

2022학년도 자바 프로젝트 완료 보고서

복권 시스템

2022년 12월 14일

컴퓨터정보과

팀명	복권사랑
팀장	김재엽(202144030)
팀원	박민우(201944031)



인하공업전문대학
INHA TECHNICAL COLLEGE

2022년도
컴퓨터정보과 특성화 사업
자바 프로젝트 진행 결과보고서
(분야 : 윈도우 응용프로그램)

연구과제명 : 복권 윈도우 응용프로그램 개발

2022. 12. 14



인하공업전문대학
INHA TECHNICAL COLLEGE

본 결과물은 인하공업전문대학 컴퓨터정보과 자바 프로젝트 수업의 연구 결과입니다.

제 출 문

- 분 야 : 자바 윈도우 응용프로그램
- 연구과제명 : 복권 윈도우 응용프로그램
- 프로젝트 책임자 : 인하공업전문대학 컴퓨터정보과 김재엽, 박민우

2022년도 인하공업전문대학 컴퓨터정보과 자바 프로젝트 수업의
결과물로 이 보고서를 제출합니다.

2022. 12. 14

프로젝트 책임자 : 컴퓨터정보과(학번) 김 재 엽 (인)

공동 참여자 : 컴퓨터정보과(학번) 박 민 우 (인)

인하공업전문대학 컴퓨터정보과 학과장 귀하

프로젝트 요약문

분 야	자바 윈도우 응용프로그램 개발
프로젝트명	복권 시스템
프로젝트 책임자	컴퓨터정보과(202144030) 김 재 업 컴퓨터정보과(201944031) 박 민 우
<p>연 구 내 용</p> <p>복권 시스템의 유저 친화적인 GUI 설계를 위해 Windows Java Swing을 활용하여 GUI 설계</p> <p>보안을 위해 로그인 절차에 Chapcha 시스템을 추가하여 차용</p> <p>사용자의 데이터 및 로또 당첨 내역 조회를 위한 데이터를 Microsoft SQL을 이용하여 DB 설계 및 데이터 저장</p> <p>이러한 DB설계 과정에서 사용자의 민감한 개인정보를 암호화를 위해 유저의 개인정보 데이터와 Salt (무작위 랜덤 문자열)을 자바 MessageDigest 라이브러리의 SHA256 (Secure Hash Algorithm)라는 복호화가 불가능한 단방향 암호화 기법을 활용하여, 두 개의 값을 합쳐 해커들이 공격으로부터 유저들의 데이터를 안전하게 저장</p>	

참여자 인적사항

1. 프로젝트 팀장

성명	김재엽	학번	202144030
학과	컴퓨터정보과		
연락처	H.P	010 4196 5219	
	E-mail	rlawo5219@naver.com	

2. 프로젝트 팀원

성명	박민우	학번	201944031
학과	컴퓨터정보과		
연락처	H.P	010 2326 7886	
	E-mail	psmcm123@gmail.com	

목 차

1. 프로젝트 목적	1
2. 프로젝트 목표와 기대 효과	1
3. 프로젝트 진행범위 및 방법	1
4. 프로젝트 주요 내용	3
5. 연구 결과물	5
5-1. 프로젝트명	5
5-2. 프로젝트 개요	5
5-3. 프로젝트 수행 세부 일정 및 내용	6
5-4. 프로젝트 결과	16
- 부 록 -	17
사용자 설명서	19

1. 프로젝트 목적

사용자의 실제 복권 구매 심리완화

2. 프로젝트 목표 및 기대효과

이 복권 시뮬레이션을 통해 사용자가 실제 복권을 구매하기 전에 당첨 확률을
거시적으로 보여주어 실제 로또 구매를 한 번 더 고려하게 하는 것이 이번
프로젝트의 목표이자 기대효과.

3. 프로젝트 진행범위 및 방법

기존의 사이트 회원가입 및 로그인 화면을 벤지마킹하여 회원가입 페이지 설계와 로그인
페이지, 캡차화면 구현. 자동/수동 복권을 선택 가능하며 자동 복권 선택 시 랜덤
번호 6개를 자동선택 후 당첨번호와 랜덤번호 출력하여 번호 일치율과 당첨 금액 출력.
수동 복권도 위의 과정과 동일하나, 번호를 사용자가 임의의 번호 6개 선택이 가능.
이러한 사용자 로또 정보가 DB에 저장되어 사용자 정보 조회에서 최근 15개의
당첨내역을 조회기능과 회원탈퇴 및 금액이 0원이 됐을 시 금액충전이 가능 구현.

4. 프로젝트 주요 내용

Java Swing을 이용하여 로또 시스템 제작

MsSQL DB 연동 후 사용자 정보 저장 및 불러오기 (로그인 회원가입)

사용자의 민감한 개인정보의 암호화

로또 당첨 금액 및 내역을 DB에 저장하여 조회 기능 구현

5. 프로젝트 결과물

5-1. 프로젝트명

로또 시스템

5-2. 프로젝트 개요

실제 로또와 동일한 자동/수동 복권 시스템 제작

5-3. 프로젝트 수행 세부일정 및 내용

■ 프로젝트 수행 세부 일정

프로젝트 진행 계획(2022. 08. 29. - 2022. 12. 14)							
진 행 내 용	책임자	8	9	10	11	12	비고
벤치마킹	김재업		완료				-
GUI 디자인	박민우			완료			-
회원가입 및 로그인	김재업				완료		-
로또 기능 구현	김재업				완료		-
DB설계 및 암호화	박민우				완료		-

■ 프로젝트 수행 내용

벤치마킹

22.09.30

로또

45

로또6/45 당첨정보

당첨번호

당첨내역

지급안내

추첨방송

로또6/45 당첨번호

1045회 (2022년 12월 10일 추첨)

6

14

15

19

21

41

+

37

등위	당첨 복관수	총 당첨금
1등	13개	25,870,785,759 원
	1개당 당첨금	1,990,060,443 원
	자동 10개, 수동 2개, 번자동 1개	
2등	67개	4,311,797,643 원
	1개당 당첨금	64,355,189 원
	2,699개	4,311,798,246 원
3등	1개당 당첨금	1,597,554 원
	135,872개	6,793,600,000 원
	1개당 당첨금	50,000 원
4등	2,280,432개	11,402,160,000 원
	1개당 당첨금	5,000 원
5등		

동행 복권 사이트 (<https://m.dhlottery.co.kr/common.do?method=main>)

기존 로또의 경우 6개의 번호와 1개의 보너스 번호가 있는 반면,
이번 프로젝트의 경우 6개의 당첨번호 만을 만들었고, 2개 일치 시 로또 구매금액인 1000원을
환급해주는 것으로 변경하여 실제 로또보다, 기댓값이 더 크게 만들었다.

UI 제작

10월 1일 ~ 10월 8일

Java Swing을 이용하여 UI를 제작하였음, 개발 과정에서 지속적으로 수정/보완



DB연동 회원가입, 로그인 구현

11월 1일 ~ 11월 11일

MsSQL을 이용하여 DB 설계 및 구축한 후, 회원가입 기능 및 로그인 기능 구현완료 후
사용자 정보를 암호화하여 DB에 저장.



