게임 프로그래밍

HW1 보고서

박종승 교수님

컴퓨터 공학부

201401414 곽용준

∙ 키 조작

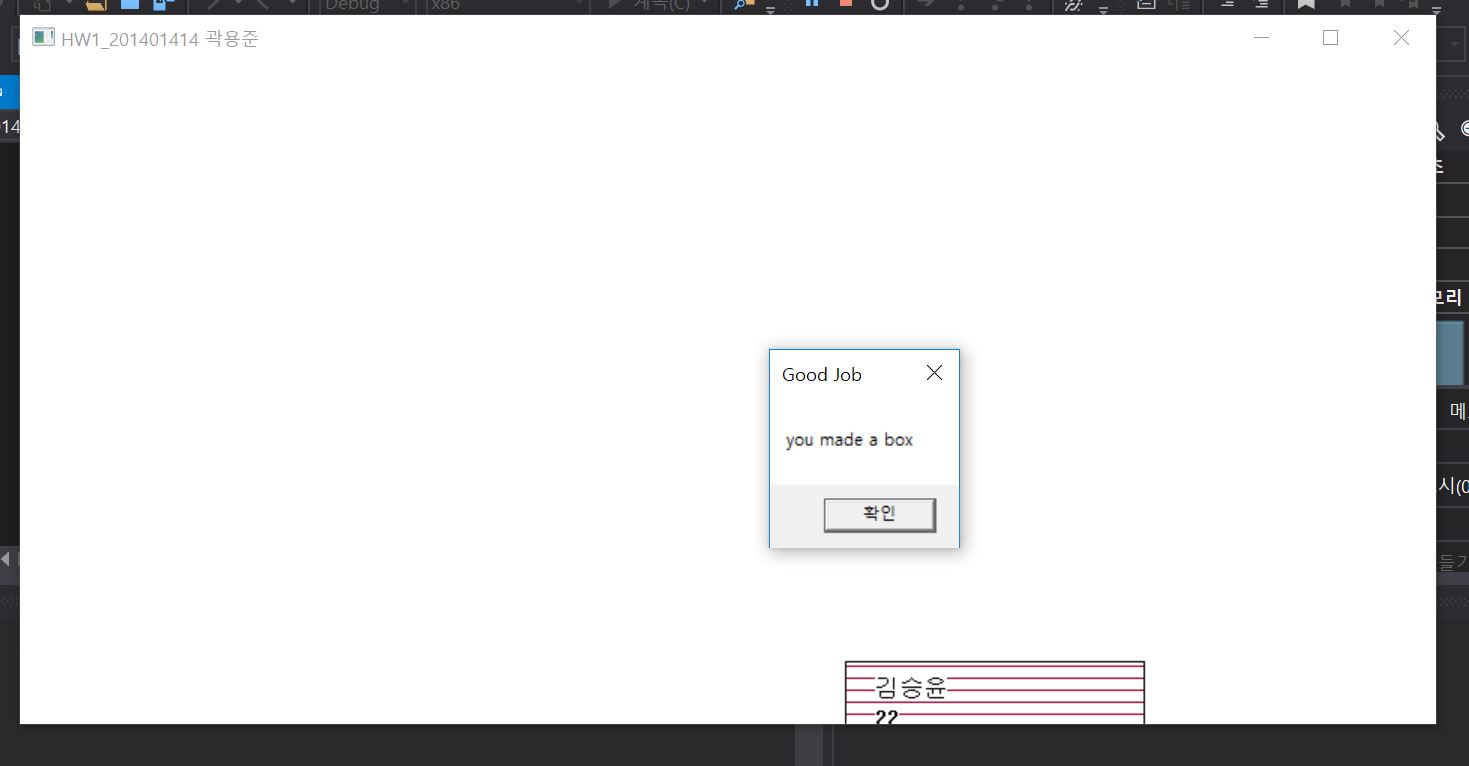
처음 화면을 클릭시에는 임의의 한글 이름과 20~50사이의 성적으로 구성된 박스가 생성된다. 마지막 생성된 박스를 제외한 이외의 곳을 클릭하여 박스를 추가할 수 있으며 이는 최대 6개까지 가능하다. 가장 마지막에 생성된 박스를 클릭시에는 해당 박스가 제거된다.

ESC를 누를 시에는 화면이 종료된다.

