

# 소프트웨어 프로젝트 2

20131790

권소현

# 목차

1. 프로젝트 목표 & 2. 프로젝트 내용 (p.3)
3. 알고리즘 구현 (p.4)
4. 프로그램 상세구현 (p.5)
5. 최종 테스트 (p.6~9)
6. 코드 분석 (p.10)
7. 소스 프로그램 (p.11~15)
8. 평가표 (p.16~17)

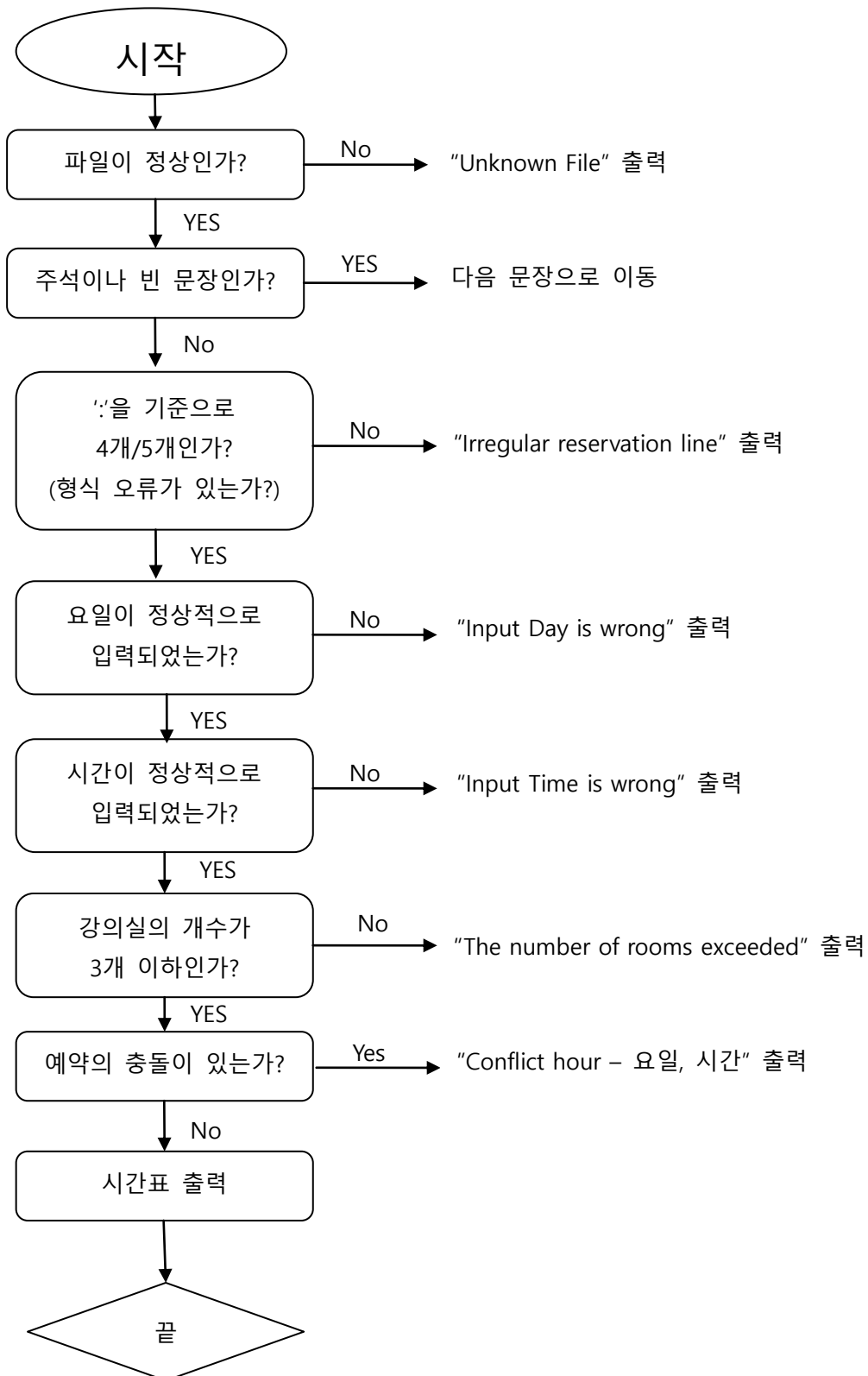
## 1. 프로젝트 목표

주어진 양식에 따라 기술된 호실 예약 파일을 읽어서 각 호실 별로 예약 상황표를 출력하는 프로그램을 작성한다.

