프로그래밍과제 #3

보고서

프로그래밍언어(나)

글로벌미디어학부

20182705 빙기범

0.

- C와 C++의 실행 결과는 사실상 동일함.

- 모든 상황에서 뒤로가기가 가능하며, 잘못된 입력을 받을 경우 다시 입력받음.

- C++는 class를 사용하여 메뉴의 지위에 따라 객체화 하여 함수를 나누어 놓았고, 도구로 사용하는 클래스로부터 상속받음. (HAS-A 관계)

- Manage_room(공간 관리) -> Manage_point(지점 관리) -> Study_reserve_prog(전체 관리)

- All_inq(공간 예약 과정) -> User_tool(공간 조회, ID 예약 조회 및 수정) -> Study_reserve_prog(전체 관리)

```
class Manage_room{
public:
    int add_room(int point);
    int del_room(int point);
    int update_room(int point);
};

class Manage_point : public Manage_room{
public:
    int add_point();
    int del_point();
    int update_point();
};

class All_inq{
public:
    int room_inq_sel(char* id, int point);
    int room_info_res(char* id, int point, int room);
    int date_check(char* id, int point, int room);
    int res_time_num(char* id, int point, int room, int date);
};

class User_tool : public All_inq{
public:
    int inq_res(char id[]);

    int id_inq(char* id);
    int del_res(int num);
};

class Study_reserve_prog : public Manage_point, public User_tool{
public:
    void manager_mode();
    void user_mode();
};
```

- 두 코드 모두 저장소에 입출력하는 양식을 struct로 구현해 놓음.

(POINT 및 MANAGE는 지점 및 공간의 정보를 가지며, RESERVE는 예약 정보를 가짐.)

POINT – 지점 존재 유뮤, 각 공간의 인원 및 존재 유무

MANAGE – 예약 수, 각 지점의 존재 유무

RESERVE – id, 지점, 공간, 날짜, 시간, 인원 (시작 시간만 명시하며 파일에는 1시간 단위로 저장되며, 입력받은 사용 시간이 1시간 이상이면 여러 개의 RESERVE를 저장함.)

C

```
typedef struct _POINT{
    int point_exist;
    int people_num[5];
}POINT;

typedef struct _MANAGE{
    int reserve_num;
    POINT points[6];
}MANAGE;

typedef struct _RESERVE{
    char id[11];
    int point_num;
    int room_num;
    int date;
    int time;
    int people_num;
}RESERVE;
```

C++

```
struct POINT{
    int point_exist;
    int people_num[5];
};

struct MANAGE{
    int reserve_num;
    POINT points[6];
};

struct RESERVE{
    char id[11];
    int point_num;
    int room_num;
    int date;
    int time;
    int people_num;
};
```

1. 초기 화면

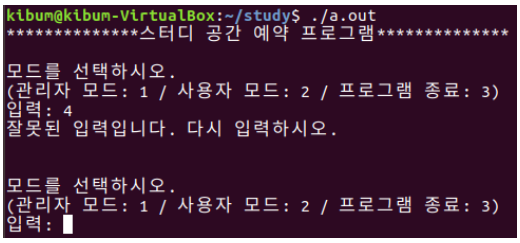
- 모드 선택을 위한 메뉴를 출력하며 입력에 따라 모드를 선택함.

- 1: 관리자 모드 / 2: 사용자 모드 / 3: 프로그램 종료 / else 다시 입력 받음

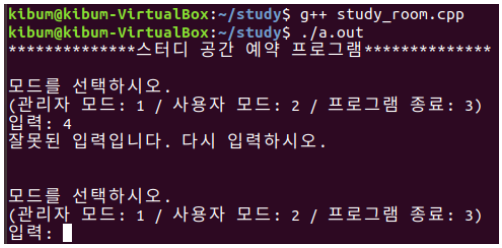
- C에서는 main()에서 실행 되며 1은 manage_mode(), 2는 user_mode() 실행.