∙ 실행 결과



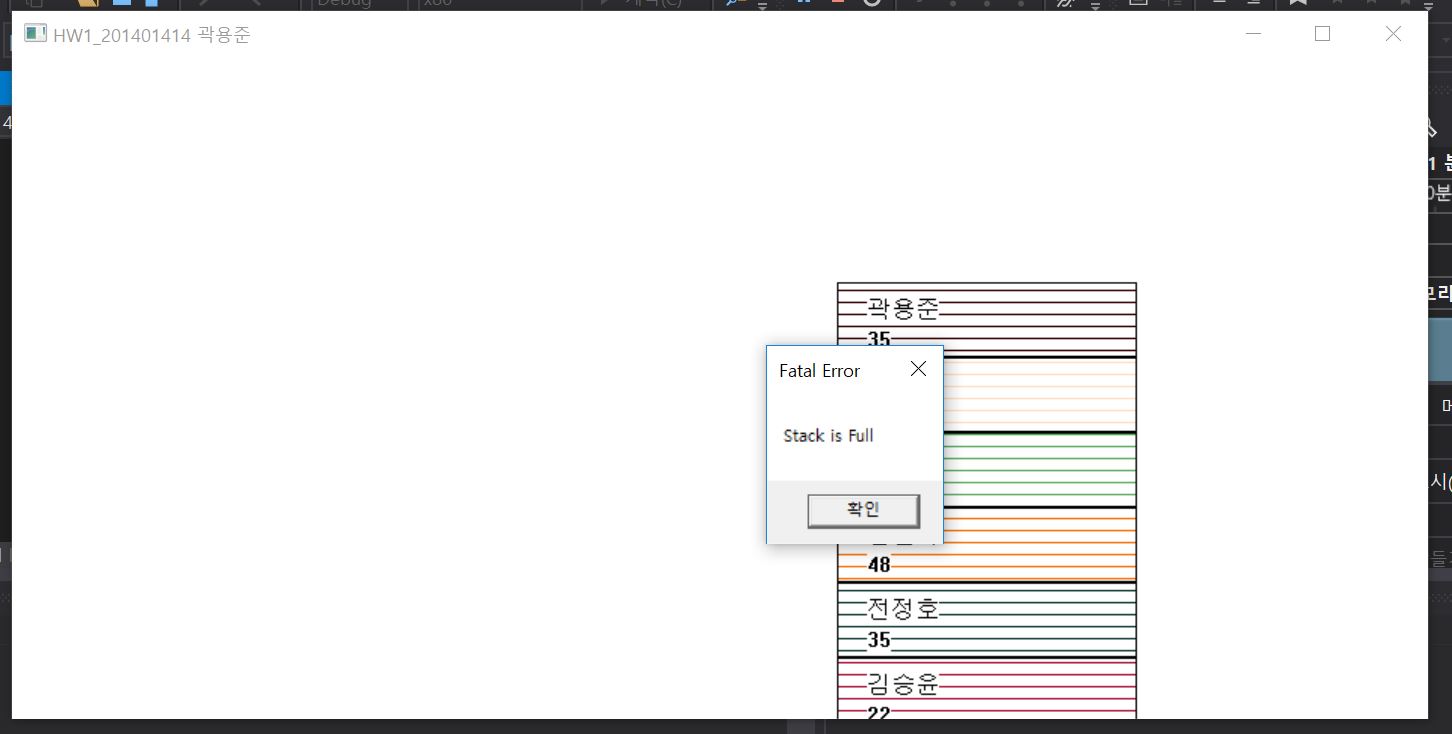
최초 실행시켰을 시의 화면이다 윈도우 창의 제목은 현재 폴더명으로 설정하였으며 흰색바탕에 아무것도 표시되고 있지 않은 모습이다.