## 2. 프로젝트 내용

1. scanner를 이용하여 파일을 읽는다.
2. ":"를 구분자로 하고, 강의실 이름, 요일, 시간, 예약자, 메모 5개를 나누어준다.
3. 복수의 호실을 처리 할 수 있고 (최대 3개), 예약 상황표를 호실 별로 출력한다.
4. 시간의 중복이 있을 경우 경고 / 안내를 표시한다.
5. 잘못된 파일명, 잘못된 규칙으로 기술된 내용에 대해 표시한다.

### 3. 알고리즘 구현



## 4. 프로그램 상세 구현

1. 파일의 데이터는 scanner를 이용하여 읽는다.
2. 한 줄씩 읽어 주석 처리된 문장이나, 빈 문장은 제외시킨다.  
line.startsWith("//")를 사용하여 주석을 거르고, line.length == 0인 경우(빈 문장)를 제외시킨다.
3. 입력된 데이터를 line.split(":")를 사용해 ":"를 구분자로 나누어 String[] tokens에 저장한다.
4. tokens의 0~3까지 비어있는 경우(null)를 형식이 잘못 입력된 경우로 간주한다.
5. 메모가 있는 경우와 없는 경우를 나누어 .trim()을 사용하여 앞뒤 공백을 제거해주고 데이터를 ReservationRecord로 저장한다.
6. 강의실 예약 데이터를 저장할 3차원 배열 rooms, 강의실 이름을 저장하는 room\_name, 강의실의 개수를 나타내는 number\_of\_rooms를 초기화 한다.
7. 저장된 record들을 하나씩 요일이 정상적인지, 시간이 정상적인지 체크한다.
8. Get\_room\_index 메소드를 이용하여 이미 예약된 강의실 이름이라면 강의실이 저장되어있는 주소를 찾아주고, 없다면 강의실에 새로 이름을 저장한다.
9. 그 시간 & 요일에 예약이 없다면 예약 사항을 저장하고, 예약이 이미 있다면 "Conflict hour – 요일 시간"을 출력한다.
8. Display 메소드를 이용하여 강의실 예약 시간표를 출력한다.

## 5. 최종 테스트

### 1. roomreserve-norm.data

```
<terminated> Test (1) [Java Application] C:\Program Files\Java\jre1.8.0_77\bin\javaw.exe (2016. 4.
Room Name :: Room514
  Sun    Mon    Tue    Wed    Thr    Fri    Sat
-----
1
2          Park
3          Kim
4          Kim
5
6
7          Java
8
Room Name :: Room515
  Sun    Mon    Tue    Wed    Thr    Fri    Sat
-----
1
2          Lee    Kim
3
4
5
6
7
8
```

### 2. AutoTest Case1:

```
AutoTest Result
주소 표시줄 자동 완성 표시
Fri Apr 15 16:51:54 KST 2016 by 20131790
Case1: Normal Input
Room Name :: Room514
  Sun    Mon    Tue    Wed    Thr    Fri    Sat
-----
1
2          Park
3          Kim
4          Kim
5
6
7          java
8
Room Name :: Room515
  Sun    Mon    Tue    Wed    Thr    Fri    Sat
-----
1
2          Lee    Kim
3
4
5
6
7
8
End of Case1
```

### 3. AutoTest Case2:

Case2: Normal Input - Out-of-Order  
Room Name :: Room514

	Sun	Mon	Tue	Wed	Thr	Fri	Sat
1			Park				
2				Kim			
3				Kim			
4							
5							
6							
7					java		
8							

Room Name :: Room515

	Sun	Mon	Tue	Wed	Thr	Fri	Sat
1							Kim
2						Lee	Kim
3							
4							
5							
6							
7							
8							

End of Case2

### 4. AutoTest Case3:

Case3: Check String Trimming  
The result must be the same as the normal case above. (The first room only)  
Room Name :: Room514

