객체지향프로그래밍 정리노트(Week9)

202004029 김정호, 202004040 노성민

Q1. 12-1 키보드로 입력 받아 텍스트 파일 저장하기

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

int main() {
    char name[10], dept[20];
    int sid;
    cout << "이름>>";
    cin >> name;
    cout << "학번>>";
    cin >> sid;
    cout << "학과>>";
    cin >> dept;
    ofstream fout("c:\\temp\\student.txt");
    if (!fout) {
        cout << “c:\\temp\\student.txt 파일을 열 수 없다";
            return 0;
    }
    fout << name << endl;
    fout << sid << endl;
    fout << dept << endl;
    fout.close();
}
```

김정호: 이 예제는 ofstream 클래스의 객체를 만들어 파일을 열어 그 안의 정보들을 바꾸는 예제야.
먼저 ostream 객체인 fout을 만들고 ofstream fout("c:\\temp\\student.txt"); 파일 출력 스트림을 생성과 동시에 파일을 열 수 있도록 코드를 작성했어. 이 방법 말고 파일을 여는 방법이 뭔지 알아?

노성민: open 함수를 이용해서 student.txt 가 없다면 생성하고 이미 존재한다면 기존의 student.txt 의 내용을 모두 지우고 파일의 맨 앞에서부터 쓸 준비를 할 수 있도록 해줘.

김정호: 맞아. 이제 << 연산자를 통해 name, sid, dept 의 값을 출력하고 close() 함수를 호출해서 파일을 닫아. close() 함수는 호출 이후 fout을 이용하여 파일 쓰기를 할 수 없고 다시 파일을 열어야 쓸 수 있다는 점을 참고해.

Q2. 12-3 get()을 이용한 텍스트 파일 읽기

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

int main() {
    const char* file = "c:\\windows\\system.ini";
    ifstream fin(file);
    if (!fin) {
        cout << file << " 열기 오류" << endl;
        return 0;
    }
    int count = 0;
    int c;
    while ((c = fin.get()) != EOF) {
        cout << (char)c;
        count++;
    }
    cout << "읽은 바이트 수는 " << count << endl;
    fin.close();
}
```

김정호: 이 예제는 get 함수를 이용해서 텍스트 파일을 읽는 예제야. 코드를 살펴보면 우선 캐릭터형 포인터 file을 선언해 주소를 저장해주고 ifstream fin(file)을 통해 파일 입력 스트림을 생성하고, 텍스트 I/O로 파일을 열어 줬어. 그리고 get 함수를 통해 EOF를 만날 때까지 반복해서 얻고자 하는 결괏값을 출력한 예제라고 볼 수 있어.

Q3. 12-4 텍스트 파일 연결

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

int main() {

    const char* firstFile = "c:\\temp\\student.txt";
    const char* secondFile = "c:\\windows\\system.ini";

    fstream fout(firstFile, ios::out | ios::app);
    if (!fout) {
        cout << firstFile << " 열기 오류";
        return 0;
    }
```

```
    fstream fin(secondFile, ios::in);
    if (!fin) {
        cout << secondFile << " 열기 오류";
        return 0;
    }

    int c;
    while ((c = fin.get()) != EOF) {
        fout.put(c);
    }

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

노성민: 이 예제는 파일 모드에 관한 예제인데 여기서 살펴볼 부분은 색칠해 놓은 부분인데 먼저 fstream fout(firstFile, ios::out | ios::app); ios::app을 이용해서 student 파일에 덧붙여서 파일을 쓸 수 있도록 했어. fstream fin(secondFile, ios::in); 이 부분은 파일을 읽기위 한 부분으로 생략해도 디폴트 파일 모드로 지정되어 있어.

Q4. 12-8 read()로 텍스트 파일을 바이너리 I/O로 읽기

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

int main() {
    const char* file = "c:\\windows\\system.ini";
    ifstream fin;
    fin.open(file, ios::in | ios::binary);
    if (!fin) {
        cout << "파일 열기 오류";
        return 0;
    }
    int count = 0;
    char s[32];
    while (!fin.eof()) {
        fin.read(s, 32);
        int n = fin.gcount();
        cout.write(s, n);
        count += n;
    }
    cout << "읽은 바이트 수는 " << count << endl;
    fin.close();
```

}

김정호: 이번에는 텍스트 파일 말고 바이너리 파일 ios::binary를 통해 바이너리 I/O 모드로 파일을 열었어. 이를 통해 앞서 텍스트 I/O 모드로 파일로 읽었을 때는 생략되었던 '\r' 문자를 생략하고 '\n'문자만 리턴하는데 이러한 점에서 차이가 있어.

Q5. 13-4 0으로 나누는 예외 처리

합과 인원수를 입력 받아 평균을 내는 코드에, 인원수가 0이거나 음수가 입력되는 경우 예외 처리하는 프로그램을 작성하라.

```
#include <iostream>
using namespace std;
int main() {
    int n, sum, average;
    while (true) {
        cout << "합을 입력하세요>>";
        cin >> sum;
        cout << "인원수를 입력하세요>>";
        cin >> n;
        try {
            if (n <= 0) // 오류 탐지
                throw n; // 예외 발생. catch(int x) 블록으로 점프
            else
                average = sum / n;
        }
        catch (int x) {
            cout << "예외 발생!! " << x << "으로 나눌 수 없음" << endl;
            average = 0;
            cout << endl;
            continue;
        }
        cout << "평균 = " << average << endl << endl; // 평균 출력
    }
}
```

노성민: try-catch 문은 try 문에서 오류가 발생하면 catch를 통해 그 오류에 대응하는 catch 문을 작성하면 돼. 나머지 코드는 기본적인 평균을 구하는 코드라 설명은 더 안할게.