자동/수동 로또 및 마이페이지 기능 구현

11월 14일 ~ 11월 30일

사용자의 로또 당첨 내역을 DB에 저장한 후 이 정보를 불러와 마이페이지에 보여주는 기능 구현

마이페이지

logout

BACK

ID : id

잔액 : 포인트

회차	당첨번호	복권유형	일치	당첨금액
----	------	------	----	------

금액 충전 ex) 충전 시 10만원이 충전됩니다.

회원 탈퇴

5-4. 프로젝트 결과

1. 메인화면

1-1 잘못된 로그인 시도



1-2 로그인 2회 실패 시 캡차출력

1-3 잘못된 캡차 입력 시



2. 회원가입 화면



회원가입

아이디 중복확인

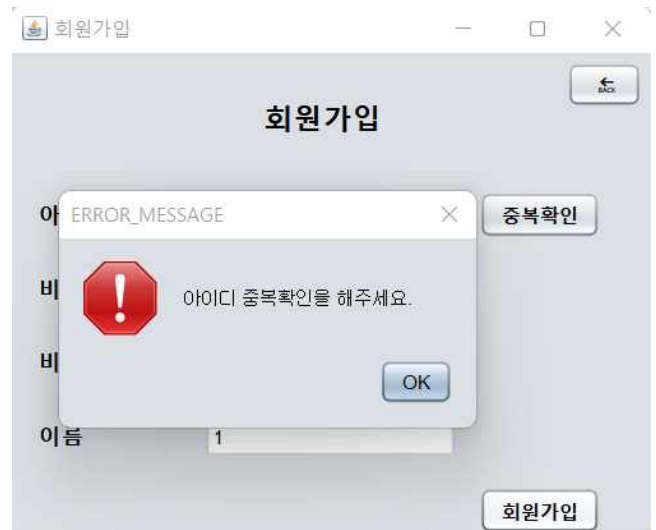
비밀번호

비밀번호 확인

이름

회원가입

2-1. 중복검사 안하고 회원가입 시도



회원가입

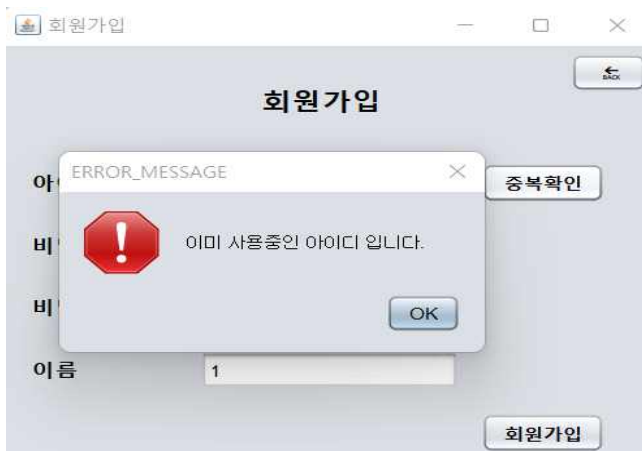
ERROR_MESSAGE

아이디 중복확인을 해주세요.

OK

회원가입

2-2. 이미 사용중인 아이디 중복검사



회원가입

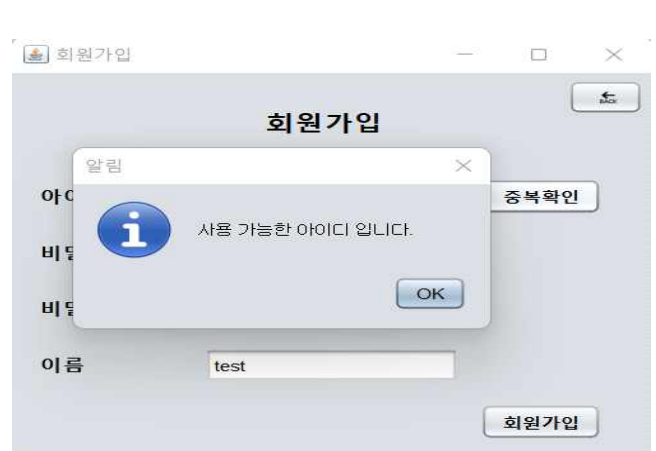
ERROR_MESSAGE

이미 사용중인 아이디입니다.

OK

회원가입

2-3. 사용 가능한 아이디 중복검사



회원가입

알림

사용 가능한 아이디입니다.

OK

회원가입

2-3 정상적인 가입 완료 및 로그인 시도



로그인

AUTO/MANUAL Lottery

ID :

PASSWD :