	Sun	Mon	Tue	Wed	Thr	Fri	Sat
1			Park				
2				Kim			
3				Kim			
4							
5							
6							
7					Java		
8							

End of Case3

## 5. AutoTest Case4-0:

```
*** From now on, Test Reactions on Abnormal Inputs ***
Your Answer (your program's reaction) should be similar to the Correct Answer

Case4-0: Check Hour Conflicts
Correct Answer: Conflict hour -- Wed 2
Your Answer : Conflict hour -- wed 2
Room Name :: Room514
  Sun   Mon   Tue   Wed   Thr   Fri   Sat
-----
1      Park
2      Kim
3      Kim
4
5
6
7
8
End of Case4-0
```

## 6. AutoTest Case4-1:

```
End of Case4-0

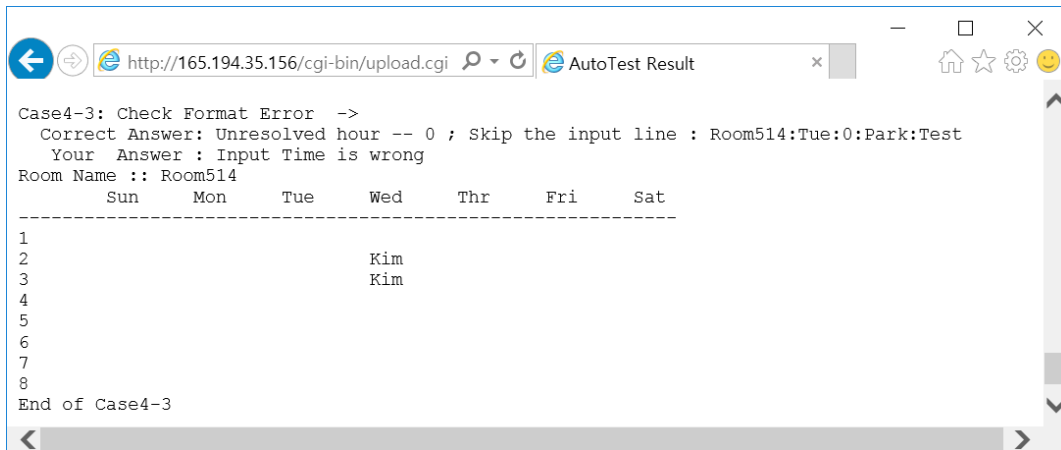
Case4-1: Check Format Error ->
Correct Answer: Irregular reservation line -- ; Skip the input line : Room514::Wed::3:Kim
Your Answer : Irregular reservation line
Room Name :: Room514
  Sun   Mon   Tue   Wed   Thr   Fri   Sat
-----
1      Park
2      Kim
3
4
5
6
7
8
End of Case4-1
```

## 7. AutoTest Case4-2:

```
Case4-2: Check Format Error ->
Correct Answer: Unknown day -- *Wed* ; Skip the input line : Room514:*Wed*:2:Kim:
Your Answer : Input Day is wrong
Room Name :: Room514
  Sun   Mon   Tue   Wed   Thr   Fri   Sat
-----
1      Park
2
3      Kim
4
5
6
7
8
End of Case4-2
```

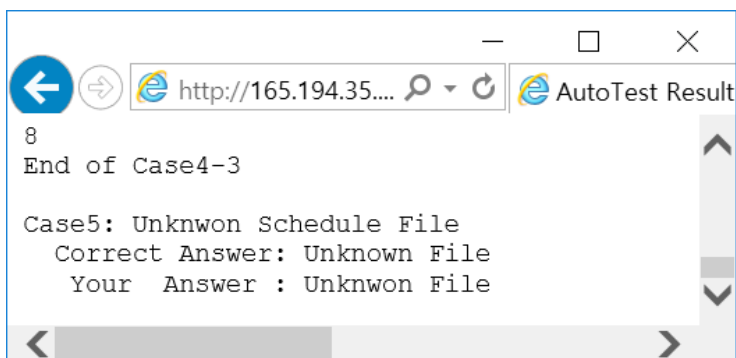


## 8. AutoTest Case4-3:



```
Case4-3: Check Format Error ->
Correct Answer: Unresolved hour -- 0 ; Skip the input line : Room514:Tue:0:Park:Test
Your Answer : Input Time is wrong
Room Name :: Room514
-----
1      Sun      Mon      Tue      Wed      Thr      Fri      Sat
2
3      Kim
4      Kim
5
6
7
8
End of Case4-3
```

