[ 객체지향 프로그래밍 ]

[ 제어문 및 동적할당 응용 과제 ]

|  |  |
| --- | --- |
| 과제 체크리스트 | 확인 |
| 1. 이 과제물은 내가 직접 연구하고 작성한 것이다. | 예 |
| 2. 인용한 모든 자료(책, 논문, 보고서, 인터넷 자료 등)의 출처를 밝혔다. | 예 |
| 3. 정확한 출처 제시나 사용 허락 없이 다른 사람의 아이디어를 가져오지 않았다. | 예 |
| 4. 이 과제물을 다른 사람으로부터 받거나 구매하여 제출하지 않았다. | 예 |
| 5. 이 과제의 결과물에 나의 학번 이름이 출력되어 있다. | 예 |

담당 교수 : 지정희 교수님

학 번 : 201711430

이 름 : 차준혁

제출일 : 2021년 3월 26일

|  |  |
| --- | --- |
| 1 | **문제정의 및 분석** |

영화 정보를 담고 있는 파일을 읽어서 영화관 관련 정보를 동적으로 생성한 다음, 이 동적으로 생성한 영화 정보를 이용하여 사용자에게 간단한 영화 예매, 취소, 확인을 도와주는 영화관 좌석 예약 시스템을 만들어야 하는 문제입니다.

영화와 좌석 정보를 담고 있는 txt파일을 읽어서 영화관 관련 정보를 동적 생성해야 합니다. 저는 구조체를 원소로 가지는 배열을 동적 할당하여 먼저 txt파일이 담고 있는 영화관 관련 정보를 동적 생성했습니다. 구조체는 영화 이름, 좌석의 행과 열, 예약 가능한 좌석 수를 뜻하는 변수를 가지게 만들었고 예약 가능한 좌석 수는 좌석의 행과 열을 곱한 수로 초기화 했습니다. 영화의 숫자만큼 구조체를 배열을 이용해 생성했습니다. 각 구조체가 가지고 있는 영화정보인 좌석 행과 열의 크기를 이용하여 면은 영화의 종류를, 행은 영화관의 좌석 행의 수를, 열은 영화관의 좌석 열의 수를 뜻하는 bool형 3차원 배열을 동적 할당했습니다. 동적 할당과 동시에 false로 배열의 모든 원소를 초기화했습니다.

영화 예매, 예매 취소, 예매 확인, 종료라는 총 4가지 메뉴로 나누어 사용자에게 선택할 메뉴번호를 입력 받고 do while문으로 종료 메뉴를 선택할 때까지 메인 메뉴로써 출력하게 하였고 각 메뉴의 기능을 여러가지 함수를 정의하여 구현했습니다.

영화 예매를 선택하면 생성했던 구조체를 통해 현재 영화 이름, 예약 가능한 좌석수/전체 좌석수 정보를 리스트 형태로 출력합니다. 이때, 예약 가능한 좌석 수가 0이라면 매진된 상태이므로 조건문을 이용하여 매진은 따로 출력해줍니다. 출력된 리스트에서 영화를 선택하면 예매할 좌석을 선택할 수 있게 선택할 영화의 번호를 입력 받고 현재 좌석 정보를 보여줍니다. 삼항 연산자를 이용하여 예약 가능한 좌석을 뜻하는 False를 배열의 원소로 가지면 ○로, 예약된 좌석을 뜻하는 True를 배열의 원소로 가지면 ●로 현재 좌석 정보를 출력합니다. 예약할 좌석의 행과 열을 char형으로 입력 받고 형변환을 활용해서 입력 받은 좌석 정보의 배열에 접근하여 예약이 가능한 자리면 false -> true로 원소를 바꿔주고 구조체의 예약 가능한 좌석수를 -1해줍니다. 예약이 된 자리면 이미 예약이 된 자리라고 출력해주고 범위 밖의 입력에 대해서는 조건문으로 존재하지 않는 좌석이라고 출력해서 예외처리를 해줍니다. 예약을 할 때, 중복되지 않는 예약번호를 부여해야 하는데 저는 이를 동적할당한 3차원 배열의 index를 활용했습니다. 모든 좌석은 3차원 배열의 index로 접근할 수 있고 각 배열의 index는 당연히 중복이 되지 않습니다. 단순히 배열의 index만 조합하여 예약번호를 부여하기에는 사용자가 예약번호의 생성원리를 다소 쉽게 파악할 수 있으므로 형변환을 활용하여 각 int형 index에 정해진 char형 문자(‘A’, ‘C’, ‘D’)를 더하여 중복을 피하면서 각각의 사용자가 쉽게 추리하기 힘들게 예약번호로 문자로 3개 부여했습니다.

