**전체화면 및 우측 하단 인쇄 모양 보기로 봐주세요!**



객체지향프로그래밍

Report #7

|  |  |
| --- | --- |
| 학번 | 20174627 |
| 이름 | 김혜진 |

문제 명세

**● 주제 선택**

* Library System
  + 도서 관리(조회, 삽입, 삭제 기능 등)
  + 도서 대여, 회원 등록(회원 구분: 학생, 대학원생, 교수 등
* Movie World
  + 영화 DVD 관리(조회, 삽입, 삭제 기능)
  + 대여 기능, 회원 등록(회원 구분: 일반, 골드 회원 등)
* Product Inventory System (여러분이 선호하는 품 목 선택)
  + 물품 관리 (조회, 삽입, 삭제 기능 등)
  + 구매 기능, 회원 등록(회원 구분)

**● C++ 요구 기능**

* 상속
* 연산자 중복
* 동적 바인딩(가상 함수)
* 예외 처리
* 포괄적 함수
* 그 외
* **지침 사항**
* 보고서는 문제 명세, 분석, 설계, 구현, 테스트/검증, 의견 순으로 작성
* 분석에는 클래스 계층구조객체 관계 다이어그램 포함
* 물품(도서, 영화, 품목)과 회원의 클래스 설계시에 반드시 클래스 계층 구조를 고려해야 한다.
* 설계에는 클래스 설계, 함수 알고리즘 기술 포함

물폼 목록은 파일상에 관리 (파일 읽기/쓰기 포함) (여기까지15점)

* 부가 기능은 사용자 편의성 제공을 위한 제반 기능을 포함한다(GUI 등 프로그램 기능 이용 편의성 제공, 스마트폰 어플 등을포함, GUI 이외 코드는 반드시 C++로 작성해야 함) (5점)

C++ 기능표

|  |  |  |
| --- | --- | --- |
| **기능** | **내용** | **사용 분야** |
| **C++ 기본** | 포인터 | 동적 할당 시 구조체포인터 및 클래스포인터 사용 |
| 배열 및 구조체 | 영화 정보를 입력하는 Movie 구조체 사용 |
| 레퍼런스 | MENU 함수에서 좌표를 지정하는 x,y에 사용 |
| **동적메모리 할당** | New, delete | Movie , Reservation , VIP\_Reservation를  동적 할당, 해제할 때 사용 |
| **예외처리** | Try, catch문 | 예외처리가 필요한 구간에 Try, catch문 사용으로 예외처리 |
| **포괄적 함수** | template | 호출 시 매개변수의 타입별로 실제 함수를 작성 |
| **개선된 함수** | 디폴트인자 | check 함수에서 사용 |
| 인라인 함수 및  매크로 함수 | PrintMovie에서 사용 |
| 함수오버로딩 | check 함수에서 사용 |
| **문자열** | C 및 C++  문자열의 사용 | Movie의 변수에 사용 |
| 문자/문자열 입출력 | cout , cin 빈번하게 사용 |
| **클래스 기초** | 클래스 설계 | Reservation, VIP\_Reservation에 클래스 사용 |
| 분할 컴파일 | 각 클래스와는 헤더와 cpp를 따로 구성 |
| **생성자 및 소멸자** | 디폴트 생성자 | 클래스들이 멤버변수를 초기화 해주는 용도로 사용 |
| 인자 있는 생성자 | 클래스들은 복수의 인자있는 생성자를 가지고 있음 |
| 소멸자 | 클래스들이 소멸자가 호출되면 화면을 지우고 안내메세지를 출력해주는 용도로 사용 |
| const 객체와  const 멤버함수 | 전달 인자에 const 객제 사용, 객제의 보호가 보장 |
| **여러 가지 객체의**  **생성 방법** | 동적객체, 객체배열,  객체에 대한  포인터 배열 등 | Movie \*M = new Movie[max]  Reservation \*R = new Reservation[max]; 등 동적 객체 사용 |
| **상속과 다형성** | 상속 | Reservation 클래스를 상속받은 VIP\_Reservation 클래스에 사용 |
| 동적바인딩 | Reservation 클래스와 상속받은 VIP\_Reservation 클래스의  void place\_situation(int x, int y)를 동적바인딩(가상함수) |
| **그 외** | gotoxy(int x, int y) | 좌표를 옮겨주는 함수 |
| system("cls") | 전체화면을 초기화 해준다 |