로그인

회원가입

3. 로그인 완료 시 출력화면



로또 ID : test

마이페이지

logout

로또

47, 24, 17, 51, 3, 33, 3, 22, 61

자동

수동

4. 자동버튼 클릭 후 화면 및 복권참여

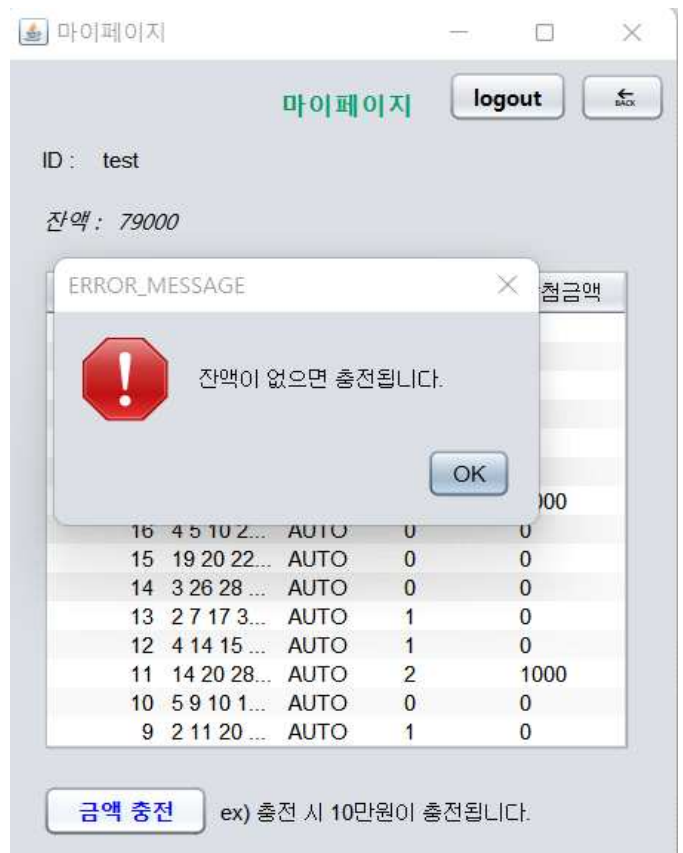
5. 수동버튼 클릭 후 화면 및 복권참여

5-1. 수동번호를 6개 초과 선택 시

5-2. 수동번호를 6개 미만 선택 시

6. 마이페이지

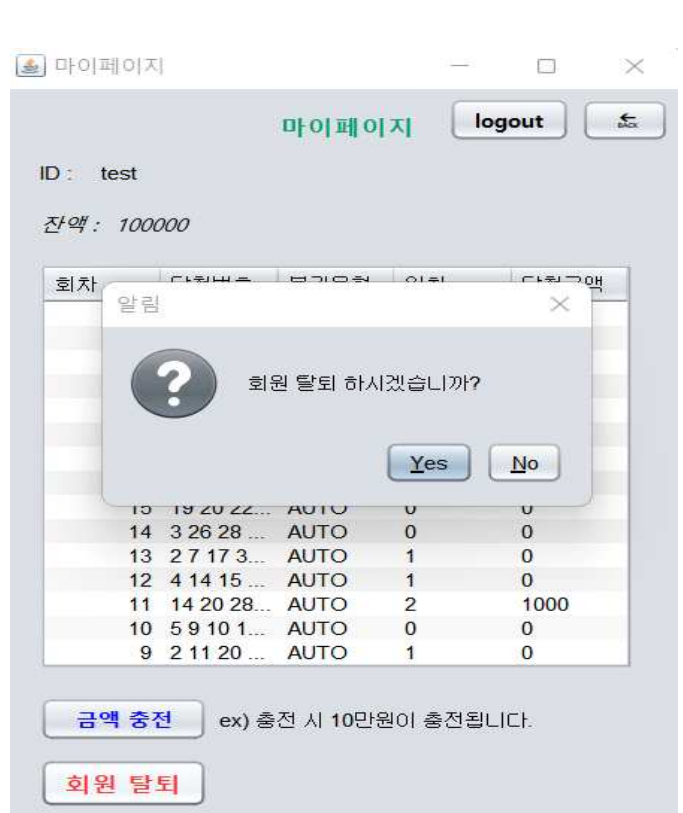
6-1. 잔액이 있을 때 충전 시도



6-2. 잔액이 없을 때 충전 시도



6-3. 회원탈퇴 버튼 클릭 시



7-1 사용자 정보 DB 및 암호화

	ID	Password	Name	Money	Salt
▶	1	7e4c90dbae...	test	95000	f6b752e6ef...
	test	d7b21b5c0...	test	79000	018e5ba235...
*	<i>NULL</i>	<i>NULL</i>	<i>NULL</i>	<i>NULL</i>	<i>NULL</i>

7-2 사용자 로또 당첨내역 DB

	Round	Numbers	LType	Matched	Prize	ID
	2	3 22 23 25 ...	AUTO	1	0	test
	3	1 14 20 30 ...	AUTO	1	0	test
	4	4 7 10 11 1...	AUTO	0	0	test
	5	5 9 13 20 2...	AUTO	0	0	test
	6	10 17 21 22...	AUTO	1	0	test
	7	1 32 36 39 ...	AUTO	0	0	test
	8	5 15 32 39 ...	AUTO	1	0	test
	9	2 11 20 40 ...	AUTO	1	0	test
	10	5 9 10 13 2...	AUTO	0	0	test
	11	14 20 28 30...	AUTO	2	1000	test
	12	4 14 15 16 ...	AUTO	1	0	test
	13	2 7 17 37 3...	AUTO	1	0	test
	14	3 26 28 33 ...	AUTO	0	0	test
	15	19 20 22 29...	AUTO	0	0	test
	16	4 5 10 28 3...	AUTO	0	0	test
	17	15 17 19 24...	AUTO	2	1000	test
	18	1 5 7 11 25 ...	AUTO	0	0	test
	19	8 17 18 20 ...	AUTO	0	0	test
	20	4 8 18 19 2...	AUTO	0	0	test
	21	1 19 28 31 ...	AUTO	0	0	test
	22	4 6 20 27 3...	AUTO	1	0	test
	23	13 17 19 22...	Manual	0	0	test

- 프로젝트 소스 •

DB_MAN.java

```
import java.sql.*;
import java.io.*;

public class DB_MAN{
    String strDriver = "com.microsoft.sqlserver.jdbc.SQLServerDriver";
    String strURL = "jdbc:sqlserver://localhost:1433;DatabaseName=UserInfo;";
    String strUser = "sa";
    String strPWD = "inha1958";

    Connection DB_con;
    Statement DB_stmt;
    ResultSet DB_rs;

    public void dbOpen() throws IOException {
        try{
            Class.forName(strDriver);
            strURL += "encrypt=true;trustServerCertificate=true;";
            DB_con = DriverManager.getConnection(strURL, strUser, strPWD);
            DB_stmt = DB_con.createStatement();
        }catch(Exception e){
            System.out.println("SQLException : " + e.getMessage());
        }
    }

    public void dbClose() throws IOException {
        try{
            DB_stmt.close();
            DB_con.close();
        }catch(SQLException e){
            System.out.println("SQLException : " + e.getMessage());
        }
    }
}
```

UserIDC.java (아이디 저장 위한 클래스)

```
public class UserIDC {
    static String user_id = "";
}
```


Utils.java (DB 암호화를 위한 클래스)

```
import java.security.MessageDigest;
import java.util.Random;

public class Utils {
    public static String getSalt() {
        Random random = new Random();
        byte[] salt = new byte[10];

        random.nextBytes(salt);

        StringBuffer sb = new StringBuffer();

        for(int i=0; i<salt.length; i++) {
            sb.append(String.format("%02x", salt[i]));
        }

        return sb.toString();
    }

    public static String getEncrypt(String pwd, String salt) {

        byte[] salts = salt.getBytes();
        String result = "";

        byte[] temp = pwd.getBytes();

        try {
            MessageDigest md = MessageDigest.getInstance("SHA-256");
            md.update(temp);
            md.update(salts);

            byte[] b = md.digest();

            StringBuffer sb = new StringBuffer();

            for(int i=0; i<b.length; i++) {
                sb.append(Integer.toString((b[i] & 0xFF) + 256, 16).substring(1));
            }

            result = sb.toString();

        } catch (Exception e) {
            e.getMessage();
        }
        return result;
    }
}
```

LoginFrame.java (로그인 프레임)

```
import java.util.List;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.Collections;
import javax.swing.JOptionPane;

public class LoginFrame extends javax.swing.JFrame {
    DB_MAN DBM = new DB_MAN();
    String strSQL = "Select id From Users ";
    int loginFailCnt = 0;
    int againFail = 0;
    boolean isRight = false;

    public String Random(){
        String[] randomNumber = new String[6];
        ArrayList<String> num = new ArrayList<String>();
        char[] eng = new char[26];

        String[] alphabet = "ABCDEFGHIJKLMNOPQRSTUVWXYZ".split("");
        for(int i = 1 ; i <= 9 ; i++) {
            num.add(Integer.toString(i));
        }
        for(int i = 0; i < 26; i++){
            eng[i] = ((char)(i + 97));
        }
        List<String> alphabetList = Arrays.asList(alphabet);
        Collections.shuffle(num);
        Collections.shuffle(alphabetList);
        for(int i = 0 ; i < 3 ; i++) {
            randomNumber[i] = num.get(i);
            randomNumber[i+3] = alphabetList.get(i);
        }
        List<String> listRandomNumber = Arrays.asList(randomNumber);
        Collections.shuffle(listRandomNumber);

        String strRandomNumber = listRandomNumber.toString().replaceAll("[^0-9^A-Z ]","");
        return strRandomNumber;
    }

    public LoginFrame() {
        initComponents();
        this.setLocation(300, 300);
        lblRandom.setVisible(false);
        txtInput.setVisible(false);
        btnRefresh.setVisible(false);
    }

    private void btnloginActionPerformed(java.awt.event.ActionEvent evt) {
        if (txtID.getText().isEmpty()){
            JOptionPane.showMessageDialog(null, "아이디를 입력하세요.", "ERROR_MESSAGE",
```

```

JOptionPane.ERROR_MESSAGE);
    return;
}

    if (txtPasswd.getText().isEmpty()){
        JOptionPane.showMessageDialog(null, "비밀번호를 입력하세요.", "ERROR_MESSAGE",
JOptionPane.ERROR_MESSAGE);
        return;
    }

    strSQL = "Select * From Users Where Users.ID = '" + txtID.getText() + "'";

    try{

        DBM.dbOpen();
        DBM.DB_rs = DBM.DB_stmt.executeQuery(strSQL);
        String id = "";
        String pass = "";
        String salt = "";

        while(DBM.DB_rs.next()){
            id = DBM.DB_rs.getString("ID");
            pass = DBM.DB_rs.getString("Password");
            salt = DBM.DB_rs.getString("Salt");
        }

        if (loginFailCnt > 0){
            againFail ++;
            lblRandom.setVisible(true);
            txtInput.setVisible(true);
            btnRefresh.setVisible(true);
            if (loginFailCnt > 1){
                String random = lblRandom.getText();
                String input = txtInput.getText().toUpperCase();
                String random2 = random.replace(" ", "");
                if (!(input.equals(random) || input.equals(random2)) ){
                    JOptionPane.showMessageDialog(null, "아래 보이는 문자를 정확하게 입력해주세요!",
"ERROR_MESSAGE", JOptionPane.ERROR_MESSAGE);
                    String strRandomNumber = Random();
                    lblRandom.setText(strRandomNumber);
                    txtInput.setText("");
                    return;
                }
            }
        }

        if(id.equals("")){
            JOptionPane.showMessageDialog(null, "존재하지 않은 아이디입니다.", "ERROR_MESSAGE",
JOptionPane.ERROR_MESSAGE);
            loginFailCnt++;
            DBM.DB_rs.close();
            DBM.dbClose();

