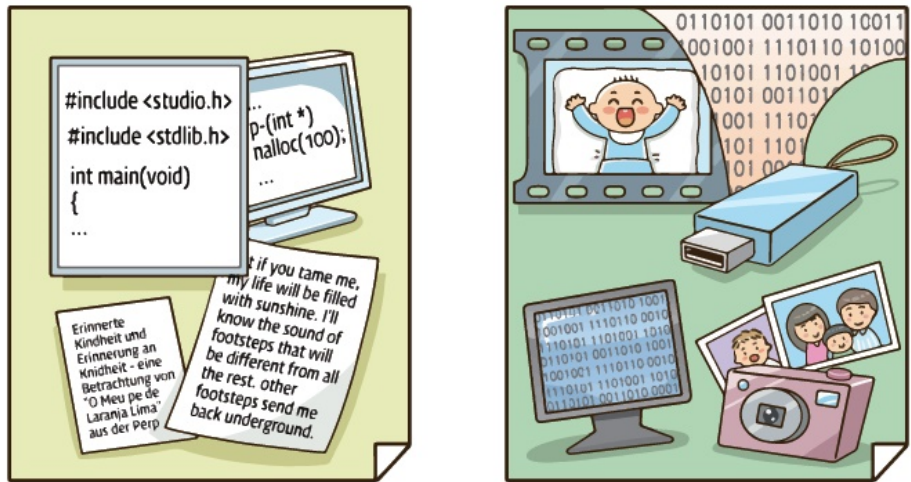
19

Solution

```
char c;
int line_number = 1;
is.get(c);
os << line_number << ": ";
while (!is.eof())
{
    os << c;
    if (c == '\n') {
        line_number++;
        os << line_number << ": ";
    }
    is.get(c);
}
return 0;
}
```

20

13.4 텍스트와 이진 파일



텍스트 파일: 문자로 구성된 파일

이진 파일: 데이터로 구성된 파일

그림 13.3 텍스트 파일과 이진 파일

21

텍스트와 이진 파일의 저장 방법 비교

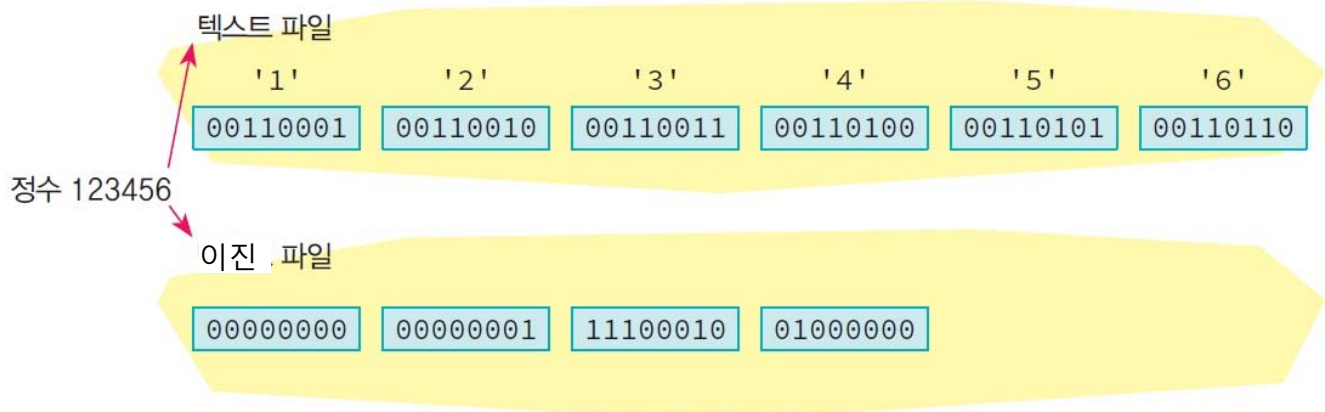


그림 13.4 정수 123456을 텍스트 파일에 저장하는 경우와 이진 파일에 저장하는 경우의 비교

22

이진 파일 입출력

```
ofstream os{ "test.dat", ofstream::binary };
int x=5;
os.write((char*)&x, sizeof(int));    // 정수 변수는 4바이트로 이루어져 있다.
```