예매 취소를 선택하면 예약할 때 받았던 예약 번호를 입력 받는데 예매번호를 문자 3개로 부여했으므로 입력도 문자 3개로 받습니다. 형변환을 활용하여 int형으로 바꾸고 배열의 index로 이용하여 배열의 원소에 접근합니다. 접근한 배열의 원소가 true라면 해당 예약번호의 예약이 존재하는 것이므로 예매가 취소되었다는 출력과 함께 false를 대입하고 구조체의 예약 가능한 좌석 수에 ++연산을 합니다. 접근한 배열의 원소가 false라면 예약번호가 존재하지 않는 것이므로 예약번호가 존재하지 않는다고 출력한 후 메인 메뉴로 돌아갑니다.

예매 확인을 선택하면 예약 번호를 또 입력 받는데 예매번호를 문자 3개로 부여했으므로 입력도 문자 3개로 받습니다. 마찬가지로 형변환을 활용하여 int형으로 바꾸고 배열의 index로 이용하여 배열의 원소에 접근합니다. 조건문을 이용하여 배열의 원소가 true일 때 해당 영화의 정보인 영화 이름을 구조체를 활용하여 출력하고 예매한 좌석도 형변환을 활용해서 출력합니다. 또한, 반복문과 조건문을 활용하여 ‘배열의 원소가 true일 때’가 아닌 예약번호의 배열 원소일 때만 ○ 대신 ☆을 예약한 좌석 정보의 위치로 출력합니다. 영화 예매 때처럼 원소가 true일 때 ☆로 출력하면 입력한 예약번호의 좌석이 아닌 다른 예약된 좌석도 같이 표현될 수 있으므로 예약번호에 해당하는 좌석만 출력하려면 조건문을 활용하여 예약번호의 배열 원소일 때만 ○ 대신 ☆을 예약한 좌석 정보의 위치로 출력합니다.

종료 메뉴를 선택하면 delete를 통해 동적 할당 받은 3차원 배열, 구조체 배열 순으로 메모리를 삭제해주고 프로그램을 종료합니다.

|  |  |
| --- | --- |
| 2 | **주요 소스코드 설명** |

struct MovieData {

string MovieName;

int row, col, availableNum;

};

🡺영화 정보를 담을 구조체입니다.

MovieName : 영화제목, row와 col = 좌석의 행과 열, availableNum = 예약가능한 좌석 수

struct MovieData\* TheaterSize(int& MovieNum) {

ifstream fin("moviedata.txt");

if (!fin.is\_open()) {

cerr << "파일 오픈 실패";

exit(0);

}

fin >> MovieNum;

MovieData\* md = new MovieData[MovieNum];

//delete 체크

while (!fin.eof()) {

for (int i = 0; i < MovieNum; i++) {

fin >> md[i].MovieName;

fin >> md[i].row;

fin >> md[i].col;

md[i].availableNum = (md[i].row)\*(md[i].col);//예약 가능한 좌석 수 초기화

}

}

return md;

}

🡺앞에서 선언한 구조체의 배열을 동적 할당하고 초기화하기 위한 함수입니다.

구조체를 가리키는 포인터를 반환하고 int& MovieNum이라는 참조변수를 입력인자로 받습니다.

