[한밭대생을 위한 일정 관리 프로그램]

한밭대학교 컴퓨터공학과
이민주

1. 시스템 요구 정리

◎ 기능 정리

가. 사용자의 일정 등록

나. 캘린더 출력

다. 한밭대학교의 학사일정을 포함한 일정 검색

◎ 시나리오 정리

가. 사용자의 일정 등록

1) 날짜와 일정 이름 입력 - 사용자의 일정 텍스트 파일에 저장 2) 일정 등록 3) 추가로 등록할 일정의 여부 확인

나. 캘린더 출력

1) 메뉴에서 년도와 달을 선택 - 해당 달의 캘린더 출력

나. 한밭대학교의 학사일정을 포함한 일정 검색

1) 메뉴에서 년도와 달을 선택 - 해당 달의 학사 일정 출력 - 해당 달의 개인 일정 출력 2) 일정을 다시 검색할 의사가 있는지 확인

2. 기초 자료 수집 및 분석

◎ 한밭대학교 학사일정(2019년 12월 - 2020년 2월)

2019년 12월 18일 - 12월 25일 2학기성적입력기간

2019년 12월 20일 종강

2019년 12월 23일 - 1월 21일 겨울계절학기

2019년 12월 25일 성탄절

2019년 12월 26일 - 12월 31일 성적이의신청기간

2020년 1월 2일 - 1월 3일 성적전산처리
2020년 1월 3일 - 1월 31일 1학기현장실습신청기간
2020년 1월 21일 - 1월 23일 성적입력기간
2020년 1월 21일 - 2월 12일 장학생선발기간
2020년 1월 24일 - 1월 27일 설날
2020년 1월 28일 - 2월 03일 수강신청기간
2020년 2월 18일 - 2월 21일 등록금납부기간
2020년 2월 24일 - 2월 25일 신입생 OT

3. 시스템 설계

◎ 코드 설계

가. 일정 날짜 : 미국식 날짜 표기법 사용
2019년 12월 2일 -> 12 2 2019 로 저장

나. 일정 이름 : 영어로 입력받아서 일정을 저장

다. 달 력 : 일월화수목금토 순으로 출력
- 일요일 0, 월요일 1, 화요일 2, 수요일 3, 목요일 4,
금요일 5, 토요일 6 으로 설정
- 토요일에 해당되는 일을 출력하면 줄바꿈 적용

◎ 입출력 설계

가. 화면 설계

1) 시작화면

```
<  Launches a calendar management program >

          ...Loading...
```

2) 주메뉴 화면

```
[ Calendar Management Program ]

  1. Register schedule
  2. Search schedule
  3. Exit

--> Please enter the number of the Main menu you want:
```

3) 부메뉴 1

```
        [  1. Register schedule ]
-    Enter date(month day year):
     Calendar name(Please enter in ENG):


        < Successfully registered! >
  1. Register another schedule
  2. Return to Main menu
 -->
```

4) 부메뉴 2

```
      [ 2. Search schedule ]
    Month Year: ** ****


              ** / ****
---------------------------------------
SUN   MON  TUE  WED  THU  FRI  SAT
 1     2    3    4    5    6    7
 8     9   10   11   12   13   14
 15   16   17   18   19   20   21
 22   23   24   25   26   27   28
 29   30   31
---------------------------------------
[ 학사 일정 ]
 **/**/**** Christmas_Day
[ 개인 일정 ]
 **/**/****

       < Your search is complete! >
1. Search again
2. Return to Main menu
-->
```

◎ 파일 설계

1) 한밭대학교 학사일정 정리 파일 (“academic_calendar.txt”)

- 첫 줄에 학사일정의 수 저장

2) 사용자의 일정 정리 파일 (“user.txt”)

- 첫 줄에 개인일정의 수 저장
- 월-일-년-일정 순으로 저장

◎ 구조체 설계

구조체명	Day		
변수명	내용	자료변환	비고
year	년	int	사용자의 일정
month	월	int	
day	일	int	
name	일정 이름	string	