```

```

        if (againFail > 0){
            String strRandomNumber = Random();
            lblRandom.setText(strRandomNumber);
        }
        txtInput.setText("");
        return;
    }

    String pwdgetEncrypt = Utils.getEncrypt(txtPasswd.getText().trim(), salt);

    if(!(pass.equals(pwdgetEncrypt))){
        JOptionPane.showMessageDialog(null, "비밀 번호를 잘못 입력 하였습니다.",
"ERROR_MESSAGE", JOptionPane.ERROR_MESSAGE);
        loginFailCnt++;
        DBM.DB_rs.close();
        DBM.dbClose();
        if (againFail > 0){
            String strRandomNumber = Random();
            lblRandom.setText(strRandomNumber);
        }
        txtInput.setText("");
        return;
    }

    loginFailCnt = 0;
    UserIDC.user_id = txtID.getText();
    ChoiceFrame sf = new ChoiceFrame();
    sf.setVisible(true);
    sf.setLocation(300, 300);
    this.dispose();

    DBM.DB_rs.close();
    DBM.dbClose();
} catch (Exception e){
    System.out.println("SQLException : " + e.getMessage());
}
}

private void btnMemActionPerformed(java.awt.event.ActionEvent evt) {

    JoinMemberFrame mf = new JoinMemberFrame();
    mf.setVisible(true);
    mf.setLocation(300, 300);
    this.dispose();
}

private void btnRefreshActionPerformed(java.awt.event.ActionEvent evt) {
    String strRandomNumber = Random();
    lblRandom.setText(strRandomNumber);
}

public static void main(String args[]) {

```

```

        try {
            for (javax.swing.UIManager.LookAndFeelInfo info :
                javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {
                    javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
                }
            }
        } catch (ClassNotFoundException ex) {
            java.util.logging.Logger.getLogger(LoginFrame.class.getName()).log(java.util.logging.Level.SEVERE,
null, ex);
        } catch (InstantiationException ex) {
            java.util.logging.Logger.getLogger(LoginFrame.class.getName()).log(java.util.logging.Level.SEVERE,
null, ex);
        } catch (IllegalAccessException ex) {
            java.util.logging.Logger.getLogger(LoginFrame.class.getName()).log(java.util.logging.Level.SEVERE,
null, ex);
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {
            java.util.logging.Logger.getLogger(LoginFrame.class.getName()).log(java.util.logging.Level.SEVERE,
null, ex);
        }
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new LoginFrame().setVisible(true);
            }
        });
    }
}

```

JoinMemberFrame.java (회원가입 프레임)

```

import java.security.MessageDigest;
import java.util.Arrays;
import java.util.Random;
import javax.swing.JOptionPane;

public class JoinMemberFrame extends javax.swing.JFrame {
    DB_MAN DBM = new DB_MAN();
    String strSQL = "Select id From Users ";

    public JoinMemberFrame() {
        initComponents();
    }

    int reBtnClick = 1;
    String idText = "";

    private void btnMemberActionPerformed(java.awt.event.ActionEvent evt) {
        if(txtID.getText().equals("")){
            JOptionPane.showMessageDialog(null, "아이디를 입력하세요.", "ERROR_MESSAGE",
JOptionPane.ERROR_MESSAGE);
            return;
        }else if(!(idText.equals(txtID.getText()))) || reBtnClick != 0){

```

```

        JOptionPane.showMessageDialog(null, "아이디 중복확인을 해주세요.", "ERROR_MESSAGE",
JOptionPane.ERROR_MESSAGE);
        return;
    }else if(txtPass.getText().equals("")){
        JOptionPane.showMessageDialog(null, "패스워드를 입력하세요.", "ERROR_MESSAGE",
JOptionPane.ERROR_MESSAGE);
        return;
    }else if(txtPassCheck.getText().equals("")){
        JOptionPane.showMessageDialog(null, "패스워드 확인란을 입력하세요.", "ERROR_MESSAGE",
JOptionPane.ERROR_MESSAGE);
        return;
    }else if(txtName.getText().equals("")){
        JOptionPane.showMessageDialog(null, "이름을 입력하세요.", "ERROR_MESSAGE",
JOptionPane.ERROR_MESSAGE);
        return;
    }
}

String pwd = txtPass.getText().trim();
String salt = Utils.getSalt();
String pwdgetEncrypt = Utils.getEncrypt(pwd, salt);

if(Arrays.equals(txtPass.getPassword(),txtPassCheck.getPassword())){
    strSQL = "Insert Into Users Values (";
    strSQL += "'" + txtID.getText().trim() + "',";
    strSQL += "'" + pwdgetEncrypt + "',";
    strSQL += "'" + txtName.getText().trim() + "',";
    strSQL += "100000 + ",";
    strSQL += "'" + salt + "'";

    try{
        DBM.dbOpen();
        DBM.DB_stmt.executeUpdate(strSQL);
        DBM.dbClose();
    }catch(Exception e){
        System.out.println("SQLException : " + e.getMessage());
    }

    reBtnClick = 1;
}else{
    JOptionPane.showMessageDialog(null, "비밀번호가 맞지 않습니다.", "ERROR_MESSAGE",
JOptionPane.ERROR_MESSAGE);
    return;
}

LoginFrame jf = new LoginFrame();
jf.setVisible(true);
this.dispose();
}

private void btnDupliActionPerformed(java.awt.event.ActionEvent evt) {
    if (txtID.getText().isEmpty()){
        JOptionPane.showMessageDialog(null, "아이디를 입력하세요.", "ERROR_MESSAGE",