- C에서는 main()에서 실행 되며 1은 Study_reserve_prog 객체를 만들고, 그 메소드를, 1은 manage_mode(), 2는 user_mode() 실행.

C



C++



2. 관리자 모드

- 먼저 지점 관리를 위한 선택지를 출력함.

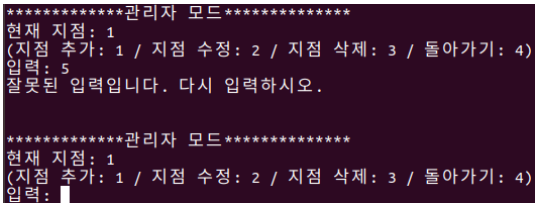
- 1: 지점 추가 / 2: 지점 수정 / 3: 지점 삭제 / 4. 돌아가기 / else 다시 입력 받음.

- 2: 지점 수정 선택시 해당 지점의 공간 관리로 들어감.

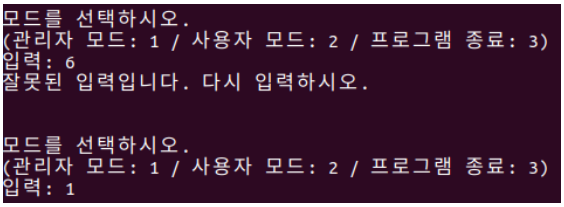
- 모든 판단은 파일에서 공간 정보 및 예약 정보를 읽어와서 판단 후 파일 입력.

- C: manage_mode() / C++: Study_reserve_prog::manage_mode()

C



C++



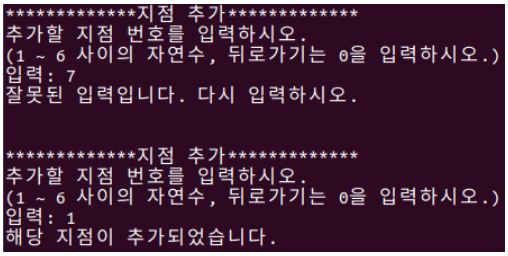
1) 지점 관리

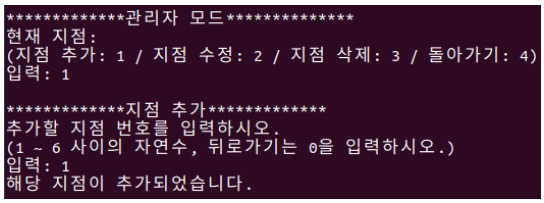
* 추가

- 지점을 추가함.

- 잘못된 입력, 1-6의 자연수가 아닌 입력, 이미 존재하는 지점은 입력으로 받지 않음.

- C: add_point() / C++: Manage_point::add_point()

C


C++


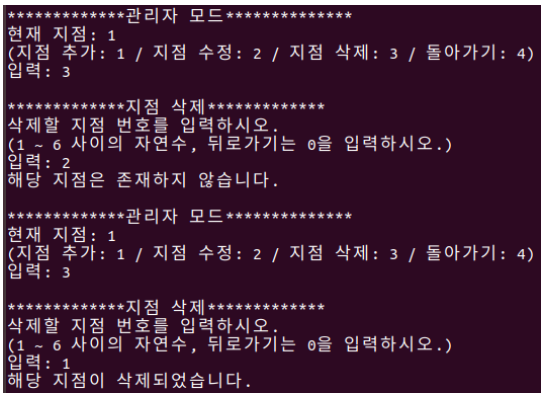
* 삭제

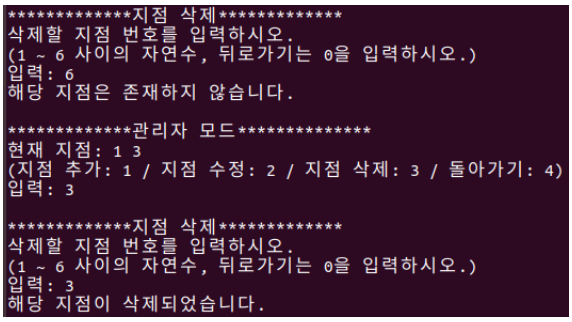
- 지점을 삭제함.

