**비주얼프로그래밍**

**요소설계 프로젝트**

|  |  |
| --- | --- |
| 프로젝트 명 | *같은 그림 찾기* |
| 팀 명 | *111213* |
| 문서 제목 | 결과보고서 |

|  |  |
| --- | --- |
| **Version** | 1.5 |
| **Date** | 2014-NOV-26 |

|  |  |
| --- | --- |
| **팀원** | 이 동욱(조장) |
| 박 지현 |
| 박 혜란 |
|  |
| **지도교수** | 김 상철 교수님 |

|  |
| --- |
| **CONFIDENTIALITY/SECURITY WARNING**  이 문서에 포함되어 있는 정보는 국민대학교 전자정보통신대학 컴퓨터공학부 및 컴퓨터공학부 개설 교과목 비주얼프로그램밍 수강 학생 중 프로젝트 “같은 그림 찾기”를 수행하는 팀 “111213”의 팀원들의 자산입니다. 국민대학교 컴퓨터공학부 및 팀 “111213”의 팀원들의 서면 허락없이 사용되거나, 재가공 될 수 없습니다. |

**문서 정보 / 수정 내역**

|  |  |
| --- | --- |
| **Filename** | 중간보고서-같은그림찾기.doc |
| **원안작성자** | 이동욱, 박지현, 박혜란 |
| **수정작업자** | 이동욱, 박지현, 박혜란 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 수정날짜 | 대표수정자 | Revision | 추가/수정 항목 | 내 용 |
| 2014-11-15 | 이동욱 | 1.0 | 최초 작성 | 아이디어 공유 및 향후 계획 설립 |
| 2014-11-18 | 박지현 | 1.1 | 내용 수정 | 계획한 내용에 대한 기능구현 1차 |
| 2014-11-20 | 박혜란 | 1.2 | 내용 수정 | 계획한 내용에 대한 기능구현 2차 |
| 2014-11-22 | 이동욱 | 1.3 | 내용 수정 | 계획한 내용에 대한 기능구현 3차 |
| 2014-11-23 | 박지현 | 1.4 | 보고서 및 PPT | 완성된 프로그램 보고서 및 PPT 최초작성 |
| 2014-11-24 | 박혜란 | 1.5 | 보고서 및 PPT | 완성된 프로그램 보고서 및 PPT 완성 |
|  |  |  |  |  |
|  |  |  |  |  |

**목 차**

[**1** **프로젝트 목표** 4](#_Toc230855220)

[**2** **수행 내용 및 중간결과** 5](#_Toc230855221)

[**2.1** **계획서 상의 연구내용** 5](#_Toc230855222)

[**2.2** **수행내용** 5](#_Toc230855223)

[**3** **최종보고서 본문** 7](#_Toc230855224)

[**4** **자기평가** 8](#_Toc230855225)