```

```

JOptionPane.ERROR_MESSAGE);
        return;
    }
    strSQL = "Select ID From Users Where Users.ID = " + txtID.getText().trim() + " ";
    try{
        DBM.dbOpen();
        DBM.DB_rs = DBM.DB_stmt.executeQuery(strSQL);
        String id = "";
        while(DBM.DB_rs.next()){
            id = DBM.DB_rs.getString("ID");
        }
        if(id != ""){
            JOptionPane.showMessageDialog(null, "이미 사용중인 아이디 입니다.", "ERROR_MESSAGE",
JOptionPane.ERROR_MESSAGE);
            reBtnClick = 1;
            DBM.DB_rs.close();
            DBM.dbClose();
            return;
        }
        JOptionPane.showMessageDialog(null, "사용 가능한 아이디 입니다.", "알림",
JOptionPane.INFORMATION_MESSAGE);
        reBtnClick = 0;
        idText = txtID.getText();

        DBM.DB_rs.close();
        DBM.dbClose();
    }catch(Exception e){
        System.out.println("SQLException : " + e.getMessage());
    }
}

private void btnBackActionPerformed(java.awt.event.ActionEvent evt) {
    LoginFrame lf = new LoginFrame();
    lf.setVisible(true);
    lf.setLocation(300, 300);
    this.dispose();
}

```

ChoiceFrame (자동/수동 선택 프레임)

```

public class ChoiceFrame extends javax.swing.JFrame {
    public ChoiceFrame() {
        initComponents();
        jLabel3.setText(UserIDC.user_id);
    }
    private void btnAutoActionPerformed(java.awt.event.ActionEvent evt) {
        AutomaticFrame af = new AutomaticFrame();
        af.setVisible(true);
        af.setLocation(300, 300);
        this.dispose();
    }
}

```

```

}

private void btnManualActionPerformed(java.awt.event.ActionEvent evt) {
    ManualFrame mf = new ManualFrame();
    mf.setVisible(true);
    mf.setLocation(300, 300);
    this.dispose();
}

private void btnMyActionPerformed(java.awt.event.ActionEvent evt) {
    MyPageFrame mpf = new MyPageFrame();
    mpf.setVisible(true);
    mpf.setLocation(300, 300);
    this.dispose();
}

private void btnLogoutActionPerformed(java.awt.event.ActionEvent evt) {
    LoginFrame lf = new LoginFrame();
    lf.setVisible(true);
    lf.setLocation(300, 300);
    this.dispose();
}

```

AutomaticFrame.java (자동복권 프레임)

```

import java.util.ArrayList;
import java.util.Arrays;
import java.util.Collections;
import javax.swing.JOptionPane;

public class AutomaticFrame extends javax.swing.JFrame {
    DB_MAN DBM = new DB_MAN();
    String strSQL = "";

    public AutomaticFrame() {
        initComponents();
        lblSubTitle.setVisible(false);

        String user_money = "";
        strSQL = "Select Money From Users where Users.ID = " + UserIDC.user_id + "";
        try{
            DBM.dbOpen();
            DBM.DB_rs = DBM.DB_stmt.executeQuery(strSQL);
            while(DBM.DB_rs.next()){
                user_money = DBM.DB_rs.getString("Money");
            }

            lblChange.setText(user_money);
            DBM.DB_rs.close();
            DBM.dbClose();

```



```

        }catch(Exception e){
            System.out.println("SQLException : " + e.getMessage());
        }
    }

    private void btnMyActionPerformed(java.awt.event.ActionEvent evt) {
        MyPageFrame mpf = new MyPageFrame();
        mpf.setVisible(true);
        mpf.setLocation(300, 300);
        this.dispose();
    }

    private void btnLogoutActionPerformed(java.awt.event.ActionEvent evt) {
        LoginFrame lf = new LoginFrame();
        lf.setVisible(true);
        this.dispose();
    }

    private void btnBackActionPerformed(java.awt.event.ActionEvent evt) {
        ChoiceFrame sf = new ChoiceFrame();
        sf.setVisible(true);
        sf.setLocation(300, 300);
        this.dispose();
    }

    private void btnClickActionPerformed(java.awt.event.ActionEvent evt) {
        String user_money = "";
        strSQL = "Select Money From Users where Users.ID = " + UserIDC.user_id + " ";
        try{
            DBM.dbOpen();
            DBM.DB_rs = DBM.DB_stmt.executeQuery(strSQL);
            while(DBM.DB_rs.next()){
                user_money = DBM.DB_rs.getString("Money");
            }

            lblChange.setText(user_money);
            DBM.DB_rs.close();
            DBM.dbClose();
        }catch(Exception e){
            System.out.println("SQLException : " + e.getMessage());
        }

        if(Integer.parseInt(user_money) == 0){
            JOptionPane.showMessageDialog(null, "잔액이 없습니다.", "ERROR_MESSAGE",
JOptionPane.ERROR_MESSAGE);
            return;
        }

        strSQL = "Select Round from Lottery where Lottery.ID = " + UserIDC.user_id + " "
            + "order by Round";
        String strRound = "";
        int round = 0;
        try{

```

```

        DBM.dbOpen();
        DBM.DB_rs = DBM.DB_stmt.executeQuery(strSQL);
        while(DBM.DB_rs.next()){
            strRound = DBM.DB_rs.getString("Round");
        }

        if(strRound.equals("")){
            round = 1;
        }else{
            round = Integer.parseInt(strRound) + 1;
        }

        lblSubTitle.setText("<<" + round + "번 회차 즉석 복권>>");
        lblSubTitle.setVisible(true);

        DBM.DB_rs.close();
        DBM.dbClose();
    }catch(Exception e){
        System.out.println("SQLException : " + e.getMessage());
    }

    int[] showLottoNumber = new int[6];
    int[] showUserNumber = new int[6];

    ArrayList lottoNumber = new ArrayList<Integer>();
    ArrayList userNumber = new ArrayList<Integer>();
    for(int i = 1 ; i <= 45 ; i++) {
        lottoNumber.add(i);
        userNumber.add(i);
    }
    Collections.shuffle(lottoNumber);
    Collections.shuffle(userNumber);
    for(int i = 0 ; i < 6 ; i++) {
        showLottoNumber[i] = (int)lottoNumber.get(i);
        showUserNumber[i] = (int)userNumber.get(i);
    }

    Arrays.sort(showLottoNumber);
    Arrays.sort(showUserNumber);

    String strShowLottery = Arrays.toString(showLottoNumber).replaceAll("[^0-9 ]", "");
    String strShowUser = Arrays.toString(showUserNumber).replaceAll("[^0-9 ]", "");

    lblRealNum.setText(strShowLottery);
    lblRealMyNum.setText(strShowUser);

    int checkCount = 0;
    int prize = 0;
    String strCheck = "";

```

```

for(int i = 0; i < showLottoNumber.length; i++){
    for(int j = 0; j < showUserNumber.length; j++){
        if(showLottoNumber[i] == showUserNumber[j]){
            checkCount++;
        }
    }
}

switch(checkCount){
    case 0:
    case 1:
        strCheck = "낙첨 되었습니다.";
        break;
    case 2:
        prize = 1000;
        strCheck = "5등 당첨! " + checkCount + "개 일치합니다. " + prize + "원 적립";
        break;
    case 3:
        prize = 10000;
        strCheck = "4등 당첨! " + checkCount + "개 일치합니다. " + prize + "원 적립";
        break;
    case 4:
        prize = 100000;
        strCheck = "3등 당첨! " + checkCount + "개 일치합니다. " + prize + "원 적립";
        break;
    case 5:
        prize = 10000000;
        strCheck = "2등 당첨! " + checkCount + "개 일치합니다. " + prize + "원 적립";
        break;
    case 6:
        prize = 1000000000;
        strCheck = "1등 당첨! " + checkCount + "개 일치합니다. " + prize + "원 적립";
        break;
}