분석

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | | **Reservation** | | **일반 예매를 하는 클래스** | | int remain(); // 남아있는 좌석의 개수를 리턴해주는 함수  void ticket(); // 선택한 좌석을 예매하는 함수  void place\_situation(); // 현재 좌석 상황을 표시하는 함수  int USE(); // 해당 좌석이 사용 중인지 확인하는 함수 |  |  | | --- | | **VIP\_ Reservation** | | **VIP 회원으로 영화를 예매하는 클래스 - Reservation 클래스 상속** | | void discount(); // 할인쿠폰을 적용하는 함수  void place\_situation(); // 좋은 자리와 현재 좌석의 상황을 표시하는 함수 | |
|  |

설계(설명은 주석)

|  |
| --- |
| **main.cpp** |
| #include "R.h"  typedef struct Movie{  string stdio; // 영화사 이름  string now\_showing; // 현재 상영중인 영화  string coming\_soon; // 개봉 예정중인 영화  }Movie;  int check(Reservation\* R, int m , int L, int C); // 좌석 범위 체크  int check(Reservation\* R, int m , char L, int C); // 좌석 범위 체크  void InputMovie(Movie \*M, int n);  void inline PrintMovie(Movie \*M, int n , Reservation\* R);  template <class T>  int sumMoney(T sum, T cost);  void MENU(int &x, int &y);  int login();  void MovieR(Movie\* M);  int state;  int num ;  string name;  int main()  {  int max = MAX; // 이미 할당된 배열의 최대 크기를 세는 변수  int N;  int sum=0; //영화 결제금액  num = 0; //등록 된 영화 수 시작은 0  Movie\* M = new Movie[max]; //영화의 정보를 담는 구조체 포인터  login(); // 로그인기능 관리자, 일반 고객, VIP 3가지 경우  MovieR(M); // 영화를 등록하며, 관리자는 수동으로 나머지는 자동으로 기본값이 등록  int X=15,Y=4+num ,x,y; //프롬프트 출력 좌표 설정  int menu; //메뉴 선택에 대한 번호 저장  int m; //선택한 영화 상영관 번호 정보  Reservation\* R = new Reservation[max]; // 영화 갯수만큼 R 동적할당  VIP\_Reservation \*VIP\_R = new VIP\_Reservation[max]; // 영화 갯수마다 VIP\_R 동적할당  while(1) // 무한루프  {  PrintMovie(M,num,R); // 영화 정보 출력  x=X,y=Y; // 좌표값 변수  MENU(x,y); // 메뉴 출력  cin >> menu;  if(menu == 1 && state==2) // 일반 영화 예매  {  char L; // 행  int C; // 열  y+=2;  gotoxy(x,y);  cout << "영화관의 번호를 입력해주세요 : ";  cin >> m;  m--;  y+=2; x+=6;  R[m].place\_situation(x,y); // m관의 영화 자리 표시  cout << "좌석을 입력해주세요(ex> A 1) : ";  cin >> L >> C;  if(check(R,m,L,C) == 99) // 입력된 좌석이 범위에 맞지않으면  {  while(check(R,m,L,C) == 99) // 맞지 않으면 계속 반복  {  gotoxy(x,y+7);  cout << " "; // 표시된 좌석을 지워주는 역활  gotoxy(x,y+7);  cout << "좌석을 다시 입력해주세요 : "; // 좌석 다시 입력받기  cin >> L >> C;  check(R,m,L,C); // 범위에 맞는지 계속 체크  }  }  // 범위에 맞으면  R[m].ticket(L,C); // 해당 좌석 티켓팅  R[m].place\_situation(x,y); // 바뀐 좌석 출력  gotoxy(x,y+9);  cout << "예매가 완료되었습니다." ;  gotoxy(x-1,y+10);    if(M[m].coming\_soon != "X") // 개봉영화가 있다면  { // 해당 stdio 의 개봉영화 홍보  cout << M[m].stdio <<"의 '"<<M[m].coming\_soon <<"'이 개봉예정입니다.";  gotoxy(x,y+11);  cout <<"많은 관심 부탁드립니다.";}  VIP\_R[m].ticket(L,C); // VIP 클래스 같은자리에 티켓팅 (두개의 클래스의 좌석이 티켓팅)  sum = sumMoney(sum, (int)10000);  }  else if(menu == 2 && state == 3) // VIP 영화 예매  {  char L;  int C;  y+=2;  gotoxy(x,y);  cout << "영화관의 번호를 입력해주세요 : ";  cin >> m;  m--;  y+=2; x+=6;  VIP\_R[m].place\_situation(x,y); // VIP 좌석 표시 (+기능) 좋은자리 표시  cout << "좌석을 입력해주세요(ex> A 1) : ";  cin >> L >> C;    if(check(R,m,L,C) == 99) // 입력된 좌석이 범위에 맞지않으면  {  do  {  gotoxy(x,y+7);  cout << " ";  gotoxy(x,y+7);  cout << "좌석을 다시 입력해주세요 : ";  cin >> L >> C;  check(R,m,L,C);  }while(check(R,m,L,C) == 99);  }  // 좌석이 범위에 맞으면  VIP\_R[m].ticket(L,C); // 해당 자리 티켓팅  VIP\_R[m].place\_situation(x,y); // 바뀐 좌석 출력  gotoxy(x-4,y+9);  VIP\_R[m].discount(); // 할인쿠폰 적용 (VIP클래스 전용)  y++;  gotoxy(x,y+10);  cout << "예매가 완료되었습니다." ;  gotoxy(x-1,y+11);  if(M[m].coming\_soon != "X") // 개봉영화가 있다면  { //해당 studio의 개봉영화 홍보  cout << M[m].stdio <<"의 '"<<M[m].coming\_soon <<"'이 개봉예정입니다.";  gotoxy(x,y+12);  cout <<"많은 관심 부탁드립니다.";}  R[m].ticket(L,C); // 일반자리도 티켓표시  sum = sumMoney(sum, (int)7000);    }  else if (menu == 3 ) // 종료  {  gotoxy(X+4,Y+4);  cout << "프로그램이 종료됩니다.";  break;  }  Sleep(3000); // 3초 정지 후  system("cls"); // 화면 초기화  gotoxy(15, 20);  cout << "총" << sum << "원 결제하셨습니다";  }  Sleep(1000); // 무한루프를 벗어나면 1초 멈춤  delete[] R; // 메모리 해제  delete[] VIP\_R; // 메모리 해제  //소멸자 호출  }  void MENU(int &x, int &y) // 좌표가 다같이 바뀌어야하므로 레퍼런스로 매개변수  {  gotoxy(x,y);  cout << "1. 