# **프로젝트 목표**

**14장의 같은 그림 찾기 게임을 토대로 창의적이고 더욱 재미를 유발 할 수 있게 기능들을 구현하는 것을 목표로 하였다.**

**사전에 정의된 틀은 단순히 아무런 정보 없이 사진을 클릭하여 같은 사진을 찾아내는 등 게임유저로서는 단순 노동에 불과한 작업이었다.**

**여기서 저희 조( 111213 ) 는 가장 중점을 두어야 할 요소를 “흥미”로 두었다.**

**게임은 게임을 하는 사람으로부터 흥미를 이끌어내는 것이 주된 목적이기 때문이다.**

**흥미를 유발할 수 있는 부분이 무엇인지를 고려해보았다.**

**고려한 결과 두 가지의 요소로 나눌 수 있었다.**

**“경쟁”과 “긴박감”이다.**

**시중에 나오는 인기 있는 게임들은 저 두 가지 요소가 반드시 충족된다. 우리도 역시 이 게임 안에서 최대한의 흥미를 유발하기 위해서는 경쟁과 긴박감을 조성해보기로 하였다.**

**랭킹제도를 도입하여 다른 게임유저들과의 우위를 나눌 수 있도록 하여, 재미를 배가 시키고 또한 제한시간을 두어 게임의 긴박감을 조성하는 것이다.**

# **수행 내용 및 중간결과**

## **계획서 상의 연구내용**

**14 장에 구현된 “같은 그림 찾기”를 토대로 아래와 같이 수정 및 추가를 계획하였다.**

1. 난이도를 상 중 하로 분류한 후 게임 실행 시 선택 할 수 있도록 구성하였다. 난이도 “하”는 3x4, “중”은 4x5 마지막으로, “상”은 6x6으로 구성한다.
2. 게임을 시작하기 전 레벨에 따라 전체 그림을 보여주는 기능을 구현한다.
3. 게임의 긴박성을 추가하기 위해 상, 중, 하 에 따라 제한 시간을 지정한다.
4. 게임 유저들의 점수를 기록하고 랭킹을 두어 게임의 흥미를 유발한다.
5. 게임 유저의 편리를 위한 힌트기능을 2가지 추가한다. 첫째는 전체사진을 보여주는 기능을 구현한다. 두 번째는 위에서 구현한 제한시간의 기능을 이용해 추가시간을 주는 힌트를 구현한다.
6. 위에서 구현한 힌트기능을 사용할 시 점수의 차감제도를 준다.
7. 게임 진행의 효율성을 위하여 재시작, 일시정지, 이어하기를 생성한다.
8. 위에서 구현한 랭킹기능을 확인 할 수 있도록 기록 확인기능을 추가한다.

## **수행내용**

Version 1.1:

14장의 소스를 토대로 작성하기 위하여 예제를 참조하여 기본 게임을 구성하였다.

처음 계획하였던 기능들을 실행하기 위해서는 우선순위를 정할 필요가 있다고 생각하여 구현 순서를 정해 보았다.

그 중 첫 번째 기능으로 게임의 난이도 상, 중, 하를 만들고, 게임 유저를 위하여 “난이도”라는 메뉴 디렉토리를 생성하여 3x4 , 4x5, 6x6의 하부 디렉토리를 생성하였다.

게임 진행의 효율성을 위하여 툴바에도 게임의 난이도를 선택할 수 있는 아이콘들을 추가하였다.

위 기능 구현을 한 후 사이즈를 늘리는 동시 사진의수도 늘릴 필요가 있었고 또한 기존의 사진들보다는 다른 사진을 사용하는 것이 좋겠다는 의견을 수렴하여 인터넷에서 만화 캐릭터이미지를 총 17장을 저장한 후 게임의 최적화된 사이즈를 79x81픽셀로 획일화 시켰다.

Version 1.2:

게임을 시작하기 전 레벨에 따라 전체 그림을 보여주는 기능을 구현해보았다. 타이머함수를 이용하였고 구연하는데 어려움이 없었다. 타이머함수를 이용하는 기능들을 한번에 다 구현을 해보기로 하였다. 첫 번째로 남은 시간을 보여주는 바를 만든 후 그 옆에 보기 좋게 남은 시간을 보여주도록 구성하였다. 이것을 토대로 힌트 중 시간추가기능을 구현하기로 했기 때문에 먼저 힌트 디렉토리와 툴바에 링크를 걸어놓고 타이머 함수를 이용하여 시간추가기능을 주었다.(단 일정 시간 경과 후 시간추가를 할 수 있도록 제약을 두었다.)

타이머 기능 중 마지막으로 추가 구현할 부분은 재시작 일시정지 이어하기 기능이다. 일시정지 부분이 애초에 계획했던 기능이었지만, Killtimer 사용하는데 문제가 있어서 시간이 많이 투자되었다.