구조체명	Fday		
변수명	내용	자료변환	비고
year	년	int	학사 일정
month	월	int	
day	일	int	
tmonth	최대 월	int	
tday	최대 일	int	
name	일정 이름	string	

◎ 클래스 설계

클래스명	Schedule	클래스 내용	일정		
구분	내용	접근권한	자료형(반환)	변수명(함수명)	비고
멤버변수	일정 카운트 학사일정 카운트 일정 구조체 학사일정 구조체	protected	int int Day Fday	count fcount *da *fa	
멤버함수	생성자 소멸자 설정자(전체) 일정 출력 일정 등록 학사일정 출력 개인일정 출력	public	- - void void void void void	Schedule ~Schedule Set_Schedule showschedule set_Schedule out_todo out_user	

클래스명	ToDo	클래스 내용	달력을 상속받아 캘린더 출력		
구분	내용	접근권한	자료형(반환)	변수명(함수명)	비고
멤버변수	최대 날짜	private	int	max_day[12]	배열선언
멤버함수	생성자 설정자(전체) 접근자(최대 날짜) 캘린더 출력 전체 일정 검색 윤년 확인	public	- void void void void int	ToDo set_ToDo get_max_day print_calendar findschedule yoonyear	

◎ 프로그램 설계

함수명	main			
용도	기초 데이터 생성 및 메뉴 처리			
	자료명	**자료형**	**변수명**	**매체**
입력	주메뉴	int	select	화면
출력				
작업변수	스케쥴 객체 일정 객체	Schedule ToDo	sch to	
처리과정	1. 스케줄 객체와 일정 객체 생성 2. 주메뉴 선택 3. 주메뉴 선택에 따라 switch(주메뉴); 2-1. case 1: 부메뉴 1 함수(set_Schedule()) 호출 2-2. case 2: 부메뉴 2 함수(findschedule()) 호출 2-3. case 3: 프로그램 종료 2-4. default: “You've entered something wrong. Please re-enter.” 출력 3. 1~2 반복			

클래스명	Schedule	
함수명	**내용**	**처리절차**
Schedule	생성자	1. 텍스트 파일 열기 (“user.txt”) 2. 제일 첫 줄에 있는 일정 수(count)를 저장 3. Day 배열을 동적 생성(count+5) 4. i=0부터 i는 count까지 Day 배열에 저장 5. 텍스트 파일 닫기
set_Schedule	일정 등록	1. 메인메뉴에서 1번 선택 2. 날짜 입력 (imonth iday iyear) 2-1. 월 일 년 순으로 입력 2-2. imonth가 0 이하이거나 13 이상이면 재귀함수 호출 3. 일정 이름 입력 4. “user.txt”파일을 열고 뒤에 이어서 일정 쓰기 5. Day 배열에 저장하고 count+1 6. 추가로 등록할 일정의 여부 확인
out_todo	학사일정 출력	1. 학사일정 파일 열기(“acadaminc_calendar.txt”) 2. 제일 첫 줄의 fcount 저장 3. Fday 배열을 동적 생성(fcound+5) 4. 등록된 학사일정을 Fday 배열에 저장 5. 파일 닫기 6. Fday 배열을 하나씩 읽어서 사용자가 입력한 연도와 월이 있다면 그 일정을 출력
out_user	개인일정 출력	1. 개인일정 파일 열기(“user.txt”) 2. Day 배열을 동적 생성(count+1) 3. 등록된 개인 일정을 Day 배열에 저장 4. 파일 닫기 5. Day 배열을 하나씩 읽어서 사용자가 입력한 연도와 월이 있다면 그 일정을 출력