MovieNum변수에 파일로부터 영화의 개수를 제일 먼저 읽어서 대입하고 영화의 개수만큼 구조체를 배열의 원소로 가지는 구조체 배열을 동적 할당합니다. 그리고 파일의 끝까지 읽어서 각 영화의 정보를 MovieName(영화이름), row,col(좌석의 행과 열), availableNum(예약가능한 좌석 수) 각 구조체 안의 변수에 대입합니다. availableNum도 각 영화의 행과 열을 곱해 초기화합니다.

void deleteMovieData(struct MovieData\* (&md)) {

delete[]md;

md = nullptr;

}

🡺앞에서 선언한 동적 할당한 배열을 delete를 통해 메모리를 삭제해서 메모리 누수를 막기 위한 함수입니다.

bool\*\*\* loadSeats(struct MovieData\* (&md), int& MovieNum) {

bool\*\*\* seats = new bool\*\* [MovieNum];

for (int i = 0; i < MovieNum; i++) {

seats[i] = new bool\* [md[i].row];

for (int j = 0; j < md[i].row; j++) {

seats[i][j] = new bool[md[i].col];

}

}

//delete해야됨

for (int n = 0; n < MovieNum; n++) {

for (int i = 0; i < md[n].row; i++) {

for (int j = 0; j < md[n].col; j++) {

seats[n][i][j] = false;

}

}

}

return seats;

}

🡺구조체 배열과 영화의 개수를 입력인자로 받아서 영화마다 각각의 좌석을 표현하는 배열을 만들기위해 3차원배열을 동적 할당하는 함수입니다. Bool형 3차원 배열을 반환하고 false로 예약이 가능한 상태라는 뜻으로 초기화합니다.

void deleteSeats(struct MovieData\* (&md), bool\*\*\* (&seats), const int MovieNum) {

for (int i = 0; i < MovieNum; i++) {

for (int j = 0; j < md[i].row; j++)

delete[] seats[i][j];

delete[] seats[i];

}

delete[] seats;

seats = nullptr;

}

🡺앞에서 동적 할당한 3차원 배열을 delete를 통해 메모리를 삭제해서 메모리 누수를 막기 위한 함수입니다. 입력인자로 구조체를 가리키는 포인터와 3차원 배열, 영화의 개수를 받습니다.

void showMovieList(struct MovieData\* (&md), int &select, const int MovieNum) {

cout << "\n=================영화 예매==================\n";

for (int i = 0; i < MovieNum; i++) {

string name = md[i].MovieName;

int namelen =(38 - name.length());

cout << (i + 1) << ") " << md[i].MovieName;

if (md[i].availableNum == 0) {

cout << setfill(' ') << setw(namelen+3);

cout << "매진" << endl;

}

else {

cout << setfill(' ')<<setw(namelen);

cout << md[i].availableNum << "/" << (md[i].row) \* (md[i].col) << endl;

}

}

cout << endl << "예매할 영화를 선택하세요 : ";

cin >> select;

}

🡺영화 예매라는 메뉴를 선택했을 때 영화 정보를 나타내는 리스트를 출력하고 영화 선택을 입력 받기 위한 함수입니다. 입력 인자로 구조체를 가리키는 포인터와 선택한 영화종류를 입력 받기위한 참조 변수 select, 영화개수를 뜻하는 변수인 MovieNum이 있습니다.

for문을 통해 영화 수(MovieNum)만큼 한 줄씩 영화 정보를 출력하는데 각 영화의 제목의 길이와 상관없이 간격을 맞추기 위하여 int형 namelen변수를 선언하여 (38 - name.length())을 대입합니다. namelen의 크기만큼 setw()를 이용하여 출력하는 공간을 확보하고 빈 공간을 ‘ ’인 빈칸을 대입하여 채웁니다. 그리고 예약 가능 좌석 수/전체 좌석수 정보를 출력하는데 이때 조건문을 사용하여 availableNum(예약 가능 좌석 수)가 0일 때 매진으로 출력합니다. 3차원 배열의 index로 쓰기위해 select변수(영화 구분을 위해)로 입력을 받습니다.