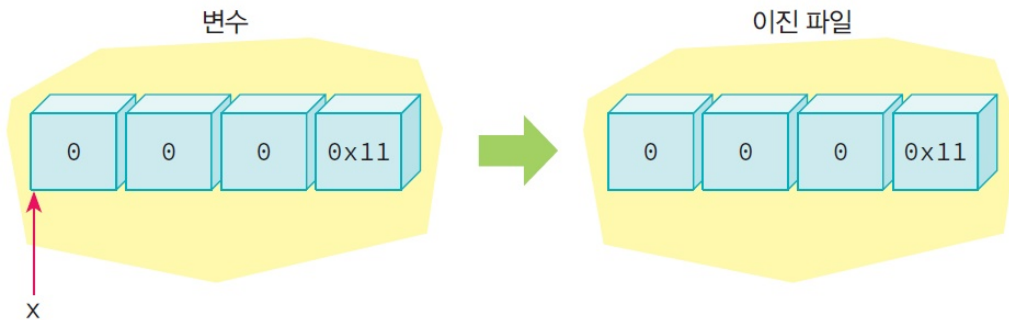


그림 13.5 이진 파일에 저장되는 모습

23

예제

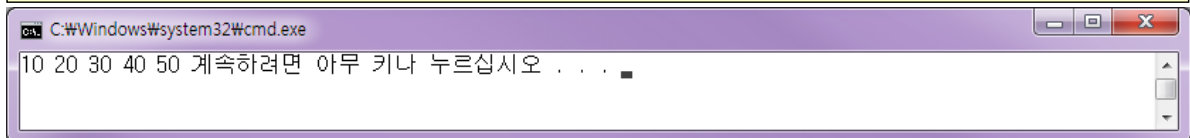
```
int main()
{
    int buffer[] = { 10, 20, 30, 40, 50 };
    ofstream os{ "test.dat", ofstream::binary };
    if (!os)
    {
        cout << "test.txt 파일을 열 수 없습니다." << endl;
        exit(1);
    }
    os.write((char *)&buffer, sizeof(buffer));
    return 0;
}
```

```
00000000: 0A 00 00 00 14 00 00 00 | 1E 00 00 00 28 00 00 00 | 00 00 00 00 00 00 00 00
00000010: 32 00 00 00                | 02 00 00 00 00 00 00 00
```

24

예제

```
int main()
{
    int buffer[5];
    ifstream is{ "test.dat", ifstream::binary };
    if (!is)
    {
        cout << "test.txt 파일을 열 수 없습니다." << endl;
        exit(1);
    }
    is.read((char *)&buffer, sizeof(buffer));
    for(auto& e: buffer)
        cout << e << " ";
    return 0;
}
```



25

Lab: 이진 파일 복사 프로그램

- 하나의 이미지 파일을 다른 이미지 파일로 복사하는 프로그램을 작성하여 보자.



26

Solution

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

int main() {
    string ifile, ofile;
    cout << "원본 파일 이름:";
    cin >> ifile;
    cout << "복사 파일 이름:";
    cin >> ofile;
    ifstream source(ifile, ios::binary);
    ofstream dest(ofile, ios::binary);
```

27

Solution

```
#if 1
    dest << source.rdbuf();
#else
    if (source.is_open() && dest.is_open()) {
        while (!source.eof()) {
            dest.put(source.get());
        }
    }
#endif
    return 0;
}
```

28

임의 접근 파일

- **순차 접근(sequential access)** 방법: 데이터를 파일의 처음부터 순차적으로 읽거나 기록하는 방법
- **임의 접근(random access)** 방법: 파일의 어느 위치에서든지 읽기와 쓰기가 가능한 방법

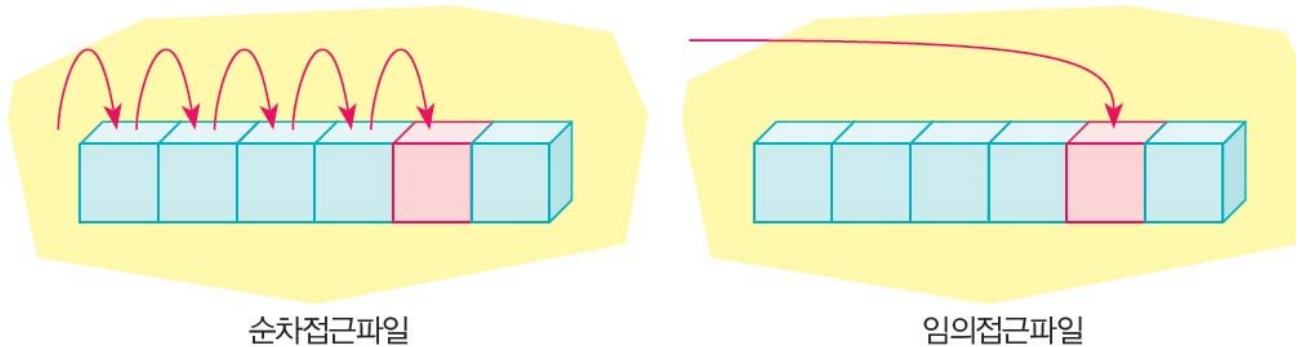
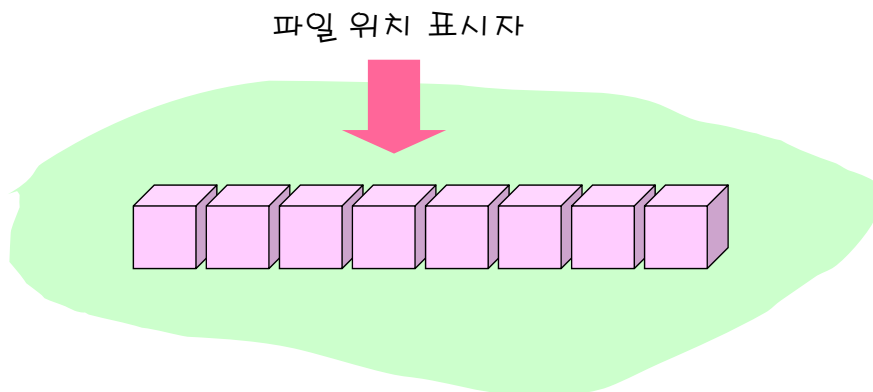


그림 13.6 순차 접근 파일과 임의 접근 파일의 비교

29

임의 접근 파일의 원리

- 파일 위치 표시자: 읽기와 쓰기 동작이 현재 어떤 위치에서 이루어지는 지를 나타낸다.



- 강제로 파일 위치 표시자를 이동시키면 임의 접근이 가능

30

임의 접근 관련 함수