클래스명	ToDo	
함수명	**내용**	**처리절차**
print_calendar	캘린더 출력	1. 7행 4열 배열 선언 2. 윤년을 확인(yoonyear() 함수) 2-1. 윤년이면 2월은 29일 2-2. 그렇지 않으면 2월은 28일 3. 1년은 윤년이 아니므로 365를 기본으로 설정. sum = 365. 1년부터 입력한 년도까지 윤년인지 하나씩 확인 3-1. 윤년이면 366을 더한다 3-2. 그렇지 않으면 365을 더한다 4. 1년 1월부터 입력한 날까지 마지막 날짜를 더한다 5. 1주일은 7일이므로 sum을 7로 나눈 나머지(k)가 입력한 날의 시작 요일이다. 6. 배열에 날짜를 하나씩 출력한다. 단, k가 6이 되면, 즉 토요일이 되면 줄바꿈을 한다.
yoonyear	윤년 확인	1. 윤년 조건 1-1. 4로 나누어 떨어진다. 1-2. 100으로 나누어 떨어지지 않는다. 1-3. 400으로는 나누어 떨어진다. 2. 위 조건을 만족하면 1을 반환 3. else 0을 반환
findschedule	일정 출력	1. 메인메뉴에서 2번 선택 2. Month Year (월 년) 입력 2-1. month가 0 이하이거나 13 이상이면 재귀함수 호출 3. print_calendar() 함수 호출해서 캘린더 출력 4. out_todo() 함수 호출해서 학사일정 출력 5. out_user() 함수 호출해서 사용자일정 출력