영화 예매하기" ;  y++;  gotoxy(x,y);  cout << "2. VIP회원으로 영화 예매하기" ;  y++;  gotoxy(x,y);  cout << "3. 종료"<< endl;  gotoxy(x+30,y);  cout << "\n" << "\n" << " " << "☞ ";  }  void gotoxy(int x, int y) // 커서의 위치를 이동시키는 함수  {  COORD Pos = {x-1, y-1};  SetConsoleCursorPosition(GetStdHandle(STD\_OUTPUT\_HANDLE),Pos);  }  void InputMovie(Movie \*M, int n)  {  char str[100];  cin.clear(); // 버퍼 비워주기  fflush(stdin); // 버퍼 비워주기  if (n >= 0) {  fgets(str, 2, stdin);  }  cout << "▶ stdio : ";  getline(cin,M[n].stdio);  cout << "▶ now\_showing : ";  getline(cin,M[n].now\_showing);  cout << "▶ coming\_soon : ";  getline(cin,M[n].coming\_soon);  cout <<"\n";  }  void inline PrintMovie(Movie \*M, int n, Reservation\* R)  {  for(int i=0;i<n;i++) // 첫번째 고객부터 입력한 고객번호까지 출력  {  gotoxy(15,3+i);  cout << (i+1) <<"관 " ;  cout.setf(ios\_base::left); // 왼쪽정렬  cout.width(10); // 너비 10  cout << M[i].stdio ;  cout.width(12);  cout << M[i].now\_showing ;  gotoxy(47,3+i);  cout << R[i].remain();  }  }  int check(Reservation\* R, int m , int L, int C=0)  {  return 99;  }  int check(Reservation\* R, int m , char L, int C) // 오류 (문자만 입력한 경우)  {  if(L < 65 || L >68) // A~D 범위에 벗어난경우  return 99;  if(C<1 || C>9) // 1~9 범위에 벗어난경우  return 99;  int use = R[m].USE(L, C); // 입력된 자리 사용여부 확인  if(use ==1) // 입력된 자리가 사용한 자리라면  return 99;  } // 99는 오류로 main에서 판정  int login() {  gotoxy(10, 2);  cout << "회원번호를 입력해주세요:";  string id;  cin >> id;  if (id == "admin") { //관리자 번호로 로그인  state = 1; //admin 번호  return 0;  }  else {  fstream r, vr;  string fname;  r.open("r.txt", ios::in); //일반 유저의 회원정보가 담긴 txt파일  vr.open("vr.txt", ios::in); //VIP 유저의 회원정보가 담긴 txt파일  while (r >> fname) { //파일에서 값 읽기  if (id == fname) { //파일내부에 입력한 정보가 있는지 확인  state = 2; //일반 유저번호  name = fname;  system("cls");;  return 0;  }  }  while (vr >> fname) { //파일에서 값 읽기  if (id == fname) { //파일내부에 입력한 정보가 있는지 확인  state = 3; //VIP 유저번호  name = fname;  system("cls");  return 0;  }  }  }  return 0;  }  void MovieR(Movie\* M) { //영화를 등록하는 함수  if (state == 1) { //관리자가 수동등록  int N;  int max = MAX;  cout << " 영화정보를 입력하세요";  cout << endl;  while (1)  {  if (num == max)  {  try {  max += 1;  Movie\* temp = new Movie[max]; // 할당된 크기만큼 새로운 구조체포인터 temp 동적할당  memcpy(temp, M, (num) \* sizeof(Movie)); // 문자열 복사기능  //delete M; // 동적으로 할당한 M 할당 해제  M = temp;  }  catch (exception e) { continue; }  }  for (int i = num; i < max; i++)  {  InputMovie(M, i);  num++;  }  cout << "영화정보를 더 입력하시겠습니까? (1.Yes, 2.No)";  cin >> N;  cout << "\n";  if (N == 2) // NO 입력시 무한루프 종료  break;  } // while 종료  cout << "영화정보를 모두 입력하셨습니다. " << endl;  cout << "영화 예매 프로그램을 시작하겠습니다." << endl;  Sleep(700);  }  else if(num==0){ //관리자가 아닐경우 디폴드 영화 등록  Movie\* temp = new Movie[2];  memcpy(temp, M, (1) \* sizeof(Movie));  M[0].stdio = "fox";  M[0].now\_showing = "제인도";  M[0].coming\_soon = "해바라기";  M[1].stdio = "showbox";  M[1].now\_showing = "유전";  M[1].coming\_soon = "어바웃타임";  num += 2;  }  system("cls");  }  template <class T>  int sumMoney(T sum, T cost) { //포괄적함수  sum+=cost;  return (int)sum;  } |