void showSeats(bool\*\*\* (&seats), const int row, const int col, int select) {

cout << endl;

for (int i = 0; i < col; i++)

cout << '\t' << i + 1;

cout << "\n\n";

for (int i = 0; i < row; i++) {

cout << static\_cast<char>('A' + i) << '\t';

for (int j = 0; j < col; j++) {

cout << (seats[select][i][j] ? "●" : "○") << '\t';

}

cout << endl;

}

}

🡺영화를 선택하고 난 뒤 좌석 현황 정보를 보여주기 위한 함수입니다. 3차원 배열, 3차원 배열의 index로 쓸 select, row, col을 입력인자로 받습니다. Static\_cast<char>이라는 형변환을 통해 배열의 행 정보를 좌측에 표현하고 삼항 연산자와 for문을 통해 배열의 원소가 true일 때 ●로 출력하고 false일 때 ○로 출력하여 좌석 현황 정보를 표현합니다.

void reserveSeats(struct MovieData\* (&md), bool\*\*\* (&seats), int select) {

cout << "\n예매할 좌석을 선택하세요(ex) A 1, B 2) : ";

char x, y;

cin >> x >> y;

char MDRow, MDCol;//영화관 좌석의 행과 열을 뜻함

MDRow = static\_cast<char>('A' + (md[select].row) - 1);

MDCol = static\_cast<char>('1' + (md[select].col) - 1);

if (x < 'A' || x > MDRow) {

cout << "\n죄송하지만 존재하지 않는 좌석입니다." << endl;

}

else if (y < '1' || y > MDCol) {

cout << "\n죄송하지만 존재하지 않는 좌석입니다." << endl;

}

else if (seats[select][x - 'A'][y - '1']) {

cout << "\n이미 예약된 자리입니다." << endl;

}

else {

seats[select][x - 'A'][y - '1'] = true;

(md[select].availableNum)--;

cout << "\n예약이 완료되었습니다." << endl;

cout << "예약번호는 " << static\_cast<char>(select + 'A');

cout << static\_cast<char>(x - 'A' + 'C');

cout << static\_cast<char>(y - '1' + 'D') << " 입니다." << endl;

}

}

🡺좌석을 선택하고 예약하고 예약번호를 부여해서 예약을 완료하는 함수입니다. 구조체를 가리키는 포인터와, 3차원 배열, 영화 구분을 위한 int형 변수 select를 입력인자로 받습니다. char형으로 예매할 좌석 자리를 입력 받고 조건문을 통해 MDRow, MDCol과 비교하여 영화관 좌석 배열 범위 밖의 입력을 제한합니다. 여기서 MDRow와 MDCol은 각 영화마다 다른 행과 열의 개수를 대입해서 비교하기 위한 char형변수인데 static\_cast<char>을 통한 형변환으로 영화관 좌석 배열의 행과 열의 끝의 기준을 정할 수 있습니다. 조건문을 통해 예약이 가능한 자리이면 true를 대입하고 그 영화의 이용가능한 좌석 수를--합니다. ((md[select].availableNum)--;) 그리고 예약번호를 출력하는데 예약번호는 3차원 배열의 index인 select, x – ‘A’, y –‘1’ 을 이용해서 중복을 피하고 각각 ‘A’, ‘C’, ‘D’를 더하고 char형으로 형변환을 하여 3개의 문자로 출력합니다.

void cancelSeats(struct MovieData\* (&md), bool\*\*\* (&seats)) {

char c1, c2, c3;

cout << "취소할 영화의 예약번호를 입력해주세요(ex) 예약번호가 AEF라면 입력: AEF): ";

cin >> c1 >> c2 >> c3;

int dNum1 = static\_cast<int>(c1 - 'A');

int dNum2 = static\_cast<int>(c2 - 'C');

int dNum3 = static\_cast<int>(c3 - 'D');

if (seats[dNum1][dNum2][dNum3]) {

seats[dNum1][dNum2][dNum3] = 0;

(md[dNum1].availableNum)++;

cout << "\n예매가 취소되었습니다." << endl;

}

else {

cout << "\n예약번호가 존재하지 않습니다." << endl;

