객체지향프로그래밍

Term Project

지능로봇공학과

2021042031

장 준 혁

프로그램 소스코드 및 설명

#1 header.h

```
#pragma once
#include <map>
#include <string>
#include <iostream>
#include <vector>
#include <list>
#include <fstream>
#pragma warning(disable:4996)
using namespace std;
class student {
public:
       string Name;//이름
        int password;//비밀번호
        int StudentID;//학번
        string Department;//학과
       map <string, string> classTable; //성적 정보를 저장하는 map
       student() {};
       void setInfo(int id, string name, string department,int pw);
       void setTable(string lect, string score);
};
#2 main.cpp
#include "header.h"
void run();
void readLoginFile();
int main()
{
        readLoginFile(); //로그인 정보가 담긴 파일을 불러옴.
        run(); //프로그램 실행
}
```

#3 function.cpp

```
#include "header.h"
#include <fstream>
//객체 정보를 설정함
void student∷setInfo(int id, string name, string department, int pw) {
        this->StudentID = id;
        this->Name = name;
        this->Department = department;
        this->password = pw;
}
//성적 정보를 저장함
void student::setTable(string lect, string score) {
        classTable.insert({ lect,score });
}
vector<student> studentList;
                                 //학생 객체를 담는 vector
map<string, int> Login; //로그인 정보를 담는 map
void run();
int input();
void readLoginFile();
void adminMenu();
void studentMenu();
void writeFile();
int input();
void readLoginFile();
void stop();
void addStudent();
void viewTableStud();
void viewTableProf();
void deleteStudent();
void run() {
        while (1) {
                system("cls"); //기존 화면 지우기
                cout << "Availiable Login operations:" << endl;</pre>
                cout << "1. Admin Login" << endl;</pre>
                cout << "2. Student Login" << endl;</pre>
                cout << "3. Exit" << endl;</pre>
                cout << "\nEnter menu : ";
                switch (input()) {
                case 1:
                         adminMenu();
                         break;
                case 2:
                         studentMenu();
                         break;
                case 3:
                         stop(); //종료함수
                default:
                                         //보기에 없을 시 재실행
                         continue;
                }
        }
}
```

```
string tmpID;
       int tmpPW;
       system("cls");
       cout << "관리자 아이디를 입력하세요 >> ";
       cin >> tmpID;
       cout << "관리자 비밀번호를 입력하세요 >> ";
       cin >> tmpPW;
       if (tmpID == "Admin" && tmpPW == Login.find("Admin")->second) { //key값이
"Admin"이고 비밀번호가 value와 일치할 때
       readmin:
               system("cls");
               cout << "- Logged in as Admin -" << endl;</pre>
               cout << "1. Add Students" << endl;</pre>
               cout << "2. Delete Students" << endl;</pre>
               cout << "3. View Table" << endl;</pre>
               cout << "4. Main Menu" << endl;</pre>
               cout << "5. Exit" << endl;</pre>
               cout << "\n Enter menu : ";
               switch (input()) {
               case 1:
                       addStudent();
                                      //학생 추가 함수 실행
                       goto readmin;
               case 2:
                       deleteStudent(); //학생 삭제 함수 실행
                       goto readmin;
               case 3:
                       viewTableProf(); //학생 정보 조회 함수 실행
                       goto readmin;
               case 4:
                       return; //관리자 메뉴 함수를 종료 -> 메인메뉴로 복귀
               case 5:
                       stop(); //종료함수
               default:
                       goto readmin; //보기에 없을 시 재실행
               }
       }
       else {
               cout << "ID와 비밀번호가 일치하지 않습니다.\n";
               system("pause");
               return;
       }
}
void studentMenu() {
       string name;
       int PW;
       system("cls");
       cout << "아이디를 입력하세요 >> ";
       cin >> name;
       cout << "비밀번호를 입력하세요 >> ";
       cin >> PW;
        if (name == Login.find(name)->first && PW == Login.find(name)->second) { //이름과
```