- 잘못된 입력, 1-6의 자연수가 아닌 입력, 존재하지 않는 지점은 입력으로 받지 않음.

- 지점 삭제 시 해당 지점의 예약은 모두 삭제됨.

- C: del_point() / C++: Manage_point::del_point()

C


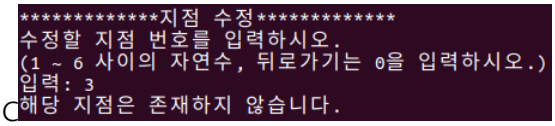
C++


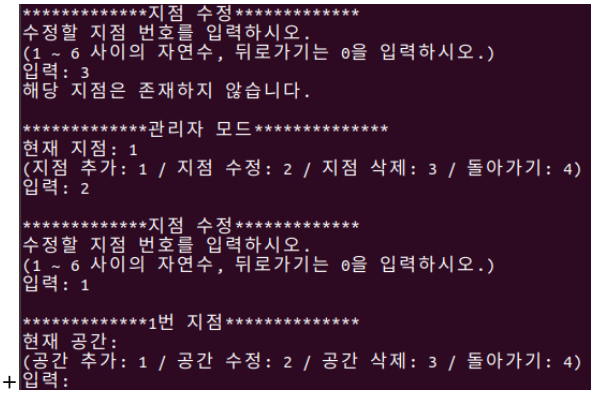
2) 지점 별 스터디 공간 관리

- 지점을 수정. 해당 지점의 공간 관리로 들어감.

- 잘못된 입력, 1-6의 자연수가 아닌 입력, 존재하지 않는 지점은 입력으로 받지 않음.

- C: update_point() / C++: Manage_point::update_point()

C


C++


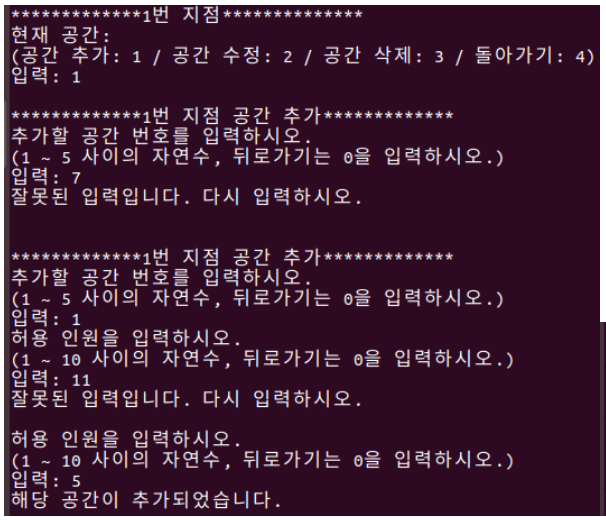
* 공간 추가

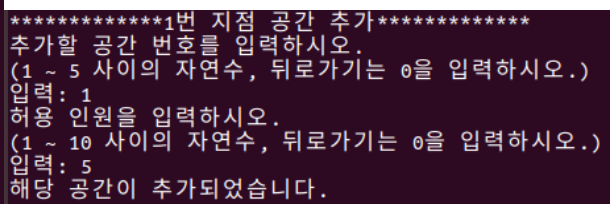
- 공간을 추가함.

- 잘못된 입력, 1-5의 자연수가 아닌 입력, 이미 존재하는 공간은 입력으로 받지 않음.

- 공간 번호를 입력 받고, 사용 인원을 입력 받아 저장함.