}

}

🡺예매한 좌석을 취소하는 함수입니다. 구조체를 가리키는 포인터와 3차원 배열이 입력 인자입니다. 예약번호를 char형 문자3개로 부여했기 때문에 취소할 예약번호를 받을 때도 char형으로 문자 3개를 입력 받습니다. char형으로 받은 문자 각각에서 예약번호를 부여할 때 더한 ‘A’, ‘C’, ‘D’를 빼고 int형으로 형변환을 하여 3차원 배열의 index로 활용합니다. 조건문을 통해서 확인한 배열의 원소가 true일 때 해당 예약번호의 예약이 존재하는 경우이므로 false를 대입해서 해당 영화의 예매를 취소하고 예매가 취소되었다는 문자열을 같이 출력합니다. 이때 예약 가능한 좌석수인 availableNum도 ++합니다. 예약번호가 존재하지 않는 나머지 경우에는 예약번호가 존재하지 않는다고 출력합니다.

void checkSeats(struct MovieData\* (&md), bool\*\*\* (&seats)) {

char c1, c2, c3;

cout << "\n=================예약 확인==================\n";

cout << "예약번호를 입력하세요(ex) 예약번호가 AEF라면 입력: AEF): ";

cin >> c1 >> c2 >> c3;

int searchNum1 = static\_cast<int>(c1 - 'A');

int searchNum2 = static\_cast<int>(c2 - 'C');

int searchNum3 = static\_cast<int>(c3 - 'D');

if (seats[searchNum1][searchNum2][searchNum3]) {

cout << "예매하신 영화 : " << md[searchNum1].MovieName << endl;

cout << "예매하신 좌석 : " << static\_cast<char>(searchNum2 + 'A') << (searchNum3 + 1) << endl;

cout << "\n=====================영화 예매 좌석====================\n";

cout << endl;

for (int i = 0; i < md[searchNum1].col; i++)

cout << '\t' << i + 1;

cout << "\n\n";

for (int i = 0; i < md[searchNum1].row; i++) {

cout << static\_cast<char>('A' + i) << '\t';

for (int j = 0; j < md[searchNum1].col; j++) {

cout << ( ( (i== searchNum2)&&(j==searchNum3) )? "☆" : "○") << '\t';

}

cout << endl;

}

system("pause");

}

else {

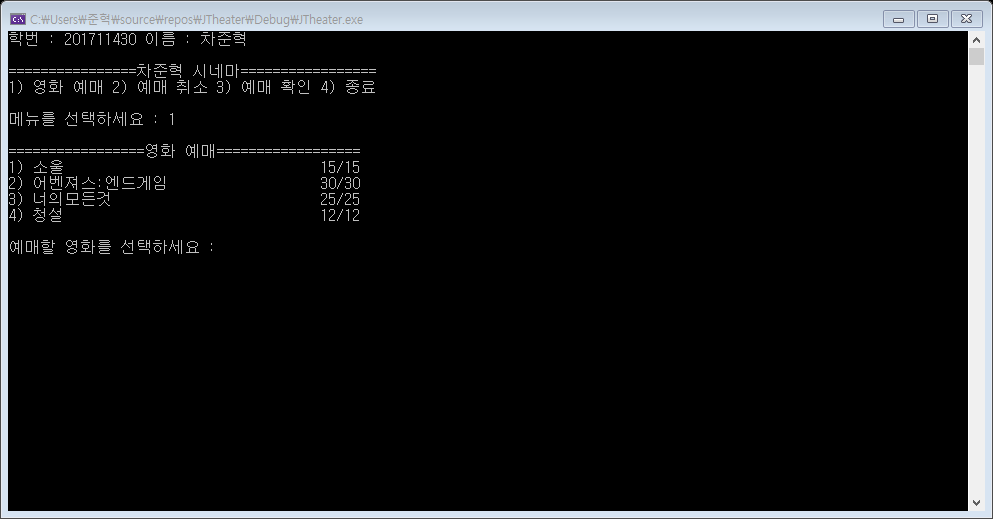
cout << "\n예약번호가 존재하지 않습니다." << endl;