Version 1.3

게임의 경쟁심을 유발하기 위해 랭크를 구현하기 위해서 맞춘 카드 쌍에 10을 곱한 점수에 빠른 시간 내에 게임을 완료하면 고득점을 받을 수 있도록 하기 위해서 시간도 곱해준다. 그리고 찬스 기능을 사용하면 점수를 조금 깎았다. 그래서 점수 값을 비교하여 높은 점수의 유저이름이 먼저 출력될 수 있도록 하였다.

# **최종보고서 본문**

**1-1. 기본적인 화면 구성을 한다.**

**- 게임 점수와 남은 시간을 숫자와 그래프로 표시한다.**

**>게임 점수 : 그림의 짝을 맞추는 경우 한 쌍 당 점수 10점씩을 획득한다.**

**>게임 시간 : 시간의 흐름에 따라 빨간색 bar가 줄어드는 형태로 표현한다.**

**<부분 코드>**

void CGameView::OnDraw(CDC\* pDC)

{

CGameDoc\* pDoc = GetDocument();

ASSERT\_VALID(pDoc);

if (!pDoc)

return;

CString str;

str.Format(\_T("현재점수:%d/%d"),nMatchCount\*10+minus,pDoc->m\_nRow\*pDoc->m\_nCol/2\*10);

pDC->TextOutW(20,0,str);

CString str1;

str1.Format(\_T("남은 시간:%d"), timer);

pDC->TextOutW(180,0,str1);

CBrush redB,brush,\*pOldB;

brush.CreateSolidBrush(RGB(255,255,255));

pOldB=pDC->SelectObject(&brush);

pDC->SelectStockObject(BLACK\_PEN);

pDC->Rectangle(280,0,280+30\*5,20);

redB.CreateSolidBrush(RGB(255,0,0));

pOldB=pDC->SelectObject(&redB);

pDC->Rectangle(280,0,280+timer\*5,20);

Chance(pDC);

}

**<코드 설명>**

**그림의 짝을 맞췄을 경우 증가하는 ‘mMatchCount’ 변수에 10을 곱하고, 찬스를 쓸 때마다 감소되는 점수로 minus변수를 더하여 점수를 계산한다. 출력은 view 왼쪽 상단에 하도록 한다.**

**또한, 총 만점 점수를 계산하여 몇 점 만점일 때 몇 점을 획득했는지 알 수 있도록 하기 위하여 pDoc->m\_nRow\*pDoc->m\_Col/2\*10을 하였다. 총 그림의 수를 2로 나누어 10을 곱한 격이다.**

**시간의 흐름에 따라 빨간색 bar가 줄어드는 형태로, 남은 시간을 좀 더 눈에 띄게 시각적으로 표현했다. 처음엔 꽉 찬 Red Box로 시작하여 점점 White Box가 늘어나는 형태로 구현하였다.**

**1-2. 게임 화면에 비트맵 이미지 출력하기**

* **상, 중, 하 난이도에 따른 출력되는 비트맵 이미지의 개수를 변경한다.**

**(이미지의 개수는 ‘하’는 6개, ‘중’은 10개, ‘상’은 18개 이므로, 총 18개의 이미지와 이미지의 뒷면을 나타내는 이미지 1개가 필요하다.**

**<부분 코드>**

void CGameView::Chance(CDC\* pDC)

{

CGameDoc\* pDoc = GetDocument();

int nCount = 0;

CDC memDC;

CBitmap\* pOldBmp;

memDC.CreateCompatibleDC (pDC);

for(int n = 0; n < pDoc->m\_nRow; n++)

{

for(int m = 0; m < pDoc->m\_nCol; m++)

{

if(pDoc->m\_bShow[n][m] == true)

{

pDoc->m\_bmp[n][m].LoadBitmap(IDB\_BITMAP1+pDoc->m\_nRnd[nCount]);

pOldBmp = memDC.SelectObject(&pDoc->m\_bmp[n][m]);

//왼쪽과 위쪽에 여백을 주기 위해서 1을 더했다.