void adminMenu() {

```
비밀번호가 일치할 때 실행
       restudent:
               system("cls");
               cout << "- Logged in as Student -" << endl;</pre>
               cout << "1. View Table" << endl;</pre>
               cout << "2. Main Menu" << endl;</pre>
               cout << "3. Exit" << endl;</pre>
               cout << "₩n Enter menu : ";
               switch (input()) {
               case 1:
                       viewTableStud(name);
                       goto restudent; //학생 메뉴로 복귀
               case 2:
                       return;
               case 3:
                       stop();
               default:
                       goto restudent; //보기에 없을 시 재실행
       }
       else {
               cout << "ID와 비밀번호가 일치하지 않습니다.₩n";
               system("pause");
               return;
       }
}
void addStudent() {
       while (true) {
               string name, score, Dptmt, lect;
               int ID, PW;
               student tmp;
                             //임시 객체 생성
               system("cls");
               cout << "학생 이름(quit입력 시 나가기) >> ";
               cin >> name;
               if (name == "quit")
                                      //quit입력 시 함수 종료.
                      break;
               cout << "학번 >> ";
               cin >> ID;
               cout << "학과 >> ";
               cin >> Dotmt;
               cout << "비밀번호 >> ";
               cin >> PW;
               tmp.setInfo(ID, name, Dptmt,PW); //임시 객체 정보 저장
                                          //로그인 정보 저장
               Login.insert({ name,PW });
               while (true) { //연속하여 과목정보 입력
                       system("cls");
                       cout << "과목(quit입력 시 종료) : ";
                       cin >> lect;
                       if (lect == "quit") break;
                       cout << "성적 : ";
                       cin >> score;
                       if (cin.fail()) //정수가 아닌 입력이 들어올 시
                       {
```

```
cin.clear();
                             cin.ignore(numeric_limits<streamsize>::max(), '\mathbb{\mathbb{M}n');
                             continue;
                                            //버퍼를 비우고 재실행
                      }
                      tmp.setTable(lect, score); //임시객체의 성적 정보 입력
                                          //학생 리스트에 임시 객체 정보 저장
              studentList.push_back(tmp);
       }
}
void deleteStudent() {
       string name;
       int tmp;
       bool flag = true;
       cout << "삭제할 학생의 이름을 입력하세요. >> ";
       cin >> name;
       for (int i = 0; i < studentList.size(); i++) {</pre>
                                                   //학생 리스트를 훑으며
                                                   //이름이 같은 학생을 찾은 후
              if (studentList[i].Name == name) {
                      tmp = i; //임시변수에 위치 저장
                      flag = false; //플래그 변경
                      break;
              }
       if (flag) { //이름이 같은 학생이 없으면
              cout << "학생 정보 없음" << endl;
              return;
       studentList.erase(studentList.begin() + tmp);
                                                  //해당 학생 목록에서 제거
       Login.erase(name); //해당 학생의 로그인 정보 삭제
}
void viewTableProf() {
       for (auto iter = studentList.begin(); iter != studentList.end(); iter++) //학생
리스트를 훑으며
       {
              cout << "이름: " << iter->Name << "\n학번: " << iter->StudentID << "\n학과: "
<< iter->Department << endl; //해당 학생정보 출력
              for (auto iter2 = iter->classTable.begin(); iter2 != iter->classTable.end();
iter2++) //해당 학생의 과목정보를 훑으며
              {
                      cout << iter2->first << " : " << iter2->second << endl; //과목명과
성적 출력
              }
              cout << endl;</pre>
       system("pause");
}
void viewTableStud(string name) {
       for (auto iter = studentList.begin(); iter != studentList.end(); iter++)
       {
              if (iter->Name == name) {//이름이 같은 경우 -> 자기 자신의
                      for (auto iter2 = iter->classTable.begin(); iter2 != iter-
```

```
//과목 정보 훑으며
>classTable.end(); iter2++)
                             cout << iter2->first << " : " << iter2->second << endl;</pre>
       //과목명과 성적 출력
                      }
                      break;
              }
       }
       cout << endl;</pre>
       system("pause");
}
void readLoginFile() {
       string name, dptmt, lect, score;
       int password, id;
       ifstream file;
       //로그인 정보 입력
       file.open("Login.txt");
       if (file.fail()) {
              cerr << "파일 열기에 실패했습니다.";
              stop();
       }
       while (!file.eof()) {
              file >> name >> password; //로그인 정보(이름, 비밀번호) 읽기
              Login.insert({ name,password }); //로그인 정보 map에 저장
       }
       file.close();
       //학생 정보 입력
       file.open("info.txt");
       if (file.fail()) {
              cerr << "파일 열기에 실패했습니다.";
              stop();
       while (!file.eof()) {
              student tmp;
              name = "";
                             //이름 변수 초기화
              file >> name >> id >> dptmt;
              if (name=="") break; //추가적으로 입력된 것이 없으면 읽기 종료.
              tmp.setInfo(id, name, dptmt, Login.find(name)->second); //읽은 정보를
임시객체에 저장
              while (true) {
                      file >> lect; //한 단어 입력
                      if (lect == "--") break; //그 단어가 "--"일 때 반복 종료.
                      file >> score; //그렇지 않으면 마저 입력
                      tmp.setTable(lect, score);
                                                  //과목명과 성적 객체 성적정보 map에
저장
              studentList.push_back(tmp); //임시객체정보를 학생 리스트vector에 저장
       file.close();
}
void writeFile() {
```