- C: add_room() / C++: Manage_room::add_room()





C C++

* 공간 수정

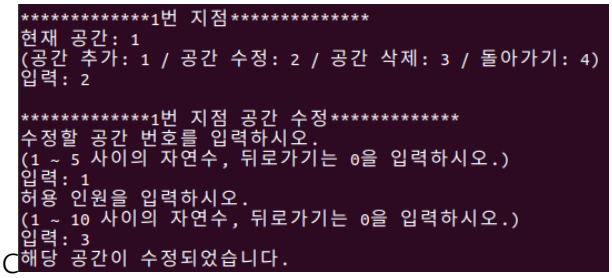
- 공간의 정보를 수정함.

- 잘못된 입력, 1-5의 자연수가 아닌 입력, 존재하지 않는 공간은 입력으로 받지 않음.

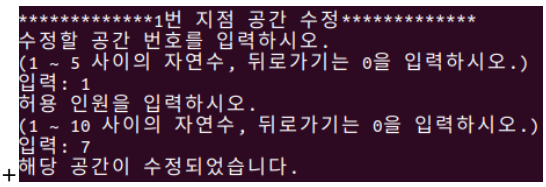
- 공간 번호를 입력 받고, 공간 정보(사용 인원)을 입력 받아 수정함.

- 수정된 공간 정보와 알맞지 않은 예약(사용 인원 초과)은 삭제함.

- C: update_room() / C++: Manage_room::update_room()



C++



* 공간 삭제

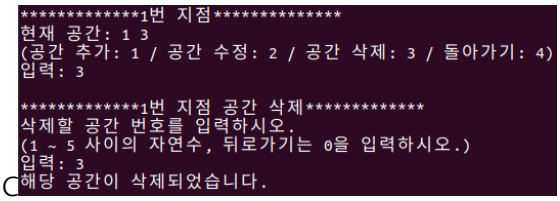
- 공간을 삭제함.

- 잘못된 입력, 1-5의 자연수가 아닌 입력, 존재하지 않는 공간은 입력으로 받지 않음.

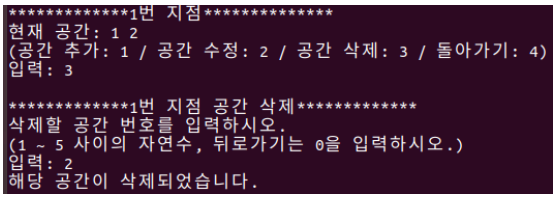
- 삭제된 공간의 예약은 삭제함.

- C: update_room() / C++: Manage_room::update_room()

C



C++



3. 사용자 모드

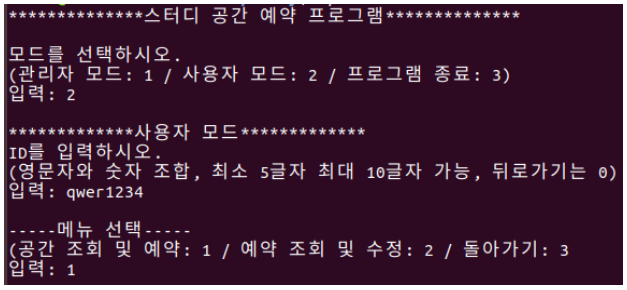
- ID를 입력 받음. ID는 영문자, 숫자 조합, 최소 5글자, 최대 10글자로 맞지 않는 ID는 입력으로 받지 않음.

- ID를 입력 받은 후 메뉴를 선택함.

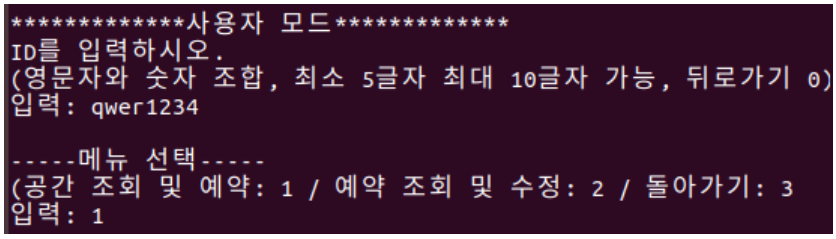
- 1: 공간 조회 및 예약 / 2: 예약 조회 및 수정 / 3. 돌아가기 / else 다시 입력 받음.