}

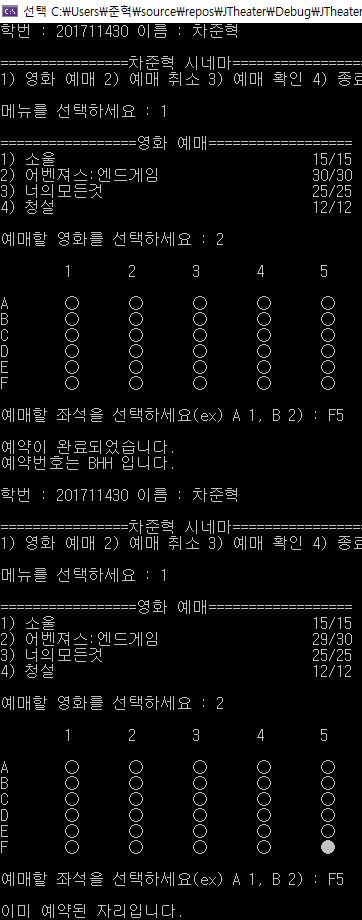
}

🡺예매한 좌석을 확인하는 함수입니다. 구조체를 가리키는 포인터와 3차원 배열이 입력 인자입니다. 예약번호를 char형 문자3개로 부여했기 때문에 확인할 예약번호를 받을 때도 char형으로 문자 3개를 입력 받습니다. char형으로 받은 문자 각각에서 예약번호를 부여할 때 더한 ‘A’, ‘C’, ‘D’를 빼고 int형으로 형변환을 하여 3차원 배열의 index로 활용합니다. 조건문을 통해서 확인한 배열의 원소가 true일 때 해당 예약번호의 예약이 존재하는 경우이므로 예매한 영화의 제목과 좌석을 형변환과 구조체를 통하여 접근해서 출력합니다. 그리고 영화 예매 좌석의 위치를 보여주기 위해서 삼항 연산자를 사용하여 for문에서 (i== searchNum2)와 (j==searchNum3)가 둘 다 만족할 때(예약번호에 해당하는 좌석이라는 뜻) ☆으로 출력합니다. 만족하지 않을 때 는 ○를 출력하여 예약번호의 좌석이 아님을 표현합니다. 조건문을 통해서 확인한 배열의 원소가 true가 아닐 때는 예약번호가 존재하지 않는 경우이므로 존재하지 않는다고 출력합니다.