```

```

lblCheck.setText(strCheck);

```

```

int money = Integer.parseInt(user_money) + prize - 1000;

```

```

strSQL = "Update Users Set Users.Money = '" + money + "' Where Users.ID = '" + UserIDC.user_id
+ "'";

```

```

try{
    DBM.dbOpen();
    DBM.DB_stmt.executeUpdate(strSQL);
    DBM.dbClose();
}catch(Exception e){
    System.out.println("SQLException : " + e.getMessage());
}

```

```

strSQL = "Select Money From Users where Users.ID = '" + UserIDC.user_id + "'";
try{
    DBM.dbOpen();
    DBM.DB_rs = DBM.DB_stmt.executeQuery(strSQL);
}

```

```

        while(DBM.DB_rs.next()){
            user_money = DBM.DB_rs.getString("Money");
        }

        lblChange.setText(user_money);
        DBM.DB_rs.close();
        DBM.dbClose();
    }catch(Exception e){
        System.out.println("SQLException : " + e.getMessage());
    }

    strSQL = "Insert Into Lottery Values (";
    strSQL += "'" + round + "',";
    strSQL += "'" + strShowLottery + "',";
    strSQL += "'AUTO',";
    strSQL += "'" + checkCount + "',";
    strSQL += "'" + prize + "',";
    strSQL += "'" + UserIDC.user_id + "')";
    try{
        DBM.dbOpen();
        DBM.DB_stmt.executeUpdate(strSQL);
        DBM.dbClose();
    }catch(Exception e){
        System.out.println("SQLException : " + e.getMessage());
    }
}

```

ManualFrame (수동복권 프레임)

```

import java.util.ArrayList;
import java.util.Arrays;
import java.util.Collections;
import javax.swing.JOptionPane;

public class ManualFrame extends javax.swing.JFrame {
    DB_MAN DBM = new DB_MAN();
    String strSQL = "";

    public ManualFrame() {
        initComponents();

        String user_money = "";
        strSQL = "Select Money From Users where Users.ID = '" + UserIDC.user_id + "'";
        try{
            DBM.dbOpen();
            DBM.DB_rs = DBM.DB_stmt.executeQuery(strSQL);
            while(DBM.DB_rs.next()){
                user_money = DBM.DB_rs.getString("Money");
            }
        }
    }
}

```

```

        lblChange.setText(user_money);
        DBM.DB_rs.close();
        DBM.dbClose();
    }catch(Exception e){
        System.out.println("SQLException : " + e.getMessage());
    }
}

private void btnMyActionPerformed(java.awt.event.ActionEvent evt) {
    MyPageFrame mpf = new MyPageFrame();
    mpf.setVisible(true);
    mpf.setLocation(300, 300);
    this.dispose();
}

private void btnLogoutActionPerformed(java.awt.event.ActionEvent evt) {
    LoginFrame lf = new LoginFrame();
    lf.setVisible(true);
    this.dispose();
}

private void btnBackActionPerformed(java.awt.event.ActionEvent evt) {
    ChoiceFrame sf = new ChoiceFrame();
    sf.setVisible(true);
    sf.setLocation(300, 300);
    this.dispose();
}

private void btnClickActionPerformed(java.awt.event.ActionEvent evt) {
    String user_money = "";
    strSQL = "Select Money From Users where Users.ID = '" + UserIDC.user_id + "'";
    try{
        DBM.dbOpen();
        DBM.DB_rs = DBM.DB_stmt.executeQuery(strSQL);
        while(DBM.DB_rs.next()){
            user_money = DBM.DB_rs.getString("Money");
        }

        lblChange.setText(user_money);
        DBM.DB_rs.close();
        DBM.dbClose();
    }catch(Exception e){
        System.out.println("SQLException : " + e.getMessage());
    }

    if(Integer.parseInt(user_money) == 0){
        JOptionPane.showMessageDialog(null, "잔액이 없습니다.", "ERROR_MESSAGE",
JOptionPane.ERROR_MESSAGE);
        return;
    }

    strSQL = "Select Round from Lottery where Lottery.ID = '" + UserIDC.user_id + "'" + "order by