|  |
| --- |
| **Reservation.cpp** |
| #include "R.h"  void Reservation::ticket(const char l,int r) // 해당 좌석을 예매하는 함수  {  r--; // 배열은 0부터 시작이므로 -1 해준다.  if(l == 'A') // A는 0부터 시작  r;  else if(l == 'B') // B줄은 10부터 시작  r+=10;  else if(l == 'C') // C줄은 20부터 시작  r+=20;  else if( l =='D') // D줄은 30부터 시작  r+=30;  else // A~D 범위가 아닐경우 오류 메세지  cout << "오류" ; // 출력  place[r] = 1; // 해당자리 사용(1)으로 표시  able--; // 현재 좌석 갯수 -1  }  int Reservation::remain() // 현재 남아있는 좌석 갯수를 리턴하는 함수  {  return able; // 남은 좌석갯수 리턴  }  void Reservation::place\_situation(int x, int y) // 현재 좌석 상황 표시하는 함수  {  char alpa = 65; // A~D 문자를 나타낼 변수  int j=0;  gotoxy(x+5,y); // 커서 이동  cout << "S C R E E N";  y++;  gotoxy(x,y);  for(int i=0;i<40;i++)  {  if(i%10 == 0)  {  if(j==0)  cout << " 1 2 3 4 5 6 7 8 9 10";  y++;  gotoxy(x,y);  cout << alpa;  alpa++;  j++;  }  if(place[i]==0) // 사용 안함(0) 자리는 □ 으로 표시  cout << "□";  else if (place[i] ==1) // 사용함(1) 자리는 ■ 으로 표시  cout << "■";  }  gotoxy(x,y+2);  }  int Reservation::USE(char l, int r) // 해당자리가 사용중인지 리턴해주는 함수  {  r--;  if(l == 'A')  r;  else if(l == 'B')  r+=10;  else if(l == 'C')  r+=20;  else if( l =='D')  r+=30;  return place[r]; // 해당 자리 상태 출력 (사용함 1, 사용안함 0)  } |