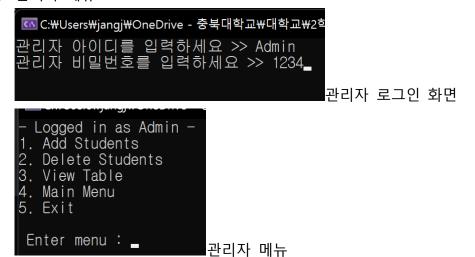
```
ofstream file;
       //로그인 정보 입력
       file.open("Login.txt");
       if (file.fail()) {
              cout << "파일 열기에 실패했습니다.";
              exit(0);
       }
       for (auto iter = Login.begin(); iter != Login.end(); iter++)
              file << iter->first << " " << iter->second << endl; //사용자명과 비밀번호 쓰기
       file.close();
       //학생 정보 입력
       file.open("info.txt");
       if (file.fail()) {
              cout << "파일 열기에 실패했습니다.";
              exit(0);
       }
       for (auto iter = studentList.begin(); iter != studentList.end(); iter++)
              file << iter->Name << " " << iter->StudentID << " " << iter->Department <<
      //학생 기본정보 입력
endl;
              for (auto iter2 = iter->classTable.begin(); iter2 != iter->classTable.end();
iter2++)
              {
                     file << iter2->first << " " << iter2->second << endl;
                                                                      //성적정보
입력
              file << "--" << endl; //모든 정보 입력 후 구분자 입력
       file.close();
}
void stop() {
       writeFile(); //변수와 STL에 저장되어있는 정보를 파일에 쓰기
       exit(0);//프로그램 종료
}
int input() { //입력받는 함수, 정수형 변수가 아니면 입력 오류로 예외처리
                    //입력 정보를 담을 변 선언
       int input;
       cin >> input;
       if (cin.fail()) //정수가 아닌 입력이 들어올 시
              cin.clear();
              cin.ignore(numeric_limits<streamsize>::max(), '₩n');
              return 0; //버퍼를 비우고 0반환->메뉴 함수 재실행 유도
       }
       return input; //입력값 반환
}
```

프로그램 실행 화면

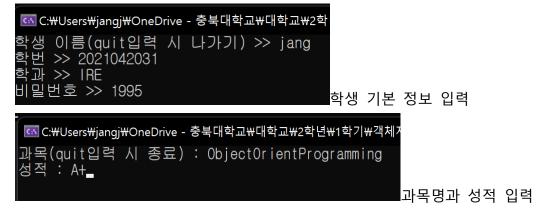
1. 프로그램 실행 첫 화면



2. 관리자 메뉴



A. 학생 추가



B. 학생 삭제

```
C:₩Users₩jangj₩OneDrive - 충북대학교₩대학교₩2학년₩1학7

- Logged in as Admin -
1. Add Students
2. Delete Students
3. View Table
4. Main Menu
5. Exit

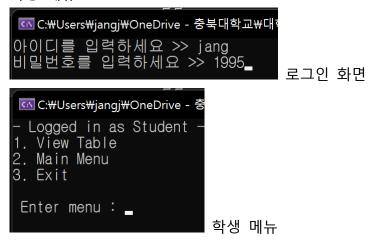
Enter menu : 2
삭제할 학생의 이름을 입력하세요. >> jang
학생 정보 삭제
```