```
seekg(long offset, seekdir way);
```

way	설명
ios::beg	처음부터의 offset
ios::cur	현재 위치부터의 offset
ios_end	파일의 끝에서부터의 offset

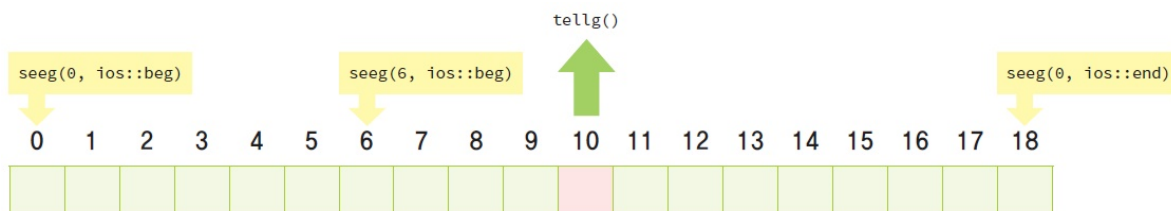
```
is.seekg(100, ios::beg);
```

```
is.seekg(0, ios::end);
```

```
is.seekg(-100, ios::cur);
```

31

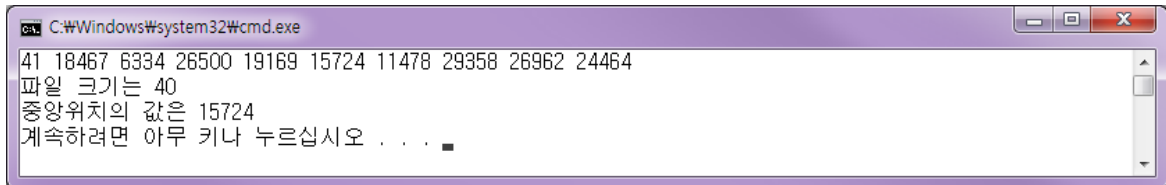
예



32

예제

- 예제에서는 10개의 난수를 저장한 이진 파일을 만들고 파일의 크기를 알아낸 다음에 파일의 중간으로 파일 위치 표시자를 이동시켜서 그 위치에 있는 난수를 읽어오는 프로그램을 작성하여 보자.



```
C:\Windows\system32\cmd.exe
41 18467 6334 26500 19169 15724 11478 29358 26962 24464
파일 크기는 40
중앙위치의 값은 15724
계속하려면 아무 키나 누르십시오 . . .
```

33

Solution

```
const int SIZE = 10;
int main()
{
    int data;
    // 이진 파일을 쓰기 모드로 연다.
    ofstream os{ "test.dat", ofstream::binary };
    if (os.fail()) {
        cout << "test.dat 파일을 열 수 없습니다." << endl;
        exit(1);
    }
    for (int i = 0; i < SIZE; i++) {
        data = rand();
        cout << data << " ";
        os.write((char *)&data, sizeof(data));
    }
    os.close();
}
```

34

Solution

```
// 이진 파일을 읽기 모드로 연다.
ifstream is{ "test.dat", ifstream::binary };
if (is.fail()) {
    cout << "test.dat 파일을 열 수 없습니다." << endl;
    exit(1);
}
// 파일 크기를 알아낸다.
is.seekg(0, ios::end);
long size = is.tellg();
cout << endl << "파일 크기는 " << size << endl;