◎ 소스 코드

```
//Calendar Management Program
#include <iostream>
#include <string>
#include <fstream>
#include <iomanip>
using namespace std;

struct Day {
	int year, month, day;
	string name;
};
struct Fday {
	int year, month, day, tmonth, tday;
	string name;
};

class Schedule {
protected:
	int count, fcount;
	Day *da;
	Fday *fa;
public:
	Schedule();
	void showschedule(int, int);
	void set_Schedule();
	void out_todo(int, int);
	void out_user(int, int);
};

Schedule::Schedule() {
	ifstream fin("user.txt");
	fin >> count;
	da = new Day[count + 5];
	for (int i = 0; i < count; i++) {
		fin >> da[i].year >> da[i].month >> da[i].day >> da[i].name;
	}
	fin.close();
}


void Schedule::showschedule(int y, int m) {
	int i;
	for (i = 0; i < count; i++) {
		if ((da[i].year == y) && (da[i].month == m)) {
			cout << da[i].month << "/" << da[i].day << "/"<< da[i].year
```

```
<< " " << da[i].name << endl;
                }
        }
}

void Schedule::set_Schedule() {
        int iyear, imonth, iday, num;
        string iname;
        do {
                system("cls");
                cout << endl << "\t [ 1. Register schedule ] " << endl;
                cout << "    Enter date(month day year):";
                cin >> imonth >> iday >> iyear;
                if ((imonth < 0) || (imonth > 13)) set_Schedule();
                da[count].year = iyear; da[count].month = imonth;
da[count].day=iday;
                cout << "    Calendar name(Please enter in ENG): ";
                cin >> iname;

                da[count].name = iname;



                ofstream fout("user.txt",ios::app); //파일이어쓰기
                fout << endl << da[count].month << " "<< da[count].day <<" " <<
da[count].year << " "<< da[count].name;
                count++;
                fout.close();

                cout << endl << "         < Successfully registered! > " << endl;
                cout << "    1. Register another schedule " << endl << "    2.
Return to Main menu " << endl << "    --> ";
                cin >> num;
                if (num != 1 && num != 2) {
                        cout << endl << "    You've entered something wrong.
Please re-enter: ";
                        cin >> num;
                }
                if (num == 2) {
                        system("cls"); break; }
        } while (num == 1);
}

void Schedule::out_todo(int iny, int inm) {
        int i;
        string fname;
        ifstream che("academic_calendar.txt");
```

```
	che >> fcount;
	fa = new Fday[fcount + 5];
	for (i = 0; i < fcount; i++) {
		che >> fa[i].month >> fa[i].day >> fa[i].tmonth >> fa[i].tday >>
fa[i].year;
		che >> fname;
		fa[i].name = fname;
	}
	che.close();
	for (i = 0; i < fcount; i++) {
		if ((fa[i].year == iny) && (fa[i].month == inm)) {
			cout << fa[i].month << "/" << fa[i].day << " - " <<
				fa[i].tmonth << "/" << fa[i].tday << "/" <<
fa[i].year << " " << fa[i].name << endl;
		}
	}
}
void Schedule::out_user(int iny, int inm) {
	int i,j;
	ifstream use("user.txt");
	use >> j;
	da = new Day[count + 1];
	for (i = 0; i < count; i++)
		use >> da[i].month >> da[i].day >> da[i].year >> da[i].name;
	use.close();
	for (i = 0; i < count; i++) {
		if ((da[i].year == iny) && (da[i].month == inm))
			cout << da[i].month << "/" << da[i].day << "/" <<
da[i].year << " " << da[i].name << endl;
	}
}

class ToDo : protected Schedule {
	int max_day[12] = { 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31 };
public:
	ToDo() {};
	void print_calendar(int, int);
	void findschedule();
	int yoonyear(int year) {
		if ((year % 4 == 0) && !(year % 100 == 0) || (year % 400 == 0))
			return 1;
		else return 0;
	}
};

void ToDo::print_calendar(int year, int month) {
```

```
	int i, sum, k;
	char week[7][4] = { {"SUN"}, {"MON"}, {"TUE"}, {"WED"}, {"THU"}, {"FRI"},
{"SAT"} };
	if (yoonyear(year))
		max_day[1] = 29;
	else max_day[1] = 28;


	sum = 365;
	for (i = 1; i < year; i++) {
		if (yoonyear(i)) sum += 366;
		else sum += 365;
	}
	for (i = 0; i < month - 1; i++) {
		sum += max_day[i];
	}
	k = sum % 7;

	cout << endl << endl << "\t       " << month << " / " << year;
	cout << "\n---------------------------------------\n";
	if (month >= 1 && month <= 12) {
		for (int i = 0; i < 7; i++)
			cout << week[i] << "    ";
	}
	cout << "\n---------------------------------------\n";
	for (int j = 0; j < k; j++) {
		cout << "       ";
	}
	for (int i = 1; i <= max_day[month - 1]; i++) {
		cout << fixed << right << setw(3) << i << "    ";
		if (k == 6) {
			k = -1;
			cout << endl;
		}
		k++;
	}
	cout << "\n---------------------------------------\n" << endl;
}


void ToDo::findschedule() {
	int num, ytodo, mtodo;
	do {
		system("cls");
		cout << endl << "\t [ 2. Search schedule ] " << endl;
		cout << "     Month Year: ";
```

```
                cin >> mtodo >> ytodo;
                if ((mtodo <= 0) || (mtodo > 13)) {
                        cout << "     You've entered something wrong. Please
re-enter: ";
                        findschedule();
                }
                print_calendar(ytodo, mtodo); //캘린더
                cout << " [ Academic Calendar ] " << endl;
                out_todo(ytodo, mtodo); //학사일정
                cout << endl << " [ Personal Schedule ]" << endl;
                out_user(ytodo, mtodo); //사용자일정

                cout << endl << "        < Your search is complete! > " << endl;
                cout << "    1. Search again " << endl << "    2. Return to Main
menu " << endl << "    --> ";
                cin >> num;
                if (num != 1 && num != 2) {
                        cout << endl << "    You've entered something wrong.
Please re-enter: ";
                        cin >> num;
                }
                if (num == 2) {
                        system("cls"); break; }
        } while (num == 1);
}


int main() {
        int select;
        Schedule sch;
        ToDo to;
        cout << "    < Launches a calendar management program >" << endl <<
endl << "        ...Loading... ";
        while (1) {
                system("cls");
                cout << endl << endl << "    [ Calendar Management Program ]"
<< endl << endl;
                cout << "    1. Register schedule" << endl << "    2. Search
schedule" << endl << "    3. Exit" << endl;
                cout << endl << "    --> Please enter the number of the Main
menu you want: ";
                cin >> select;
                cout << endl << endl;
                switch (select) {
                case 1:
                        sch.set_Schedule();
```

```
                        break;
                case 2:
                        to.findschedule();
                        break;
                case 3:
                        cout << endl << "  Exit the program." << endl;
                        return 0;
                default:
                        cout << endl << "   You've entered something wrong.
Please re-enter." << endl;
                        system("cls");
                        break;
                }
        }
        return 0;
}
```