- C: user_mode() / C++: Study_reserve_prog::user_mode()

C



C++



* 공간 조회 및 예약

- 지점, 공간을 입력 받아 공간을 조회함. (유효하지 않은 지점, 공간은 다시 입력 받음.)

- 날짜, 시작 시간, 사용 시간, 인원을 추가로 입력 받아 예약함.

- 당일 이후가 아닌 날짜는 다시 입력 받음.

- 8시-22시 사이가 아니거나, 이미 예약이 존재하는 시작 시간 및 사용 시간은 다시 입력 받음.

- 해당 공간의 사용 가능 인원보다 큰 인원은 다시 입력 받음.

- 입력 받은 데이터를 파일에 예약 데이터로 추가함.

- C: inq_res()-지점선택

  - room_inq_sel()-공간 선택, room_info_res()-공간 정보,

    date_check() – 날짜, res_time_num() – 시간 및 인원

- C++: User_tool::inq_res() – All_inq 클래스 메소드

C

```
************공간 조회 및 예약************
예약 가능한 지점: 1

예약할 지점을 선택하시오.
(1 ~ 6 사이의 자연수, 뒤로가기는 0을 입력하시오.)
입력: 1

1번 지점 예약 가능한 공간: 1
예약할 공간을 선택하시오.
(1 ~ 5 사이의 자연수, 뒤로가기는 0을 입력하시오.)
입력: 1

____1번 지점 1번 공간 정보____
사용 가능 인원: 3
예약하기는 1, 뒤로가기는 0을 입력하시오.
입력: 1

예약 일자를 선택하십시오.
(6자리 숫자(YYMMDD), 당일 예약은 불가합니다. 뒤로가기는 0을 입력하시오.)
입력: 220604
해당 날짜는 예약 불가능합니다. 당일 이후부터 예약 가능합니다.

예약 일자를 선택하십시오.
(6자리 숫자(YYMMDD), 당일 예약은 불가합니다. 뒤로가기는 0을 입력하시오.)
입력: 220605

시간 예약. 운영 시간은 8시부터 22시까지입니다. 뒤로가기는 0을 입력하시오.

예약 시작 시간: 8

사용 예정 시간: 2

사용 인원(가능 인원: 3명): 2
예약 되었습니다.
```

C++

```
************공간 조회 및 예약************
예약 가능한 지점: 1

예약할 지점을 선택하시오.
(1 ~ 6 사이의 자연수, 뒤로가기는 0을 입력하시오.)
입력: 1

1번 지점 예약 가능한 공간: 1
예약할 공간을 선택하시오.
(1 ~ 5 사이의 자연수, 뒤로가기는 0을 입력하시오.)
입력: 1

____1번 지점 1번 공간 정보____
사용 가능 인원: 7
예약하기는 1, 뒤로가기는 0을 입력하시오.
입력: 1

예약 일자를 선택하십시오.
(6자리 숫자(YYMMDD), 당일 예약은 불가합니다. 뒤로가기는 0을 입력하시오.)
입력: 220622

시간 예약. 운영 시간은 8시부터 22시까지입니다. 뒤로가기는 0을 입력하시오.

예약 시작 시간: 17

사용 예정 시간: 3

사용 인원(가능 인원: 7명): 2
예약 되었습니다.
```

* 공간 예약 조회 및 수정

- 입력 받았던 아이디로 예약한 정보 조회. 예약한 모든 예약 데이터를 탐색하여 조회.

- 예약 데이터는 한 시간 단위로 출력.

- 임의로 지정된 번호로 예약 수정 및 삭제가 가능함.