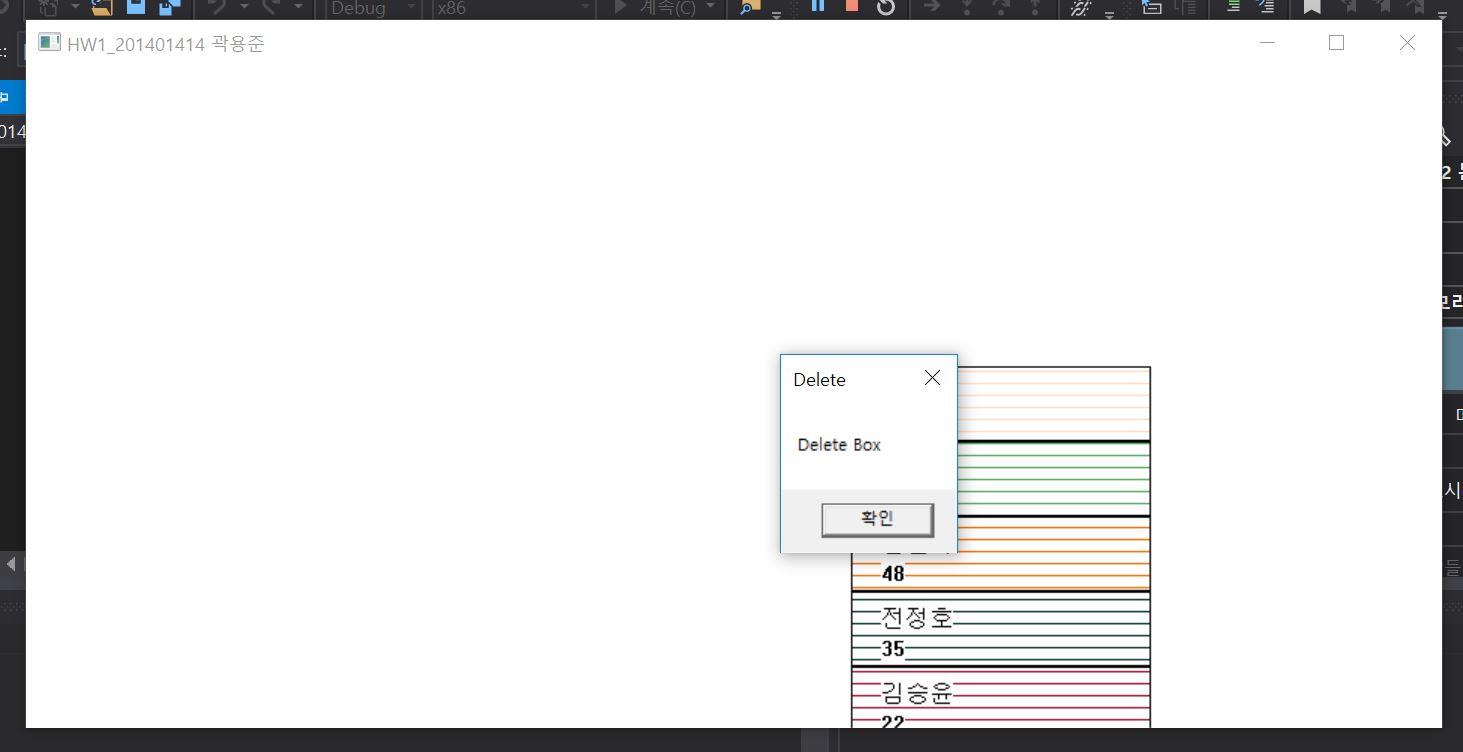
최초로 클릭 했을 시의 모습이다. 하단에 일정 크기의 박스가 생성되며 “김승윤 22”라는 임의의 값도 출력되는 모습이다. 박스를 생성시에는 다음과 같은 메시지박스도 출력되도록 하였다.



박스는 다음과 같이 최대 6개까지 생성 가능하다. 각기 다른 색상의 박스가 생성되며 이는 임의의 색상이 아닌 각 층에 맞는 색상으로 설정하였다. 즉 박스를 삭제하고 다시 생성해도 해당 층은 색상이 바뀌지 않는다.



박스를 6개까지 만들었는데 또 생성하려고 했을 시에 나오는 에러 메시지이다.



맨 위 박스를 클릭 시에는 해당 박스가 삭제되고 다음과 같은 메시지박스가 출력된다.

∙ 소스 코드

#include <windows.h> // 이 헤더파일은 Win32 API 관련 구조와 함수 선언들을 포함함.