|  |
| --- |
| **VIP\_Reservation.cpp** |
| #include "R.h"  void VIP\_Reservation::discount() // 랜덤으로 쿠폰 할인 적용  {  cout <<"(VIP)"<< coupon <<"% 할인된 가격으로 적용됩니다.";  }  void VIP\_Reservation::place\_situation(int x, int y) // 동적바인딩  {  char alpa = 65;  int j=0;  gotoxy(x+5,y);  cout << "S C R E E N";  y++;  gotoxy(x,y);  for(int i=0;i<40;i++)  {  if(i%10 == 0)  {  if(j==0)  cout << " 1 2 3 4 5 6 7 8 9 10";  y++;  gotoxy(x,y);  cout << alpa;  alpa++;  j++;  }  if(place[i]==0) // 사용 안함(0)은 □ 으로 표시  cout << "□";  else if (place[i] ==1) // 사용중(1) 은 ■ 으로 표시  cout << "■";  else if (place[i] == 9) // 좋은자리는 ☆ 으로 표시  cout << "☆";  }  gotoxy(x,y+2);  } |

|  |  |  |
| --- | --- | --- |
| **R.h** | **Reservation 클래스** | **VIP\_Reservation 클래스** |
| #include <iostream>  #include <string>  #include <Windows.h>  #include <fstream>  #include <list>  using namespace std;  #define MAX 2  ////////////////////////////////////////////////////  void gotoxy(int x, int y);  //////////////////////////////////////////////////  class Reservation  {  protected:  int place[40]; // 좌석 표시  int able ; // 남은 좌석의 갯수  public:  Reservation() // 디폴트 생성자  {  able=40; // 남은 좌석을 0 으로 초기화  for(int i=0 ; i<40 ; i++)  place[i] =0; // 좌석을 사용안함(0)으로 초기화  }  Reservation(int able\_num)  {  able = able\_num ;  for(int i=0 ; i<able ; i++)  place[i] =0; // 좌석을 사용안함(0)으로 초기화  }  ~Reservation() // 소멸자  {  system("cls");  gotoxy(20,5);  cout << "이용해 주셔서 감사합니다.";  gotoxy(20,6);  }  int remain() ; // 남아있는 좌석 출력  void ticket(const char l ,int r); // 좌석을 예매하는 함수  virtual void place\_situation(int x, int y); // 가상함수  int USE(char l, int r); // 해당 좌석이 사용중인지 확인  };  class VIP\_Reservation : public Reservation  {  protected:  int coupon;  public:  VIP\_Reservation()  {  coupon = rand()%3+1;  coupon = coupon\*10; // 10 20 30% 랜덤으로 할인  for(int i=0 ; i<40 ; i++)  place[i] =0; // 좌석을 사용안함(0)으로 초기화  for(int j=0 ; j<4 ; j++)  {  place[23+j] = 9; // 좋은자리(9) 표시  place[33+j] = 9;  }  }  void discount();  void place\_situation(int x, int y); // (+기능)좋은자리 표시  }; | | |