// 파일의 중앙으로 위치 표시자를 이동시킨다.
is.seekg(size/2, ios::beg);
is.read((char *)&data, sizeof(int));
cout << "중앙위치의 값은 " << data << endl;
return 0;
}
```

35

Lab: 행맨 업그레이드



36

Solution

```
#include <conio.h>
#include <iostream>
#include <string>
#include <fstream>
#include <vector>
using namespace std;
int main() {
    vector<string> words;
    ifstream infile("d:/words.txt");

    while (infile) {
        string word;
        infile >> word;
        words.push_back(word);
    }
}
```

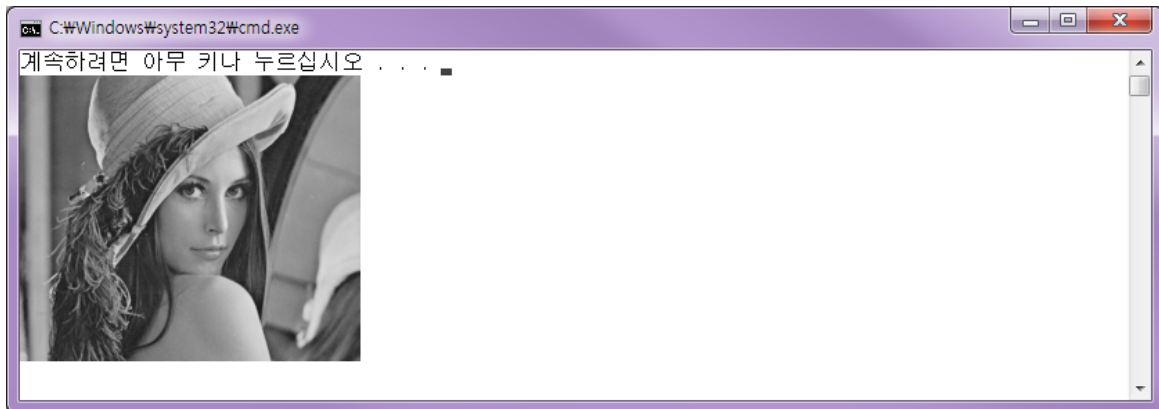
37

Solution

```
while (true) {
    string r = words[rand() % words.size()];
    cout << "이번에 선택된 단어는 " << r << endl;
    getch();
}
return 0;
}
```

38

Lab: 이미지 파일을 읽어서 표시해보자.



39

Solution

```
#include <windows.h>
#include <iostream>
#include <fstream>
using namespace std;
int main()
{
    HDC hdc = GetWindowDC(GetForegroundWindow());
    // 이진 파일을 쓰기 모드로 연다.
    ifstream is{ "d:\\lena(256x256).raw", ifstream::binary };
    if (is.fail())
    {
        cout << "d:\\lena(256x256).raw 파일을 열 수 없습니다." << endl;
        exit(1);
    }
    int size = 256 * 256;
    char * memblock = new char[size];
    is.read(memblock, size);
    is.close();
```

40

Solution

```
int r, c;
for (r = 0; r < 256; r++) {
    for (c = 0; c < 256; c++) {
        int red, green, blue;
        red = green = blue = memblock[r * 256 + c];
        SetPixel(hdc, c, r, RGB(red, green, blue));
    }
}
delete memblock;
return 0;
}
```

41

- 텍스트 파일에서 순서를 반대로 하여 문자를 읽고 화면에 출력하는 프로그램을 작성하라. **seekg()**와 **tellg()**를 이용한다.
 - ▣ 파일크기: `in.seekg(0, ios::end); i = (long) in.tellg();`

42

- 사용자로부터 다음과 같은 형식으로 사용자의 번호, 이름, 전화번호, 이메일 주소를 입력받아서 파일로 저장한다. 입력이 끝나면 사용자로부터 번호를 입력받아서 그 번호에 해당하는 전화 번호를 출력하는 프로그램을 작성하라. 벡터를 사용해도 된다.

번호	이름	전화번호	이메일주소
1	홍길동	010-1111-1111	hong@hanmail.net
2	김유신	010-2222-2222	kim@hanmail.net

- 파일에서 특정한 단어를 찾아서 단어가 위치한 라인의 문장과 그 번호를 출력하는 프로그램을 작성하라.
 - ▣ `getline()`으로 한 줄을 읽어서 `string` 클래스가 가지고 있는 `find()` 함수를 사용한다.
- 텍스트 파일을 입력받아서 파일 안에 포함된 단어의 개수를 계산하여 출력하는 프로그램을 작성하라.

Q & A