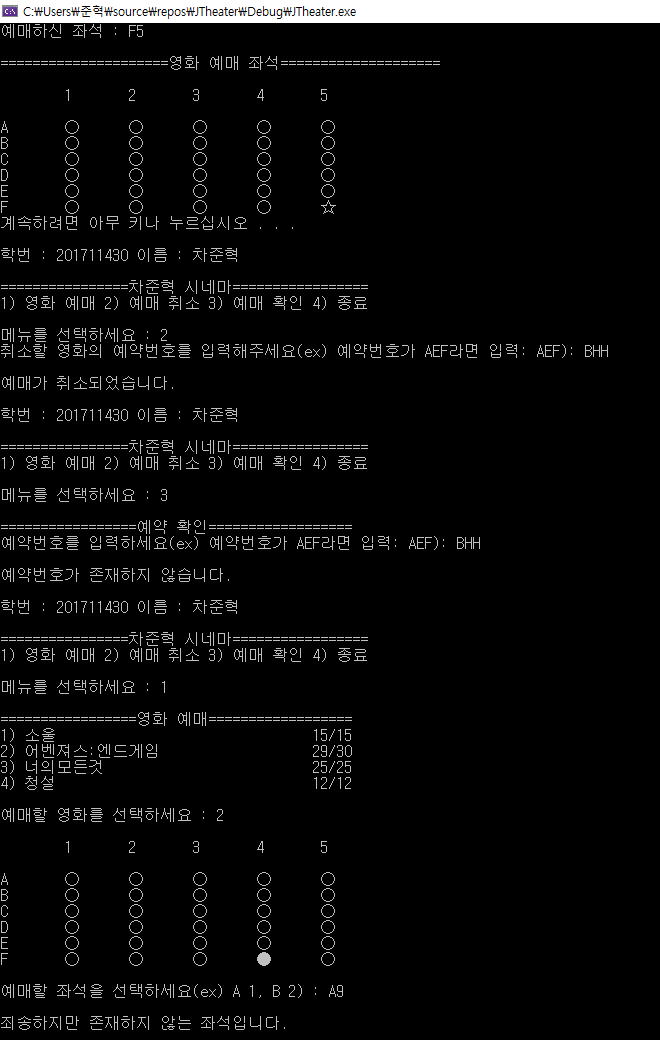
|  |  |
| --- | --- |
| 3 | **실행결과** |

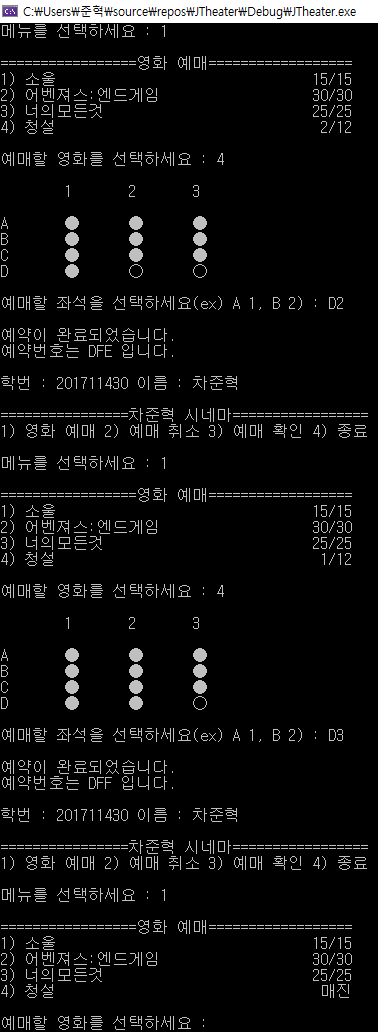


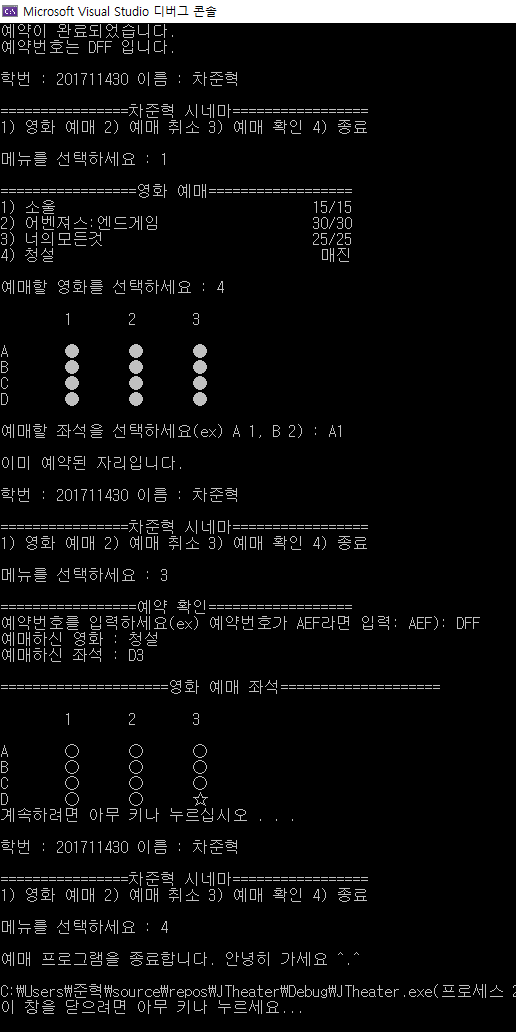
🡺첫 영화 예매 메뉴를 선택했을 때 화면입니다.

🡺예매를 하고 예약번호를 출력했습니다. 예약번호BHH자리를 또 예약하려 하니 이미 예약된 자리라고 출력 되는 걸 볼 수 있습니다.