#include <windowsx.h>

#include <string>

#include <time.h>

#include <vector>

#include <iterator>

#include <random>

using namespace std;

HWND hWndMain = 0; // 메인 윈도우 핸들.

bool initWin(HINSTANCE instanceHandle, int show);

int runWin();

LRESULT CALLBACK WndProc(HWND hWnd, UINT msg, WPARAM wParam, LPARAM lParam);

struct student {

string name;

int score;

};

//학생 구조체

string name[10] = { "곽용준", "우세훈", "김광표", "이영진" , "전정호", "김승윤", "한정훈", "김규형","안준혁" , "김민석" };

vector<student> stack;

int sp = -1;

vector<student>::iterator it;

//스택과 스택포인터

bool isFull() {

return sp == 5;

}

bool isEmpty() {

return sp == -1;

}

//스택함수

int recx = 550;

int recy = 400; //그려질 박스의 좌표 변수

int xpos = 0;

int ypos = 0; //마우스 좌표값 변수

int makeDelete = 0; //그리기 및 삭제 모드 변수

// 응용 프로그램 진입 함수.

int WINAPI WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance, PSTR pCmdLine, int nCmdShow)

{

if (!initWin(hInstance, nCmdShow)) {

MessageBox(0, "initWin - Failed", "Error", MB\_OK);

return FALSE;

}

return runWin();

}

// 초기화가 성공적으로 수행되면 true를 되돌림.

bool initWin(HINSTANCE hInstance, int nCmdShow)

{

// WNDCLASS 구조체에 값을 채워서 생성하고자 할 윈도우의 특성을 지정함.

WNDCLASS wc;

wc.style = CS\_HREDRAW | CS\_VREDRAW;

wc.lpfnWndProc = (WNDPROC)WndProc;

wc.cbClsExtra = 0;

wc.cbWndExtra = 0;

wc.hInstance = hInstance;

wc.hIcon = LoadIcon(hInstance, (LPCTSTR)IDI\_APPLICATION);

wc.hCursor = LoadCursor(NULL, IDC\_ARROW);

wc.hbrBackground = (HBRUSH)(COLOR\_WINDOW + 1);

wc.lpszMenuName = 0;

wc.lpszClassName = "Hello";

// 윈도우 클래스 description을 등록함.

if (!RegisterClass(&wc))

{

MessageBox(0, "RegisterClass - Failed", 0, 0);

return false;

}

// 등록된 윈도우 클래스 description에 기반하여 윈도우를 생성함.

// 생성된 윈도우의 HWND를 되돌림.

hWndMain = CreateWindow(

"Hello", "HW1\_201401414 곽용준", WS\_OVERLAPPEDWINDOW,

CW\_USEDEFAULT, CW\_USEDEFAULT, CW\_USEDEFAULT, CW\_USEDEFAULT,

NULL, NULL, hInstance, NULL);

if (!hWndMain) {

MessageBox(0, "CreateWindow - Failed", 0, 0);

return false;

}

// 생성된 윈도우를 show하고 update함.

ShowWindow(hWndMain, nCmdShow);

UpdateWindow(hWndMain);

return true;

}

// 메시지 루프를 포함하는 함수.

int runWin()

{

MSG msg;

ZeroMemory(&msg, sizeof(MSG));

// 윈도우가 생성된 후 메시지 루프에 진입하고 WM\_QUIT 메시지가 올 때까지 메시지 루프를 반복함.

// GetMessage 함수는 WM\_QUIT 메시지를 받을 때만 0을 되돌림. 이때 루프를 벗어나도록 함.

while (GetMessage(&msg, NULL, 0, 0) != 0) {

TranslateMessage(&msg); //윈도우가 생성한 가상키 코드를 문자메시지로 해석함

DispatchMessage(&msg); //메시지를 window procedure로 보냄

}

return (int)msg.wParam;

}

// WIN32 함수를 사용한 2차원 그리기의 예.

LRESULT OnPaint(HWND hWnd)

{

PAINTSTRUCT ps;

HDC hDC = BeginPaint(hWnd, &ps);

RECT rect;

GetClientRect(hWnd, &rect);

FillRect(hDC, &rect, ::GetSysColorBrush(COLOR\_WINDOW));

EndPaint(hWnd, &ps);

return 0;

}

LRESULT drawRectangle(HWND hwnd, HDC hdc)

{

char charName[10];

char charScore[10];

HBRUSH MyBrush, OldBrush;

COLORREF color;

it = stack.begin(); //iterator 초기화

int count = 0;

if (makeDelete == 1) { //박스 입력모드일때만 박스를 그리기 시작함

for (int i = 0; i < sp + 1; i++) {

string name = it->name;

int score = it->score;

string scoreString = to\_string((long long int)score); // int -> string 변환

sprintf\_s(charName, "%s", name.c\_str()); //string -> char 변환 name

sprintf\_s(charScore, "%s", scoreString.c\_str()); //string -> char 변환 score

///////////////////////////////////////////////////////////

switch (count) {

case 0:

color = RGB(165, 0, 38);

break;

case 1:

color = RGB(0, 48, 39);

break;

case 2:

color = RGB(244, 109, 0);

break;

case 3:

color = RGB(90, 174, 97);

break;

case 4:

color = RGB(254, 224, 200);

break;

case 5:

color = RGB(50, 0, 0);

break;