```

Round";

```
String strRound = "";
int round = 0;
try{
    DBM.dbOpen();
    DBM.DB_rs = DBM.DB_stmt.executeQuery(strSQL);
    while(DBM.DB_rs.next()){
        strRound = DBM.DB_rs.getString("Round");
    }

    if(strRound.equals("")){
        round = 1;
    }else{
        round = Integer.parseInt(strRound) + 1;
    }

    lblSubTitle.setText("<<" + round + "번 회차 즉석 복권>>");
    lblSubTitle.setVisible(true);

    DBM.DB_rs.close();
    DBM.dbClose();
}catch(Exception e){
    System.out.println("SQLException : " + e.getMessage());
}
```

```
int[] showLottoNumber = new int[6];
ArrayList<Integer> showUserNumbers = new ArrayList();
int numCount = 0;

if(cbx1.isSelected()){
    showUserNumbers.add(Integer.parseInt(cbx1.getText()));
    numCount++;
}
if(cbx2.isSelected()){
    showUserNumbers.add(Integer.parseInt(cbx2.getText()));
    numCount++;
}
if(cbx3.isSelected()){
    showUserNumbers.add(Integer.parseInt(cbx3.getText()));
    numCount++;
}
if(cbx4.isSelected()){
    showUserNumbers.add(Integer.parseInt(cbx4.getText()));
    numCount++;
}
if(cbx5.isSelected()){
    showUserNumbers.add(Integer.parseInt(cbx5.getText()));
    numCount++;
}
if(cbx6.isSelected()){
    showUserNumbers.add(Integer.parseInt(cbx6.getText()));
}
```

```
        numCount++;
    }
    if(cbx7.isSelected()){
        showUserNumbers.add(Integer.parseInt(cbx7.getText()));
        numCount++;
    }
    if(cbx8.isSelected()){
        showUserNumbers.add(Integer.parseInt(cbx8.getText()));
        numCount++;
    }
    if(cbx9.isSelected()){
        showUserNumbers.add(Integer.parseInt(cbx9.getText()));
        numCount++;
    }
    if(cbx10.isSelected()){
        showUserNumbers.add(Integer.parseInt(cbx10.getText()));
        numCount++;
    }
    if(cbx11.isSelected()){
        showUserNumbers.add(Integer.parseInt(cbx11.getText()));
        numCount++;
    }
    if(cbx12.isSelected()){
        showUserNumbers.add(Integer.parseInt(cbx12.getText()));
        numCount++;
    }
    if(cbx13.isSelected()){
        showUserNumbers.add(Integer.parseInt(cbx13.getText()));
        numCount++;
    }
    if(cbx14.isSelected()){
        showUserNumbers.add(Integer.parseInt(cbx14.getText()));
        numCount++;
    }
    if(cbx15.isSelected()){
        showUserNumbers.add(Integer.parseInt(cbx15.getText()));
        numCount++;
    }
    if(cbx16.isSelected()){
        showUserNumbers.add(Integer.parseInt(cbx16.getText()));
        numCount++;
    }
    if(cbx17.isSelected()){
        showUserNumbers.add(Integer.parseInt(cbx17.getText()));
        numCount++;
    }
    if(cbx18.isSelected()){
        showUserNumbers.add(Integer.parseInt(cbx18.getText()));
        numCount++;
    }
    if(cbx19.isSelected()){
        showUserNumbers.add(Integer.parseInt(cbx19.getText()));
```

```
        numCount++;
    }
    if(cbx20.isSelected()){
        showUserNumbers.add(Integer.parseInt(cbx20.getText()));
        numCount++;
    }
    if(cbx21.isSelected()){
        showUserNumbers.add(Integer.parseInt(cbx21.getText()));
        numCount++;
    }
    if(cbx22.isSelected()){
        showUserNumbers.add(Integer.parseInt(cbx22.getText()));
        numCount++;
    }
    if(cbx23.isSelected()){
        showUserNumbers.add(Integer.parseInt(cbx23.getText()));
        numCount++;
    }
    if(cbx24.isSelected()){
        showUserNumbers.add(Integer.parseInt(cbx24.getText()));
        numCount++;
    }
    if(cbx25.isSelected()){
        showUserNumbers.add(Integer.parseInt(cbx25.getText()));
        numCount++;
    }
    if(cbx26.isSelected()){
        showUserNumbers.add(Integer.parseInt(cbx26.getText()));
        numCount++;
    }
    if(cbx27.isSelected()){
        showUserNumbers.add(Integer.parseInt(cbx27.getText()));
        numCount++;
    }
    if(cbx28.isSelected()){
        showUserNumbers.add(Integer.parseInt(cbx28.getText()));
        numCount++;
    }
    if(cbx29.isSelected()){
        showUserNumbers.add(Integer.parseInt(cbx29.getText()));
        numCount++;
    }
    if(cbx30.isSelected()){
        showUserNumbers.add(Integer.parseInt(cbx30.getText()));
        numCount++;
    }
    if(cbx31.isSelected()){
        showUserNumbers.add(Integer.parseInt(cbx31.getText()));
        numCount++;
    }
    if(cbx32.isSelected()){
        showUserNumbers.add(Integer.parseInt(cbx32.getText()));
```



```
        numCount++;
    }
    if(cb33.isSelected()){
        showUserNumbers.add(Integer.parseInt(cb33.getText()));
        numCount++;
    }
    if(cb34.isSelected()){
        showUserNumbers.add(Integer.parseInt(cb34.getText()));
        numCount++;
    }
    if(cb35.isSelected()){
        showUserNumbers.add(Integer.parseInt(cb35.getText()));
        numCount++;
    }
    if(cb36.isSelected()){
        showUserNumbers.add(Integer.parseInt(cb36.getText()));
        numCount++;
    }
    if(cb37.isSelected()){
        showUserNumbers.add(Integer.parseInt(cb37.getText()));
        numCount++;
    }
    if(cb38.isSelected()){
        showUserNumbers.add(Integer.parseInt(cb38.getText()));
        numCount++;
    }
    if(cb39.isSelected()){
        showUserNumbers.add(Integer.parseInt(cb39.getText()));
        numCount++;
    }
    if(cb40.isSelected()){
        showUserNumbers.add(Integer.parseInt(cb40.getText()));
        numCount++;
    }
    if(cb41.isSelected()){
        showUserNumbers.add(Integer.parseInt(cb41.getText()));
        numCount++;
    }
    if(cb42.isSelected()){
        showUserNumbers.add(Integer.parseInt(cb42.getText()));
        numCount++;
    }
    if(cb43.isSelected()){
        showUserNumbers.add(Integer.parseInt(cb43.getText()));
        numCount++;
    }
    if(cb44.isSelected()){
        showUserNumbers.add(Integer.parseInt(cb44.getText()));
        numCount++;
    }
    if(cb45.isSelected()){
        showUserNumbers.add(Integer.parseInt(cb45.getText()));
```

```

        numCount++;
    }

    if(numCount < 6){
        JOptionPane.showMessageDialog(null, "번호를 6개 선택 해주세요.", "ERROR_MESSAGE",
JOptionPane.ERROR_MESSAGE);
        return;
    }
    if(showUserNumbers.size() > 6){
        JOptionPane.showMessageDialog(null, "번호를 6개만 선택 해주세요.", "ERROR_MESSAGE",
JOptionPane.ERROR_MESSAGE);
        return;
    }

    int[] showUserNumber = showUserNumbers.stream().mapToInt(Integer::intValue).toArray();

    ArrayList lottoNumber = new ArrayList<Integer>();
    for(int i = 1 ; i <= 45 ; i++) {
        lottoNumber.add(i);
    }
    Collections.shuffle(lottoNumber);

    for(int i = 0 ; i < 6 ; i++) {
        showLottoNumber[i] = (int)lottoNumber.get(i);
    }

    Arrays.sort(showLottoNumber);
    Arrays.sort(showUserNumber);

    String strShowLottery = Arrays.toString(showLottoNumber).replaceAll("[^0-9 ]", "");
    String strShowUser = Arrays.toString(showUserNumber).replaceAll("[^0-9 ]", "");

    lblRealNum.setText(strShowLottery);
    lblRealMyNum.setText(strShowUser);

    int checkCount = 0;
    int prize = 0;
    String strCheck = "";

    for(int i = 0; i < showLottoNumber.length; i++){
        for(int j = 0; j < showUserNumber.length; j++){
            if(showLottoNumber[i] == showUserNumber[j]){
                checkCount++;
            }
        }
    }

    switch(checkCount){
        case 0:
        case 1:
            strCheck = "낙첨 되었습니다.";

```

```

        break;
    case 2:
        prize = 1000;
        strCheck = "5등 당첨! " + checkCount + "개 일치합니다. " + prize + "원 적립";
        break;
    case 3:
        prize = 10000;
        strCheck = "4등 당첨! " + checkCount + "개 일치합니다. " + prize + "원 적립";
        break;
    case 4:
        prize = 100000;
        strCheck = "3등 당첨! " + checkCount + "개 일치합니다. " + prize + "원 적립";
        break;
    case 5:
        prize = 1000000;
        strCheck = "2등 당첨! " + checkCount + "개 일치합니다. " + prize + "원 적립";
        break;
    case 6:
        prize = 10000000;
        strCheck = "1등 당첨! " + checkCount + "개 일치합니다. " + prize + "원 적립";
        break;
    }
    lblCheck.setText(strCheck);

```

```

int money = Integer.parseInt(user_money) + prize - 1000;

```

```

strSQL = "Update Users Set Users.Money = '" + money + "' Where Users.ID = '" + UserIDC.user_id
+ "'";

```

```

try{
    DBM.dbOpen();
    DBM.DB_stmt.executeUpdate(strSQL);
    DBM.dbClose();
}catch(Exception e){
    System.out.println("SQLException : " + e.getMessage());
}

```

```

strSQL = "Select Money From Users where Users.ID = '" + UserIDC.user_id + "'";

```

```

try{
    DBM.dbOpen();
    DBM.DB_rs = DBM.DB_stmt.executeQuery(strSQL);
    while(DBM.DB_rs.next()){
        user_money = DBM.DB_rs.getString("Money");
    }

    lblChange.setText(user_money);
    DBM.DB_rs.close();
    DBM.dbClose();
}catch(Exception e){
    System.out.println("SQLException : " + e.getMessage());
}