- 삭제된 데이터의 정보를 쓰레기 값으로 만듦.

- 수정 선택 시 선택된 예약은 삭제되고, 새로운 예약을 추가할 수 있는 메뉴로 넘어감.

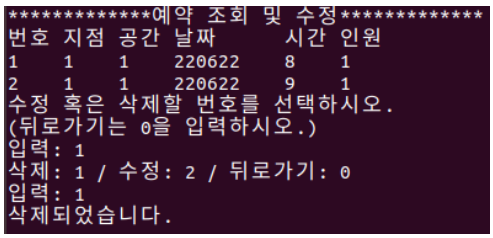
- C: id_inq() - 조회, del_res() - 삭제, int_res() -수정(삭제), 지점 선택

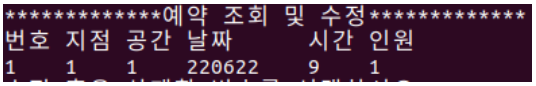
  - room_inq_sel()-공간 선택, room_info_res()-공간 정보,

  date_check() – 날짜, res_time_num() – 시간 및 인원

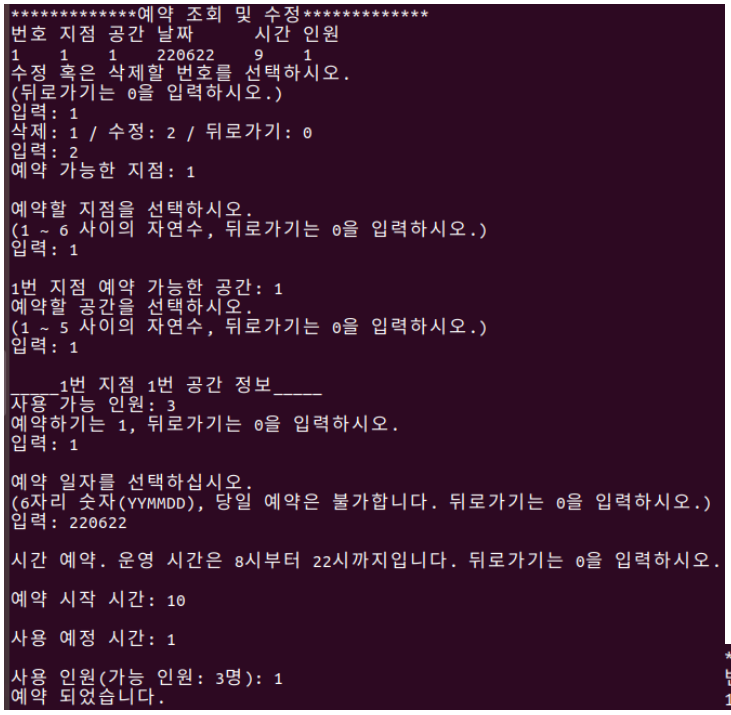
- C++: User_tool:: id_inq(), del_res(), inq_res() – All_inq 클래스 메소드

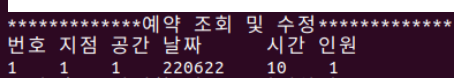
C_삭제





C_수정





C++_삭제

```
*************예약 조회 및 수정*************
번호 지점 공간 날짜      시간 인원
1    1    1    220622    17    2
2    1    1    220622    18    2
3    1    1    220622    19    2
수정 혹은 삭제할 번호를 선택하시오.
(뒤로가기는 0을 입력하시오.)
입력: 2
삭제: 1 / 수정: 2 / 뒤로가기: 0
입력: 1
삭제되었습니다.
```

```
*************예약 조회 및 수정*************
번호 지점 공간 날짜      시간 인원
1    1    1    220622    17    2
2    1    1    220622    19    2
수정 혹은 삭제할 번호를 선택하시오
```