C. 학생 정보 조회

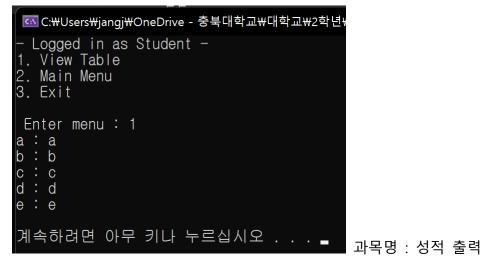
```
ጩ C:₩Users₩jangj₩OneDrive - 충북대학교₩대학교₩2학년₩1학
2. Delete Students
3. View Table
4. Main Menu
5. Exit
 Enter menu : 3
이름: jang
학번: 2021
학과: ire
a : a
b : b
c : c
d : d
e : e
이름: kim
학번: 212
학과: ijhb
kj : kj
kjn : kjn
I,m : kj
이름: qwer
학번: 1234
학파: wer
efsd : tgb
ewdf : efdv
wdf : ewfd
계속하려면 아무 키나 누르십시오 . . . _
```

모든 학생 정보 출력

3. 학생 메뉴



A. 과목명 및 성적표 출력



느낀점 및 고찰

학생관리 프로젝트를 제작하며 학생 정보를 저장하고 불러오는 부분과 어느 템플릿으로 관리해야 효율적인지 선정하는 것이 가장 어려웠다. Vector, list, map 등 객체지향프로그래 밍의 마지막 부분에 배운 STL에 대한 이해가 완벽하지 않아 복습하며 상기시켰다. 알맞은 템플릿과 함수를 정하고 이용할 때 생겨난 오류를 고쳐 나가며 코드를 다듬어 나갔다.

이번 프로그램에서 아쉬운 점은 동명이인에 대한 대응 준비를 하지 않았다는 것이다. Map과 pair를 이용하여 고유번호인 학번을 사용하려 했으나 쉽게 진행할 수 없어 아쉬운 대로 동명이인을 입력하지 않도록 유의하며 사용할 수밖에 없어졌다. 또한 관리자의 ID와 비밀번호를 txt파일에 넣어 시작해줘야 하는 단점이 있다.

하지만 UI/UX를 위해 고민한 부분이 있다. 프로그램 작동 확인을 하며 switch문을 위한 정수변수 입력에서 키보드 오타로 인한 입력 오류가 자주 발생함을 인지했다. 이를 해결하기 위해 자료형에 맞지 않는 입력이 주어졌을 때 버퍼를 비우고 다시 입력 대기하는 예외처리를 추가하여 사용자의 입력 실수로 인한 오류를 줄였다.

프로그램의 흐름은 main()은 굵은 뼈대를 이루어 프로그램을 실행하기 전 해야 할 전처리과정과 실질적인 프로그램실행을 순차적으로 실행해주고, 실질적인 main문이 되어주는 run()함수는 switch문을 통해 사용자가 원하는 기능을 실행 후 종료 시 다시 돌아오는 구조로 일종의 가지치기를 했다. 따라서 사용자가 얼마나 오래 많이 사용하든 낭비되는 메모리를 줄일 수 있다.

과제 안내에는 Admin이 입력한 학생 정보를 따로 파일에 저장하라는 지시가 없었으나학생 관리 프로그램은 항상 켜 두는 것이 아닌 중간중간 종료할 수 있다는 사용환경을생각했을 때, 기존에 관리되던 정보들이 저장되어 있다가 다음 실행 시 불러와 이어서관리할 수 있도록 제작하였다. 하지만 txt파일로 관리되어 보안에 취약하다는 단점이 있다.

이번 프로젝트는 한 학기 동안 배운 전체적인 내용을 활용할 수 있었고 또한, 표준 템플 릿 라이브러리를 이용해서 기능이 정해져 있는 함수들을 많이 사용하여 작성할 수 있었다.