pDC->BitBlt(pDoc->m\_bmCell.cx\*(m+1), pDoc->m\_bmCell.cy\*(n+1), //출력될 위치

pDoc->m\_bmCell.cx, pDoc->m\_bmCell.cy,

//그림 하나의 크기

&memDC, 0, 0, SRCCOPY);

pDoc->m\_bmp[n][m].DeleteObject();

}

//서로 다른 그림이 나타나도록 그림의 아이디 값을 1씩 증가시킴

nCount++;

**<코드 설명>**

**GetDocument함수에서 정보를 받아와 pDoc가 가진 게임의 행과 열의 개수를 이용하여 for문을 사용한다. bShow 변수를 사용하여 true라면 이미지가 보이는 상태이기 때문에 해당 이미지를 LoadBitmap으로 불러온다. BitBlt 함수로 pDC에서 memDC로 비트맵블록을 전송한다. 그 위치는 이미지 하나의 크기와 여백을 고려하여 정한다.**

**1-3 비트맵 이미지 랜덤하게 출력하기**

**<부분 코드>**

// 비트맵 이미지를 랜덤하게 출력한다.

if(pDoc->m\_bRandom)

{

nMatchCount=0;

minus=0;

pDoc->OnRandom();

pDoc->m\_bRandom = false;

}

Chance함수안에서 OnRandom함수를 위의 코드로 호출

void CGameDoc::OnRandom(void)

{

srand((unsigned)time(NULL));

BOOL bInsert = TRUE;

int nGrating = m\_nRow \* m\_nCol;

for(int n = 0; n < nGrating; n++)

{

m\_nRnd[n] = (rand() % (nGrating / 2)) + 1;

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

if(m\_nRnd[n] == m\_nRnd[m]){

if(bInsert == FALSE){

n--;

bInsert = TRUE;

break;

}

bInsert = FALSE;

}

}

}

int nCount = 0;

for(int n = 0; n < m\_nRow; n++)

{

for(int m = 0; m < m\_nCol; m++)

{

m\_nBmpID[n][m] = m\_nRnd[nCount];

nCount++;

}

}

}

**<코드 설명>**

**시간을 Seed로 잡아서 똑 같은 난수 발생을 방지하여 이미지를 랜덤하게 출력하도록 한다.**

**총 출력될 이미지의 수로 nGrating변수를 두어 그 수만큼 돌면서 난수를 발생시키다. 같은 이미지가 3번 나오는 것을 방지하기 위해 if(m\_nRnd[n]==m\_nRnd[m])과 if(bInsert==FALSE)로 2가지 조건문을 두어 검사한다.**

**2-1. 게임을 시작하기 전 레벨에 따라 전체 그림을 보여주는 기능을 구현한다.**

**<부분 코드>**

if(pDoc->m\_bShow[n][m] == false)

{

CBitmap bmp;

if(timer>28){ //잠깐 카드 보여주기

pDoc->m\_bmp[n][m].LoadBitmap(IDB\_BITMAP1+pDoc->m\_nRnd[nCount-1]);

pOldBmp = memDC.SelectObject(&pDoc->m\_bmp[n][m]);

pDC->BitBlt(pDoc->m\_bmCell.cx\*(m+1), pDoc->m\_bmCell.cy\*(n+1), //출력될 위치

pDoc->m\_bmCell.cx, pDoc->m\_bmCell.cy, //그림 하나의 크기

&memDC, 0, 0, SRCCOPY);

pDoc->m\_bmp[n][m].DeleteObject();

}

else{

bmp.LoadBitmap(IDB\_BITMAP1);

pOldBmp = memDC.SelectObject(&bmp);

pDC->BitBlt(pDoc->m\_bmCell.cx\*(m+1),pDoc->m\_bmCell.cy\*(n+1),//출력될 위치 pDoc->m\_bmCell.cx, pDoc->m\_bmCell.cy, //그림 하나의 크기

&memDC, 0, 0, SRCCOPY);

bmp.DeleteObject();

}

}

}//for

}//for

memDC.SelectObject(pOldBmp);

pDoc->ResizeWindow();

}

**<코드 설명>**

게임 시작 전에 레벨에 따른 전체 이미지를 보여주고 시작하도록 한다. 그래서 timer가 28초보다 클 때 잠깐 카드를 보여주기 위해 조건문을 두어 출력한다. 코드는 Chance함수에 구현했다.