알고리즘 설명 및 시스템 구현

|  |
| --- |
| 1. 실행하면 회원번호를 입력하라는 창이 뜬다. 2. 계정은 admin(관리자)와 일반유저 그리고 VIP로 나뉜다. 3. admin(관리자)은 영화를 추가할 수 있는 기능을 가지고 있다.      1. 일반유저와 VIP 계정에 해당하는 값이 들어있는 텍스트 파일이 있는데, 이 텍스트 파일과 값이 일치하면 로그인 할 수 있다. (r=일반, vr=VIP)      1. admin으로 로그인 하게 되면 영화 정보를 입력해달라는 문구와 동시에 영화 정보를 입력할 수 있는 텍스트가 뜨며, 타이틀에 해당하는 값을 입력해주면 된다.      1. 기본으로 2개의 영화 정보를 입력할 수 있으며, 그 이후부터는 1번 또는 2번 선택에 따라 나뉜다. (즉, 정보를 더 입력할 것인지 아닌지.)      1. 일반 유저로 로그인 했을 경우, 영화 목록이 출력 되면서 예매 화면이 뜬다. 2. 일반 유저이기 때문에 2번을 입력할 경우 조건문에 의해 잘못된 동작을 막는다.      1. 영화 예매의 경우, 예매할 영화관의 영화관 번호를 입력 받는다.      1. 번호를 입력 받으면 현재 그 영화의 좌석 상황을 출력한다.      1. 선택할 좌석을 입력 받는다. 2. 선택한 좌석이 문제가 없다면 해당 좌석을 예매한다. 3. 예매가 완료되면 좌석 색이 변하고, 예매 완료 메시지가 출력된다.      1. 예매가 끝나면 메인으로 자동 복귀되며, 예매된 영화 만큼의 숫자가 줄어들고, 결제한 영화의 값이 나온다.      1. 영화는 계속 구매할수록 시스템 하단에 값이 누적되어 출력 된다.      1. 만약, VIP 회원으로 예매하고 싶다면 vr 로 로그인 하면 된다. 2. VIP도 일반 회원의 예매와 마찬가지로 예매할 영화관 번호를 입력 받는다. 3. 단, 일반 예매와는 다르게 VIP 예매는 프리미엄 좌석을 보유하고 있기 때문에 이를 표시해준다. 4. 해당 좌석이 예매되면 예매된 자리의 상태가 바뀌고, VIP 회원의 할인 쿠폰이 적용된다.      1. VIP 또한, 예매한 영화의 좌석이 줄어듦을 확인할 수 있으며, 하단에 결제한 만큼의 값이 누적된다.        1. 프로그램을 종료하면 프로그램이 종료되면서 안내 메시지가 출력된다. |

GUI 아쉬운 점

GUI를 구현하려고 visual Studio를 사용해 windows form으로 GUI를 구현해보았습니다. 그런데 C++의 경우 C#에서 사용하던 windows form과는 다른 구조를 가지고 있어서 사용하기가 너무 어려웠습니다. 문자열 타입 변환 조차도 함수를 직접 만들어서 변환해주어야 할 만큼 번거로운 부분도 많았습니다. 하물며 제공되는 정보도 극소수이며, 제가 구현한 UI와 사람들이 구현한 UI의 구현 방법이 달랐기 때문에 간혹 레퍼런스를 참조하더라도 제 환경에 맞춰 변환해 주어야 했기 때문에 힘든 부분이 없지 않아 있었던 것 같습니다. 때문에 아쉽게도 완벽한 GUI를 구현하지 못했다는 부분에 아쉬운 마음이 컸습니다. 하지만 시작이 반이라고, 앞으로도 시간이날 때 마다 다루다 보면 언젠가 능숙하게 다룰 날이 찾아오지 않을까 싶습니다. 아래는 제가 구현한 GUI에 대한 구현 및 설명입니다.