🡺F4자리에 예매를 한 번 더 했고 메뉴 3)예매 확인을 통해 BHH를 입력하니 F4 F5 둘 다 예약된 상태임에도 예약번호를 입력한 F5만 ☆로 출력되었습니다.

🡺예약번호가 BHH인 좌석을 취소하고 다시 예매 확인으로 확인해보면 존재하지 않는 것을 볼 수 있습니다. 그리고 예매확인을 선택하고 범위를 넘어서는 A9입력을 했을 때 존재하지 않는 좌석이라고 출력합니다.

🡺영화가 다 매진된 경우도 위와 같이 출력됨을 알 수 있습니다.

🡺전좌석이 예약된 청설이라는 영화에서 예매 확인메뉴를 통해 예매번호를 입력하면 입력한 예매번호의 좌석만 보이는 것을 알 수 있습니다. 그리고 메뉴로 돌아가 종료를 선택하면 프로그램이 종료됩니다.

|  |  |
| --- | --- |
| 4 | **느낀 점 및 토의사항** |

정말 간단한 기능을 하는 영화관 예매 프로그램을 만들어봤는데 생각보다 오래 걸리고 제한이나 신경 써주어야 할 부분이 많았습니다. 실제로 우리가 흔하게 쓰는 휴대폰 어플과 같은 영화관 예매 프로그램을 만든다면 완벽한 프로그램을 위해 많은 노력과 시간이 필요하겠구나 하고 느꼈습니다. 이 문제에서는 예매 취소나 확인을 할 때 예약 번호만 입력해서 해당 예약번호의 예약이 존재할 경우 예매를 취소하거나 확인하는 것 만을 요구하지만 사용자가 무작위로 예약번호를 입력하여 자칫 다른 사용자가 예매한 좌석을 취소해버리거나 확인할 수 있는 일을 방지하기 위해서 예약번호 말고 다른 정보도 추가하고 요구하는 것을 넣으면 좋겠다고 생각했습니다. 또 매진 상태일때는 좌석정보창에서 좌석을 선택하는 것도 제한하면 좋겠다고 생각했습니다. 어차피 다 예약된 상태라서 선택해도 예약할 수는 없지만 선택하는 창에도 못 들어가게 제한하는 편이 깔끔할 것 같습니다.

|  |  |
| --- | --- |
| 5 | **과제 수행 완료 체크 리스트** |

과제 체크 리스트

1. 과제 제출 목록(보고서, 프로젝트 압축파일)은 모두 제출하였는가? 예

2. 제출한 결과물은 실행가능한가? 예

3. 보고서에는 개발 내용을 충분히 설명하고 있고, 기능별 실행결과와 그에 대한 설명이 기술되어 있는가?예

4. 모든 실행결과물에 학번 이름이 출력되어 있는가? 예

5. [1번 관련] 영화정보 파일을 읽어 동적할당을 통해 영화관 관련 정보를 생성하였는가? 예

6. [2번 관련] A. 메뉴는 종료를 선택할 때까지 반복해서 동작되는가? 예

B. [영화예매관련] i. 영화 리스트를 잘 보여주는가? 예

ii. 남은좌석/전체 좌석수 및 매진 상태를 잘 처리하였는가? 예

iii. 좌석 예약 현황을 잘 보여주고 있는가? 예

iv. 이미 예약된 자리는 예약하지 못하도록 처리하였는가? 예

v. 예약번호는 중복되지 않게 잘 결정하였는가? 예

C. [예매취소관련] i. 예약번호가 일치할 경우, 예매 취소를 수행하고, 좌석은 다시 예매 가능한 상태 로 만들었는가? 예

ii. 예약번호가 일치하지 않을 경우에 대한 처리는 하였는가? 예

D. [예매확인 관련]

i. 예약번호가 일치할 경우, 영화이름과 좌석정보 및 위치를 출력하였는가? 예

ii. 예약번호가 일치하지 않을 경우에 대한 처리는 하였는가? 예

E. [종료 관련] 메모리 누수 체크는 하였는가? 예