**2-2. 메인프레임의 크기를 변경한다.**

**<부분 코드>**

void CGameDoc::ResizeWindow(void)

{

CMainFrame\* pMain = (CMainFrame\*)AfxGetMainWnd();

CREATESTRUCT st;

CREATESTRUCT& rst = st;

pMain->PreCreateWindow(rst);

rst.lpszName = \_T("Game");

rst.cx = (m\_nRow + 1) \* m\_bmCell.cx + 180;

rst.cy = (m\_nCol + 1) \* m\_bmCell.cy + 180;

CRect rect;

pMain->GetClientRect(rect);

pMain->MoveWindow(rect.left, rect.top, rst.cx, rst.cy);}

**<코드 설명>**

**CGameDoc클래스에 ResizeWindow함수를 구현하여 메인프레임의 크기를 변경시키는 것을 구현하였다.**

**3. 게임의 긴박성을 추가하기 위해 상, 중, 하 에 따라 제한 시간을 지정한다.**

**<부분 코드>**

void CGameView::On3x4()

{

CGameDoc\* pDoc = GetDocument();

pDoc->m\_nRow = 3;

pDoc->m\_nCol = 4;

pDoc->m\_bRandom=true;

for(int n = 0; n < pDoc->m\_nRow; n++)

for(int m = 0; m < pDoc->m\_nCol; m++)

pDoc->m\_bShow[n][m] = false;

pDoc->m\_nBmpFirstID = pDoc->m\_nBmpSecondID = 0;

pDoc->m\_bMouse = false;

timer = 30;

Invalidate();

}

**<코드 설명>**

**메뉴 툴바에 난이도에 따른 메뉴를 추가하여 ID\_3x4, ID\_4x5, ID\_6x6으로 ID를 지정하고, 이벤트 처리 함수를 추가하였다. 그래서 그 함수에는 timer에 난이도에 따른 시간을 ‘하’는 30초, ‘중’은 32초, ‘상’은 35초로 지정했다.**

**4. 게임 유저들의 점수를 기록하고 랭킹을 두어 게임의 흥미를 유발한다.**

**<부분 코드>**

m\_people++;

m\_score[m\_people] = timer \* nMatchCount \* 10+ minus;

if(timer==0)

m\_score[m\_people] = nMatchCount \* 10+ minus;

OnShowDlg();

**<코드 설명>**

**랭킹을 저장할 때 게임 유저의 수를 증가시키고 m\_score라는 게임 점수를 저장하는 배열에 시간이 0일때는 시간을 고려하지 않고 저장하고, 0이 아닐 때는 남은 시간 값도 곱하여 빠르게 게임을 완료할수록 고득점 할 수 있도록 하였다. minus변수는 찬스를 사용하면 차감되는 점수를 의미한다. 자세한 설명은 뒤에서 하겠다.**

void CGameView::OnShowDlg()

{

dlgName.DoModal();

m\_str[m\_people] = dlgName.m\_nName;

for(int i =1; i<=m\_people; i++)

{

dlgRank.m\_strList[m\_people] = m\_str[m\_people];

dlgRank.m\_strScore[m\_people] = m\_score[m\_people];

}

dlgRank.mem=m\_people;

dlgName.m\_nName = "";

dlgRank.DoModal();