GUI 설계

|  |
| --- |
| **MyForm.cpp (메인 소스 파일)** |
| #include "MyForm.h"  #include "mainForm.h"  #include <iostream>  using namespace System;  using namespace System::Windows::Forms;  [STAThreadAttribute]  void Main(array<String^>^ args) {  Application::EnableVisualStyles();  Application::SetCompatibleTextRenderingDefault(false);  Project1::MyForm form;  Application::Run(% form);  std::fstream os("isitlogin.txt");  if (os) {  std::string temp;  while (os >> temp) {  if (temp == "nope") {  break;  }  else {  Project1::mainForm form1;  Application::Run(% form1);  }  }  }  std::fstream isitlogin("isitlogin.txt", std::ios::out);  isitlogin << "nope";  isitlogin.close();  } |

|  |
| --- |
| **MainForm.h (메인 헤더 파일) – 구현한 부분의 코드만 가져왔습니다.** |
| #pragma once  #include <iostream>  #include <fstream>  #include <Windows.h>  #include "re.h"  #pragma endregion  private: System::Void mainForm\_Load(System::Object^ sender, System::EventArgs^ e) {  fstream idpw("isitlogin.txt");  string name{};  idpw >> name;  label1->Text = gcnew String(name.c\_str());  reservation re{};  dataGridView1->Rows->Clear();  dataGridView1->ColumnCount = 2;  dataGridView1->Columns[0]->Name = "제목";  dataGridView1->Columns[1]->Name = "배급사";    for(auto& temp : re.MR) {  dataGridView1->Rows->Add();  dataGridView1->Rows[0]->Cells[0]->Value = gcnew String(temp.m.name.c\_str());  dataGridView1->Rows[0]->Cells[1]->Value = gcnew String(temp.m.stdio.c\_str());  }  } |

|  |
| --- |
| **MyForm.h (로그인 헤더 파일) – 구현한 부분의 코드만 가져왔습니다.** |
| #pragma once  #include "mainForm.h"  #include <iostream>  #include <string>  #include <fstream>  #include <time.h>  #include <comdef.h>  #pragma endregion  private: System::Void label1\_Click(System::Object^ sender, System::EventArgs^ e) {  }  private: System::Void pictureBox2\_Click(System::Object^ sender, System::EventArgs^ e) {  }  private: System::Void MyForm\_Load(System::Object^ sender, System::EventArgs^ e) {  }  //// 1값을 받아 로그인한 유저의 로그인 일자, 시간, 계정을 log.txt 파일에 입력  private: System::Void button10\_Click(System::Object^ sender, System::EventArgs^ e) {  string a, b;  MarshalString(textBox1->Text, a);  MarshalString(textBox2->Text, b);  if (CorrectIdPw(a, b)) {  ofstream log;  log.open("log.txt",ios::out| ios::app);  tm t;  time\_t tt;  tt = time(NULL);  localtime\_s(&t, &tt);    //// 접속한 유저의 아이디를 isitlogin.txt 파일에 입력  log << "접속 시간:" << t.tm\_year + 1900 << ":" << t.tm\_mon + 1 << ":" << t.tm\_mday << ":" << t.tm\_hour << ":" << t.tm\_min << ":" << t.tm\_sec << " \t아이디:" << a << " \t패스워드: " << b << endl;  fstream isitlogin("isitlogin.txt",ios::out);  isitlogin << a;  isitlogin.close();  this->Close();  //mainForm^ maintest = gcnew mainForm;  //maintest->ShowDialog();    }  }  public: int CorrectIdPw(string id, string pw) { //// loing.txt 파일이랑 로그인창에 입력한 값이랑 일치하는지 비교,  fstream os("login.txt"); //// 일치하면 1 값을 가지고 리턴, 다음으로 출력 파일 닫음  if (!os) {  this->Close();  }  string tid, tpw,m;  while (os >> tid >> tpw>>m) {  if (tid==id && tpw==pw) {  os.close();  return 1;  }  }  os.close();  return 0;  }  //// 아래는 타입 변환을 위해 생성해준 함수  void MarshalString(String^ s, string& os) {  using namespace Runtime::InteropServices;  const char\* chars =  (const char\*)(Marshal::StringToHGlobalAnsi(s)).ToPointer();  os = chars;  Marshal::FreeHGlobal(IntPtr((void\*)chars));  }  void MarshalString(String^ s, wstring& os) {  using namespace Runtime::InteropServices;  const wchar\_t\* chars =  (const wchar\_t\*)(Marshal::StringToHGlobalUni(s)).ToPointer();  os = chars;  Marshal::FreeHGlobal(IntPtr((void\*)chars));  }  private: System::Void textBox1\_TextChanged(System::Object^ sender, System::EventArgs^ e) {  } |