```

```

strSQL = "Insert Into Lottery Values (";
strSQL += "" + round + ",";
strSQL += "" + strShowLottery + ",";
strSQL += "Manual",";
strSQL += "" + checkCount + ",";
strSQL += "" + prize + ",";
strSQL += "" + UserIDC.user_id + ")";
try{
    DBM.dbOpen();
    DBM.DB_stmt.executeUpdate(strSQL);
    DBM.dbClose();
}catch(Exception e){
    System.out.println("SQLException : " + e.getMessage());
}

```

```

cbx1.setSelected(false);
cbx2.setSelected(false);
cbx3.setSelected(false);
cbx4.setSelected(false);
cbx5.setSelected(false);
cbx6.setSelected(false);
cbx7.setSelected(false);
cbx8.setSelected(false);
cbx9.setSelected(false);
cbx10.setSelected(false);
cbx11.setSelected(false);
cbx12.setSelected(false);
cbx13.setSelected(false);
cbx14.setSelected(false);
cbx15.setSelected(false);
cbx16.setSelected(false);
cbx17.setSelected(false);
cbx18.setSelected(false);
cbx19.setSelected(false);
cbx20.setSelected(false);
cbx21.setSelected(false);
cbx22.setSelected(false);
cbx23.setSelected(false);
cbx24.setSelected(false);
cbx25.setSelected(false);
cbx26.setSelected(false);
cbx27.setSelected(false);
cbx28.setSelected(false);
cbx29.setSelected(false);
cbx30.setSelected(false);
cbx31.setSelected(false);
cbx32.setSelected(false);
cbx33.setSelected(false);
cbx34.setSelected(false);
cbx35.setSelected(false);
cbx36.setSelected(false);

```

```

        cbx37.setSelected(false);
        cbx38.setSelected(false);
        cbx39.setSelected(false);
        cbx40.setSelected(false);
        cbx41.setSelected(false);
        cbx42.setSelected(false);
        cbx43.setSelected(false);
        cbx44.setSelected(false);
        cbx45.setSelected(false);
    }
}

```

MyPageFrame (마이페이지 프레임)

```

import javax.swing.JOptionPane;
import javax.swing.text.AttributeSet;
import javax.swing.text.BadLocationException;
import javax.swing.text.PlainDocument;

public class MyPageFrame extends javax.swing.JFrame {
    DBMAN DBM = new DBMAN();
    String strSQL = "";
    String user_money = "";
    public MyPageFrame() {
        initComponents();
        lblIDName.setText(UserIDC.user_id);

        strSQL = "Select Money From Users where Users.ID = '" + UserIDC.user_id + "'";
        try{
            DBM.dbOpen();
            DBM.DB_rs = DBM.DB_stmt.executeQuery(strSQL);
            while(DBM.DB_rs.next()){
                user_money = DBM.DB_rs.getString("Money");
            }

            lblPointPrice.setText(user_money);
            DBM.DB_rs.close();
            DBM.dbClose();
        }catch(Exception e){
            System.out.println("SQLException : " + e.getMessage());
        }

        lottoTable();
    }

    protected void lottoTable(){
        int idxRow = 0;
        strSQL = "Select TOP 15 * "
            + "From Lottery where Lottery.ID = '" + UserIDC.user_id + "' "
            + "order by Round desc";
        try{
            DBM.dbOpen();

```

```

        DBM.DB_rs = DBM.DB_stmt.executeQuery(strSQL);
        while(DBM.DB_rs.next()){
            jTable1.setValueAt(DBM.DB_rs.getString("Round"), idxRow, 0);
            jTable1.setValueAt(DBM.DB_rs.getString("Numbers"), idxRow, 1);
            jTable1.setValueAt(DBM.DB_rs.getString("LType"), idxRow, 2);
            jTable1.setValueAt(DBM.DB_rs.getString("Matched"), idxRow, 3);
            jTable1.setValueAt(DBM.DB_rs.getString("Prize"), idxRow, 4);
            idxRow++;
        }

        DBM.DB_rs.close();
        DBM.dbClose();
    }catch(Exception e){
        System.out.println("SQLException : " + e.getMessage());
    }
}

private void btnBackActionPerformed(java.awt.event.ActionEvent evt) {
    ChoiceFrame sf = new ChoiceFrame();
    sf.setVisible(true);
    sf.setLocation(300, 300);
    this.dispose();
}

private void btnLogoutActionPerformed(java.awt.event.ActionEvent evt) {
    LoginFrame lf = new LoginFrame();
    lf.setVisible(true);
    this.dispose();
}

private void btnOutActionPerformed(java.awt.event.ActionEvent evt) {
    int checkResult = JOptionPane.showConfirmDialog(null, "회원 탈퇴 하시겠습니까?", "알림",
    JOptionPane.ERROR_MESSAGE);

    if(checkResult == 0){
        strSQL = "Delete From Users where Users.ID = '" + UserIDC.user_id + "' ";

        try{
            DBM.dbOpen();
            DBM.DB_stmt.executeUpdate(strSQL);
            DBM.dbClose();
        }catch(Exception e){
            System.out.println("SQLException : " + e.getMessage());
        }

        strSQL = "Delete From Lottery where Lottery.ID = '" + UserIDC.user_id + "' ";

        try{
            DBM.dbOpen();
            DBM.DB_stmt.executeUpdate(strSQL);
            DBM.dbClose();
        }catch(Exception e){

```

```

        System.out.println("SQLException : " + e.getMessage());
    }

    LoginFrame lf = new LoginFrame();
    lf.setVisible(true);
    this.dispose();
}

}

private void btnChargingActionPerformed(java.awt.event.ActionEvent evt) {
    int money = 0;
    strSQL = "Select Money From Users Where Users.ID = '" + UserIDC.user_id + "'";

    try{
        DBM.dbOpen();
        DBM.DB_rs = DBM.DB_stmt.executeQuery(strSQL);
        while(DBM.DB_rs.next()){
            user_money = DBM.DB_rs.getString("Money");
        }

        money = Integer.parseInt(user_money);
        DBM.dbClose();

        if(money != 0){
            JOptionPane.showMessageDialog(null, "잔액이 없으면 충전됩니다.", "ERROR_MESSAGE",
JOptionPane.ERROR_MESSAGE);
            return;
        }

    }catch(Exception e){
        System.out.println("SQLException : " + e.getMessage());
    }

    strSQL = "Update Users Set Users.Money = '" + 100000 + "' Where Users.ID = '" +
UserIDC.user_id + "'";
    try{
        DBM.dbOpen();
        DBM.DB_stmt.executeUpdate(strSQL);
        DBM.dbClose();
    }catch(Exception e){
        System.out.println("SQLException : " + e.getMessage());
    }

    strSQL = "Select Money From Users where Users.ID = '" + UserIDC.user_id + "'";
    try{
        DBM.dbOpen();
        DBM.DB_rs = DBM.DB_stmt.executeQuery(strSQL);
        while(DBM.DB_rs.next()){
            user_money = DBM.DB_rs.getString("Money");

```

```
    }

    lblPointPrice.setText(user_money);
    DBM.DB_rs.close();
    DBM.dbClose();
} catch (Exception e) {
    System.out.println("SQLException : " + e.getMessage());
}
JOptionPane.showMessageDialog(null, "잔액이 충전되었습니다!", "알림",
JOptionPane.INFORMATION_MESSAGE);
}
```