}

**<코드 설명>**

**m\_str은 이름을 저장하는 CString형 배열이고, m\_strList에 m\_str을 받아 저장한다. m\_strScore는 int형 배열이고 m\_score을 받아 저장한다.**

**5. 게임 유저의 편리를 위한 힌트기능을 2가지 추가한다.**

**5-1 첫째는 전체사진을 보여주는 기능을 구현한다.**

**<부분 코드>**

void CGameView::OnChance1()

{

CDC \*pDC = GetDC();

int a=timer;

timer=30;

Chance(pDC);

timer=a;

minus-=3;

ReleaseDC(pDC);

}

**<코드 설명>**

**onChance()의 parameter로 pDC가없기 때문에 CDC \*pDC=GetDc(); 로 받아왔다.**

**그러하여 전체사진 보여주기를 누르면 시간은 현재시간을 유지해야하기 때문에 눌렀을 당시의 시간을 저장하고 찬스함수가 실행되는순간 시간을 눌렀던 시간으로 초기화시킨다.**

**그리고 점수의 차감제도를 도입하였기 때문에 점수를 합산할 때 전체 보여주기를 눌렀을 때 점수를 3점 감점하기로 기준을 세웠다. minus변수는 힌트를 눌렀을때에는 항상 불러오는 변수이다.**

**5-2 위에서 구현한 제한시간의 기능을 이용해 추가시간을 주는 힌트를 구현한다.**

**<부분 코드>**

void CGameView::OnTimer(UINT\_PTR nIDEvent)

{

CClientDC dc(this);

Invalidate();

if(timer>0){

timer--;

}

else if(timer<=0) {

OnDestroy();

OnSuccess();

}

CView::OnTimer(nIDEvent);}

**<코드 설명>**

**처음 제한시간을 각 케이스별로 30, 32,35로 초기화시켰다. 시간이 0보다 크다는 조건에서는 계속 시간을 줄어들게 한다.**

**또한 시간이 0이되었을 때 게임을 종료하기위해 OnDestroy() 와 OnSuccess() 를 불러온다.**

**<부분 코드>**

void CGameView::OnTimeplus()

{

if(timer>20)

int res=AfxMessageBox(\_T("시간 찬스를 사용할 수 없습니다."), MB\_OK);

else{

timer+=5;

minus -= 5;

}

}

**<코드 설명>**

**무한한 게임시간 추가를 미연에 방지하기 위하여 모든 테스트케이스에서 시간이 20초보다 작을때만 돌아 갈 수 있도록 구현하였다. 시간은 현재시간을 받아 +5초를 주고 점수는 minus 변수를 이용하여 5점을 감점한다.**

**6. 위에서 구현한 힌트기능을 사용할 시 점수의 차감제도를 준다.**

**<부분 코드>**

class CGameView : public CView

{

public:

//현재 마우스가 좌표값으로 그림의 위치를 계산하여 저장하고 있는 인덱스

UINT m\_nRowTempIndex;

UINT m\_nColTempIndex;

int nMatchCount;

int minus;

**<코드 설명>**

**위처럼 마이너스변수를 선언하였다.**

**<부분 코드>**

void CGameView::OnSuccess(void)

{

KillTimer(1);

int res2 = AfxMessageBox(\_T("GAME OVER! 점수를 등록하시겠습니까?"),MB\_YESNO);

if(res2 == IDYES)

{

m\_people++;

m\_score[m\_people] = timer \* nMatchCount \* 10+ minus;

if(timer==0)

m\_score[m\_people] = nMatchCount \* 10+ minus;

OnShowDlg();

}

**…**

**<코드 설명>**

**마이너스 점수를 힌트가 누를 시 계속 누적되게 초기화를 시켜준 후 마지막 게임이 끝나고 점수를 등록할 때에 축적된 minus값을 더해주어 결론적으로는 차감이 될 수 있도록 구성하였다.**

**이외에도 view창에 나오는 현재 점수 칸에도 위와 같이 동일하게 minus를 더해주었다.**

**7. 게임 진행의 효율성을 위하여 재시작, 일시정지, 이어하기를 생성한다.**

**7-1. 재시작 기능**

**<부분 코드>**

void CGameView::OnRestart()