## 9. AutoTest Case5:



```
8
End of Case4-3

Case5: Unknwon Schedule File
Correct Answer: Unknown File
Your Answer : Unknwon File
```

## 6. 코드 분석

### 1) RoomReservation 클래스

- 변수

타입	변수명	역할
String[3][8][7]	rooms	강의실 시간표를 저장한다.
String[3]	room_name	강의실의 이름을 저장한다.
int	number_of_rooms	예약 당시의 강의실 개수를 저장한다.

- 메소드

접근 제한자	반환 타입	메소드명	인자	역할
public	void	showReservation	String	파일의 내용을 읽고, 시간표를 출력한다.
public	int	Get_room_index	ReservationRecord	강의실의 위치를 찾는다.
public	void	Display	String[][][], String[], int	시간표를 형식대로 출력한다.

### 2) ReservationRecord 클래스

- 변수

타입	변수명	역할
String	room_num	강의실 이름 저장
String	day	예약 요일 저장
String	name	예약자 저장
String	memo	메모 저장
int	Time	예약 시간 저장

- 메소드

접근 제한자	반환 타입	메소드명	인자	역할
public		ReservationRecord	String[]	입력 받은 데이터를 각각의 변수에 저장
public	boolean	Day_check		요일이 정상적으로 입력되었는지 확인
public	boolean	Time_check		시간이 정상적으로 입력되었는지 확인
public	int	Get_day_index		요일의 위치를 반환

## 7. 소스 프로그램

### RoomReservation.java

```
import java.io.File;
import java.util.Scanner;
import java.util.ArrayList;

public class RoomReservation {
    String[][][] rooms = new String[3][8][7];           // 최대 3개의
    강의실과 입력받을 8x7칸의 강의실 내용을 저장할 String rooms를 만든다.

    String[] room_name = new String[3];                 // 최대
    3개의 강의실이름을 입력받을 String room_name을 만든다.

    int number_of_rooms = 0;                             // 현재
    강의실의 갯수를 알려줄 int형 number_of_rooms를 만든다.

    public void showReservation(String reservationFileName){ // 예약
    사항을 보여줄 메소드를 만든다.

        // 파일의 내용을 불러온다.
        Scanner input = null;
        File file = new File(reservationFileName);
        // 파싱된 예약사항(ReservationRecord)을 저장해줄 ArrayList를 만든다.
        ArrayList<ReservationRecord> records = new
ArrayList<ReservationRecord>();
        String[] data = new String[5];    // 파싱할 정보를 저장해줄 String
array

        try{
            input = new Scanner(file);
        }
        catch(Exception e){
            // 파일을 읽지 못했을 경우
            System.out.println("Unknwon File");
        }
        while(input.hasNext()){
            String line = input.nextLine();

            // 예약 사항 파싱
            if(!line.startsWith("//") && line.length()!=0){
                // 주석처리된 문장이나 빈줄을 제외시킨다.
                String[] tokens = line.split(":");
                // 구분자 ':'를 기준으로 문장을 나눈다.

                if(tokens[0] != null && tokens[1] != null &&
tokens[2] != null && tokens[3] != null ){
                    // 내용이 비어있는 곳이 있는 경우를 제외시킨다.
                    (입력 형식의 오류)
                }
            }
        }
    }
}
```

메모가 모두 있는 경우

records에 저장

ReservationRecord(data);

있는 경우 (메모 입력 x)

records에 저장

ReservationRecord(data);