}

//count값을 이용하여 각 박스의 색깔 지정

MyBrush = CreateHatchBrush(HS\_HORIZONTAL, color);

OldBrush = (HBRUSH)SelectObject(hdc, MyBrush);

Rectangle(hdc, recx, recy, recx + 200, recy + 50);

TextOut(hdc, recx + 20, recy + 10, charName, strlen(charName));

TextOut(hdc, recx + 20, recy + 30, charScore, strlen(charScore));

recy -= 50;

//다음의 박스는 y좌표 갱신하여 그림

it++;

count++;

//iterator, count값 증가하여 다음 박스 그리기로 넘어감

SelectObject(hdc, OldBrush);

}

}

return 0;

}

// 윈도우가 받는 이벤트들을 처리하는 윈도우 프로시져.

LRESULT CALLBACK WndProc(HWND hWnd, UINT uMsg, WPARAM wParam, LPARAM lParam)

{

PAINTSTRUCT ps;

HDC hdc;

switch (uMsg)

{

case WM\_LBUTTONDOWN:

// 왼쪽 마우스 버튼이 눌려지면 좌표값 저장

xpos = GET\_X\_LPARAM(lParam);

ypos = GET\_Y\_LPARAM(lParam);

if (isEmpty()) {

makeDelete = 1; //생성모드

student tempObj;

srand((unsigned)time(NULL));

tempObj.name = name[rand() % 9];

tempObj.score = rand() % 30 + 20;

recy = 400;

stack.push\_back(tempObj);

sp++; // 데이터 생성해서 백터에 삽입

InvalidateRect(hWnd, NULL, FALSE);

::MessageBox(0, "you made a box", "Good Job", MB\_OK); //그리기 성공 시 메시지 박스 출력

}

else {

if (((recx <= xpos) && (xpos <= recx + 200) && (recy+50 <= ypos) && (ypos <= recy+100)) && (recy <= 600)) { //박스 삭제 조건

makeDelete = 0;

RECT temp; //지울 좌표를 임시로 넣어둔 합수의 매개변수

temp.left = recx;

temp.top = recy +50;

temp.right = recx + 200;

temp.bottom = recy + 100;

recy += 50;

//y좌표 갱신

stack.pop\_back();

sp--;

//박스 지우기 전 스택에서 pop

InvalidateRect(hWnd, &temp, TRUE);

::MessageBox(0, "Delete Box", "Delete", MB\_OK); //박스 제거시 메시지 박스 출력

}

else { //박스 추가

if (isFull()) { //스택검사 실시

::MessageBox(0, "Stack is Full", "Fatal Error", MB\_OK);

::MessageBeep(MB\_OK); //꽉 찼을 경우 비프음과 함께 메시지 박스 출력

return 0;

}

else {

makeDelete = 1; //생성모드

student tempObj;

srand((unsigned)time(NULL)); // 데이터 생성해서 백터에 삽입

tempObj.name = name[rand() % 9];

tempObj.score = rand() % 30 + 20;

stack.push\_back(tempObj);

sp++;

recy = 400;

//y좌표 초기화하여 처음부터 그릴 수 있도록 설정

InvalidateRect(hWnd, NULL, TRUE);

::MessageBox(0, "you made a box", "Good Job", MB\_OK);

}

}

}

return 0;

case WM\_KEYDOWN:

// ESC 키가 눌려지면, 윈도우를 destroy함.

if (wParam == VK\_ESCAPE)

DestroyWindow(hWndMain);

return 0;

case WM\_DESTROY:

// destroy 메시지를 받으면 quit 메시지를 보내서 메시지 루프를 벗어나도록 함.

PostQuitMessage(0);

return 0;

case WM\_PAINT:

hdc = BeginPaint(hWnd, &ps);

drawRectangle(hWnd, hdc);

EndPaint(hWnd, &ps);

break;

default:

// 그 외의 메시지들은 디폴트 윈도우 프로시져로 forward함.

return DefWindowProc(hWnd, uMsg, wParam, lParam);

}

return 0;

}