{

CGameDoc\* pDoc = GetDocument();

if(pDoc->m\_nCol==4)

timer= 30;

else if(pDoc->m\_nCol==5)

timer=32;

else if(pDoc->m\_nCol==6)

timer=35;

else

timer=30;

pDoc->m\_bRandom=true

SetTimer(1,1000,NULL);

for(int n = 0; n < pDoc->m\_nRow; n++)

for(int m = 0; m < pDoc->m\_nCol; m++)

pDoc->m\_bShow[n][m] = false;

pDoc->m\_nBmpFirstID = pDoc->m\_nBmpSecondID = 0;

pDoc->m\_bMouse = false;

Invalidate();

}

**<코드 설명>**

**Case by case 로 각 난이도마다 시간을 지정해줘야하기 때문에 if else문을 이용하여 현재 난이도를 파아한후 timer값을 초기화하였다.**

**그 후 그 난이도의 값마다 뒤집힌 상태로 초기화 시킨다. 또한 카드가 눌리지않은 상태이기 때문에**

pDoc->m\_nBmpFirstID = pDoc->m\_nBmpSecondID = 0 **역시 초기화 시켰다.**

**7-2. 일시정지 기능**

KillTimer(1);

**<코드 설명>**

**단순하게 KillTiemr(1)를 추가하게되면 pause기능을 구현할 수 있었다.**

**7-3. 이어하기 기능**

SetTimer(1,1000,NULL); //OnTimer함수호출

**<코드설명>**

**onTimer함수를 호출한다.**

**8. 위에서 구현한 랭킹기능을 확인 할 수 있도록 기록 확인기능을 추가한다.**

**<부분 코드>**

void CGameDoc::Serialize(CArchive& ar)

{

CGameView\* pView=(CGameView\*)m\_viewList.GetHead();

int m\_people;

if (ar.IsStoring())

{

m\_people = pView->m\_people;

ar << pView->m\_people;

for(int i = 1; i<=pView->m\_people; i++)

ar<<pView->m\_str[i]<<pView->m\_score[i];

}

else

{

OnNewDocument();

ar >> pView->m\_people;

pView->dlgRank.mem = pView->m\_people;

for(int i = 1; i<=pView->m\_people;i++)

{

ar>>pView->m\_str[i]>>pView->m\_score[i];

pView->dlgRank.m\_strList[i]=pView->m\_str[i];

pView->dlgRank.m\_strScore[i]=pView->m\_score[i];

}

}

}

**<코드 설명>**

**OnSuccess함수를 실행하면 m\_people변수가 증가한다. 그래서 mem변수는 m\_people을 받아 랭킹 등록한 유저의 수를 저장한다. 그래서 게임 유저의 이름과 점수를 if(ar.IsStoring()) 조건문에서 저장하고, else에서 로딩한다.**

BOOL RANKING::OnInitDialog()

{

CDialogEx::OnInitDialog();

CString str;

for(int i = 1; i<=mem; i++)

{

for(int j = i+1; j<=mem; j++)

{

if(m\_strScore[i] < m\_strScore[j])

{

int temp=m\_strScore[i];

m\_strScore[i]=m\_strScore[j];

m\_strScore[j]=temp;

CString temp2=m\_strList[i];

m\_strList[i]=m\_strList[j];

m\_strList[j]=temp2;

}

}

}

for(int i=1; i<=mem; i++){

m\_ctrList.AddString(m\_strList[i]);

str.Format(\_T("%d"),m\_strScore[i]);

m\_ctrScore.AddString(str);