(입력 형식이 잘못된 경우)

reservation line");

```
if(tokens.length==5) {
    // 강의실 이름, 요일, 시간, 예약자명,

    data[0] = tokens[0].trim();
    data[1] = tokens[1].trim();
    data[2] = tokens[2].trim();
    data[3] = tokens[3].trim();
    data[4] = tokens[4].trim();
    // record 생성 후 arraylist인

    ReservationRecord record = new
    records.add(record);

}
else if(tokens.length == 4) {
    // 강의실 이름, 요일, 시간, 예약자명이

    data[0] = tokens[0].trim();
    data[1] = tokens[1].trim();

    data[2] = tokens[2].trim();
    data[3] = tokens[3].trim();
    // record 생성 후 arraylist인

    ReservationRecord record = new
    records.add(record);

}
else{
    //비어있는 내용이 있을 경우 오류 표시

    System.out.println("Irregular
reservation line");
}
```

```
    }
    }
}
for(int i = 0; i< 3; i++){
    for(int j = 0; j< 8; j++){
        for(int k = 0; k<7; k++){
            this.rooms[i][j][k]=null;
        }
    }
}
for(int i = 0; i<3; i++){
    this.room_name[i] = null;
}
this.number_of_rooms = 0;
```

// records에 있는 record가 요일check, 시간check가 정상일때 , 시간  
중복x, 강의실 3개 초과x일때 저장

```

        for(int i = 0; i<records.size(); i++){
            // 요일의 위치를 찾아 저장
            int day_index = records.get(i).Get_day_index();
            // 강의실이 저장되어있는 위치를 찾아 저장

            if(records.get(i).Day_check()){ // 요일이 정상적으로
입력된 경우

                if(records.get(i).Time_check()){ // 시간이
정상적으로 입력된 경우

                    int room_index =
Get_room_index(records.get(i));

                    if(room_index != -1){ // room이
존재하거나 존재하지 않을 경우 3개를 넘지 않을 때

                        if(rooms[room_index][records.get(i).time][day_index] == null){
// 예약이 비어있으면 내용을 저장한다.

                            rooms[room_index][records.get(i).time][day_index] =
records.get(i).name;

                        }
                        else{
// 예약이 이미 되어있는 경우 충돌
요일 & 시간을 표시한다.

                            System.out.println("Conflict
hour -- "+records.get(i).day + " "+ (records.get(i).time+1));
                        }
                    }
                }
            }
            // 강의실 시간표를 출력해준다.
            Display(rooms, room_name, number_of_rooms);
        }

        // 강의실의 위치를 구하는 메소드
        public int Get_room_index(ReservationRecord record){
            if( number_of_rooms == 0){ // 저장된 강의실이 아예 없을 때
                room_name[number_of_rooms] = record.room_num;
                number_of_rooms++; // 방의 갯수가 1개 증가
                return (number_of_rooms-1); // 이 강의실은
room_name[0]에 저장된 강의실
            }
            else{ // 저장된 강의실이 있을 경우
                for(int i = 0; i<number_of_rooms; i++){
                    // 저장된 강의실이 있고, 자신의 강의실이 있다면
                    저장되어있는 강의실의 위치를 반환

                        if(record.room_num.equals(room_name[i])){
                            return i;

```

```

    }
}
// 나의 강의실이 저장되어있지 않고, 강의실의 갯수가 2개이하라면
if(number_of_rooms<3){
    room_name[number_of_rooms] = record.room_num;
    number_of_rooms++; // 방의 갯수가 증가
    return (number_of_rooms-1);
}
// 나의 강의실이 저장되어있지 않고, 강의실의 갯수가 3개라면
강의실이 초과되었다고 표시
else if( number_of_rooms>=3){
    System.out.println("The number of rooms is
exceeded");
    return -1;
}
}
return -1;
}

// 강의실 시간표를 출력해주는 메소드
public void Display(String [][][] rooms, String[] room_name, int
number_of_rooms ){
    String[] days = {"Sun", "Mon", "Tue", "Wed", "Thr", "Fri", "Sat"};

    for( int i = 0; i<number_of_rooms ; i++){
        // 강의실 이름을 출력
        System.out.println("Room Name :: "+ room_name[i]);
        // 요일 출력
        for( int j = 0; j< days.length; j++){
            System.out.print("\t" + days[j]);
        }
        System.out.println();
        System.out.println("-----");
        // 입력된 내용이 없을경우 '\t'을 입력, 있을 경우 이름 +'\t'입력
        for(int j = 0; j< 8; j++){
            System.out.print(j+1+"\t");
            for(int k = 0; k<7; k++){
                if(rooms[i][j][k]==null){
                    System.out.print("\t");
                }
                else{
                    System.out.print(rooms[i][j][k]+"\t");
                }
            }
            // 한시간을 출력하면 다음시간으로 띄어준다.
            System.out.println("");
        }
    }
}
}