C++_수정

```
************예약 조회 및 수정************
번호 지점 공간 날짜      시간 인원
1    1    1    220622    17    2
2    1    1    220622    19    2
수정 혹은 삭제할 번호를 선택하시오.
(뒤로가기는 0을 입력하시오.)
입력: 2
삭제: 1 / 수정: 2 / 뒤로가기: 0
입력: 2
예약 가능한 지점: 1

예약할 지점을 선택하시오.
(1 ~ 6 사이의 자연수, 뒤로가기는 0을 입력하시오.)
입력: 1

1번 지점 예약 가능한 공간: 1
예약할 공간을 선택하시오.
(1 ~ 5 사이의 자연수, 뒤로가기는 0을 입력하시오.)
입력: 1

____1번 지점 1번 공간 정보____
사용 가능 인원: 7
예약하기는 1, 뒤로가기는 0을 입력하시오.
입력: 1

예약 일자를 선택하십시오.
(6자리 숫자(YYMMDD), 당일 예약은 불가합니다. 뒤로가기는 0을 입력하시오.)
입력: 220622

시간 예약. 운영 시간은 8시부터 22시까지입니다. 뒤로가기는 0을 입력하시오.

예약 시작 시간: 20

사용 예정 시간: 1

사용 인원(가능 인원: 7명): 4
예약 되었습니다.
```

```
*************예약 조회 및 수정*************
번호 지점 공간 날짜      시간 인원
1    1    1    220622    17    2
2    1    1    220622    20    4
수정 혹은 삭제할 번호를 선택하시오
```

4. 프로그램 종료

C

```
모드를 선택하시오.
(관리자 모드: 1 / 사용자 모드: 2 / 프로그램 종료: 3)
입력: 3
프로그램을 종료합니다.
kibum@kibum-VirtualBox:~/study$
```

C++

```
모드를 선택하시오.
(관리자 모드: 1 / 사용자 모드: 2 / 프로그램 종료: 3)
입력: 3
프로그램을 종료합니다.
kibum@kibum-VirtualBox:~/study$
```

C언어와 OOP언어의 차이

스터디 공간 프로그램은 많은 기능이 서로 관계를 가지고 있는 프로그램이다. C언어 코딩 시, 나름 복잡하고 많은 양의 코딩을 필요로 했기 때문에, 많은 양의 코드가 줄지어 있어서 Readability가 굉장히 떨어졌다. 함수 하나하나의 구조를 비슷하게 코딩했기 때문에, 코딩 도중 함수를 헷갈리기 십상이었으며, 한 파일로 제출하라는 조건 때문에 이는 더 악화되었다. 이리저리 널려 있는 함수를 사용하기 위해, 코드 위와 아래를 반복해야 했으며, 수정 또한 불편했다. 코드를 직접 짠 입장이지만, 다른 사람이 이 코드를 이해하기 위해서는 굉장한 시간이 필요할 것 같다.

사실, C++로 코딩한다고 해서 굉장하게 보기가 좋아지지는 않았다. 함수의 선언과 정의를 한 파일에 구현해야 하는 것은 그대로이기 때문이다. 하지만 class를 통해 함수 간의 관계를 명시해 줄 수 있었다. 같은 class에 두어 이들이 같은 메뉴에서 선택됨을 명시하여, 그들의 지위가 비슷함을 보여줄 수 있었다. 또, 각 함수에서 사용하는 다른 함수들을 class로 묶어 상속하여, 소유 개념의 HAS-A 관계를 만들 수 있었다. 개인적으로 이를 바탕으로 코드 구조에 대해 정리하는 것에 도움이 되었으며, 처음 보는 사람도 이 코드를 봤을 때 클래스들의 관계를 보고, 어느 정도 코드 작동 구조를 이해할 수 있을 것으로 생각한다. 추가로 C++에서 지원하는 string은 char 배열로 이루어진 문자열보다 훨씬 사용이 편리했다. 하지만 컴파일 시 C++의 속도가 C언어에 비해 조금 느린 것처럼 느껴졌다.