}

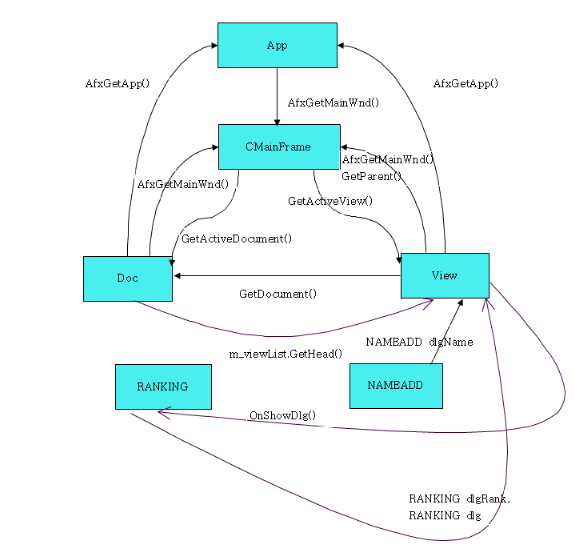
return TRUE;

}

**<코드 설명>**

**mem이 랭킹등록한 게임유저의 수를 의미하므로 for문으로 그 사람의 수만큼 도는데, 점수를 배열하기 위해 비교하여 중간값을 temp에 저장하고 CListBox형인 m\_ctrList와 m\_ctrScore에 AddString으로 추가하였다.**

**\*\*\* 클래스 참조 \*\*\***

****

# **자기평가**

**-이동욱(20113304)-**

**처음에 수립한 목표 “흥미”를 유발하기 위해 아이디어를 얻어내는 것이 쉽지 않았다. 정해져 있는 “같은 그림 찾기”라는 게임의 틀 속에 구현할 수 있는 범위 안에서 창의성을 발휘하는 것이 쉬운 일이 아니었다.**

**최초에 설립하였던 아이디어들을 모두 구현하는 것 역시 쉬운 일이 아니었다. 특수한 테스트 케이스 때문에 예외를 발견하고 수정하는 것이 이번 프로젝트에서 가장 어려운 부분이었던 것 같다.**

**하지만 이번 프로젝트를 통하여 지금까지 수업시간에 배워 온 것과 과제로 습득한 기능들을 쓰는데 문제가 없었다. 또한 이번 기회로 기존에 있었던 기능들에 대하여 다시금 습득할 수 있는 좋은 기회였다.**

**-박혜란(20123367)**

**처음 이 프로젝트를 구성할 때에는 어떤 기능을 넣어야 좋을까 많이 고민되었는데 함께 토론하며 어떻게 하면 게임을 흥미롭게 만들어 게임을 하는 사람에게 즐거움을 느끼게 해줄까에 주안점을 주다 보니 기능들이 하나 둘 씩 추가 되게 되었다. 수업시간에 배웠던 내용들을 바탕으로 구현하다 보니 큰 문제없이 순조롭게 진행되게 되었다. 이번 프로젝트를 통하여 이제까지 배웠던 내용들을 복습할 수 있는 시간이 되어서 좋았고 다 완성된 후에 실행해보니 뿌듯하기도 했고 보람 있는 시간이 된 것 같다.**

**-박지현(20133224)**

**같은 그림 찾기라는 게임을 직접 프로그래밍 한다는 것이 처음부터 흥미로웠다. 처음 게임을 기획할 때는 창의적으로 다양한 기능을 가진 게임을 만들고 싶었고 할 수 있을 것 같았다. 하지만 생각했던 것을 모두 구현하는 것이 쉬운 일이 아니었다. 그래서 어려운 점이 생길 때마다 수업시간에 배운 내용을 생각하고, 책을 참조하고 검색하면서 해결책을 찾으며 해결하려고 노력했다. 여태까지는 게임을 만드는 것처럼 큰 프로젝트를 하지 않아서 큰 보람을 느끼기 어려웠는데 이번 기말 프로젝트를 통해 제대로 완성된 게임을 하나 만든 것 같아서 뿌듯했다.**