|  |
| --- |
| **re.h (구현 못 함)** |
| #pragma once  #include <iostream>  #include <string>  #include <fstream>  #include <vector>  typedef struct Movie {  std::string stdio; // 영화사 이름  std::string name; // 현재 상영중인 영화  }Movie;  typedef struct MovieReservation {  Movie m;  int seatState[20][20]{};  }MovieReservation;  class reservation {  private:    public: // movie.txt에 있는 정보를 가져와 객체화 시켜줌  std::vector<MovieReservation> MR;  reservation() {  std::fstream ml("movie.txt");  std::string temp;  while (ml >> temp) {  int i =0;  MovieReservation movie;  movie.m.stdio = temp;  ml >> temp;  movie.m.name=temp;  for (int i = 0; i < 20; i++) {  for (int j = 0; j < 20; j++) {  ml >> temp;  movie.seatState[i][j] = atoi(temp.c\_str());  }  }  MR.push\_back(movie);  }    }  //void showseat(int n);  //void Update(fstream ofs);  }; |

GUI 알고리즘 설명 및 시스템 구현

|  |
| --- |
| 1. 프로그램 실행 시 제일 먼저 뜨는 로그인 창입니다.      1. 로그인은 텍스트 파일(login.txt)에 저장되어 있는 계정을 통해 절차가 이루어 집니다      1. 로그인에 성공하면 log.txt 파일에 접속 일자와 시간 그리고 계정이 자동으로 입력되어 집니다.      1. 또한, isitlogin.txt파일 에는 현재 로그인 되어 있는 유저의 아이디가 입력되며, 로그아웃 시 유저 아이디는nope 값으로 변경됩니다.      1. 로그인이 끝나면 메인 화면으로 들어오게 됩니다. 우측 상단에는 로그인한 유저의 ID가 보여 지며, Movie List와 Seats의 경우 그리드를 사용하여 배열로 영화 제목 및 배급사 값을 넣어주고, 배열로 좌석을 만들어 잔여석을 보여주려고 하였으나, 해당 부분 까지는 구현하지 못하였 습니다. |

의견

기말고사 이후에도 과제가 있어 씁쓸한 마음이 있었으나, 기한은 넉넉하게 주셨기 때문에 한 편으로는 씁쓸한 마음을 덜 수 있었습니다. 이 프로그램을 만들면서 아직 배울 것이 많다는 걸 수도 없이 느꼈으나, 반대로 이만큼 해낸 자신을 돌아보니 대견하다는 생각도 들었습니다. 배울 것이 많은 만큼 미숙한 점도 많겠지만, 이를 계기로 포기하지 않고 꾸준히 노력하다 보면 머지않아 정상을 찍을 수 있지 않을까 라는 기대도 해봅니다. 한 학기 동안 고생 많으셨습니다. 코로나 조심하시고 날이 부쩍 추워졌으니 건강 잘 챙기시길 바랍니다. 감사합니다.