```

## ReservationRecord.java

```
public class ReservationRecord {
    // 저장할 강의실 이름, 요일, 시간, 예약자, 메모를 만들어준다.
    String room_num, day, name, memo;
    int time;

    public ReservationRecord(String[] data){
        // 입력 받은 데이터를 저장
        this.room_num = data[0];
        this.day = data[1].toLowerCase();
        this.time = Integer.parseInt(data[2])-1;
        this.name = data[3];
        this.memo = data[4];
    }

    // 요일이 정상적으로 입력되었는지 확인하는 메소드
    public boolean Day_check(){
        String[] days = {"sun", "mon", "tue", "wed", "thr", "fri", "sat"};
        boolean ret = false;
        for (int i = 0; i<days.length; i++){
            if(day.equals(days[i])){
                ret = true;
            }
        }
        if(ret == false){
            System.out.println("Input Day is wrong");
        }
        return ret;
    }

    // 시간이 정상적으로 입력 (1~8교시) 되었는지 확인하는 메소드
    public boolean Time_check(){
        boolean ret = false;
        if(0<=time && time<=7){
            ret = true;
        }
        if(ret == false){
            System.out.println("Input Time is wrong");
        }
        return ret;
    }

    // 요일의 위치를 반환해주는 메소드
    public int Get_day_index(){
        int index = -1;
        String[] days = {"sun", "mon", "tue", "wed", "thr", "fri", "sat"};
        for (int i = 0; i<days.length; i++){
            if(day.equals(days[i])){
                index = i;
            }
        }
        return index;
    }
}
```

## 평 가 표

평가 항목	학생 자체 평가 (리포트 해당 부분 표시 및 간단한 의견)	평가 (빈칸)	점수 (빈칸)
기본 동작 - autotest 결과화면	각 case 별로 정답이 맞게 출력되었습니다. (p6~9)		
설계 사항 - 설계 착안점 - 사용한 클래스 - 시행 착오 및 해결책	<p>1. 데이터의 저장을 어떻게 할 것인가? 강의실 예약 시간의 저장을 위해서 강의실의 수가 최대 3 개이기 때문에 2 차원 배열을 3 개 만들려고 했다. 하지만 이미 저장된 강의실을 찾거나 예약 사항을 출력 할 때 비효율 적인 것 같아 3 차원 배열을 사용하고 강의실 명을 저장하는 배열을 따로 만들게 되었다.</p> <p>2. 이 과제에서는 강의실 예약 &amp; 출력을 하는 것과 강의실 예약 기록을 저장하는 것이 중요하다고 생각되어 그 것들을 담당하는 두 가지 클래스를 만들었다.</p> <p>3. 처음 시작할 때 강의실 예약 사항을 저장하는 3 차원 배열과, 강의실 이름을 초기화 시키지 않으면 Case1~3 까지는 정상적으로 나왔으나, Case4 에서 4-1 의 데이터가 남아서 진행 되었다. 항상 처음에 초기화가 중요하다는 것을 느꼈다.</p>		



<p>본인 인증</p>	<div data-bbox="529 226 1110 994" data-label="Image"> </div> <p>한밤중에 열심히 코딩하는 제 분신입니다.... 4/12 일 밤 12 시 18 분입니다 미리미리 제가 오래 걸려서 힘들게 했어요..</p>		
<p>기타</p>			
<p>총평/계</p>	<p>여태껏 코딩을 할 때도 그렇고 처음에 프로젝트를 시작 할 때 생각 나는 대로 무조건 코딩을 했더니 결국엔 뒤죽박죽 망쳐버려 며칠 동안 하던 것을 뒤엎었었다. 두 번째로 만들 땐 어떻게 해야 할지 설계를 자세히 세우고 했더니 오타 때문에 생긴 오류나, 자잘한 오류 외에는 프로그램이 잘 돌아갔다. 설계의 중요성을 절실하게 느낀 계기가 되었다.</p>		

\* 학생 자체 평가는 점수에 반영되지 않음.

\* 학생 스스로 자신의 보고서를 평가하면서, 체계적으로 프로젝트를 마무리하도록 유도하는 것이 목적임.