Math Game Android Application

08/01/13

Joseph Casal

Jorge Varela

Fernando Acosta

COP 2805 ADV JAVA PROGRAMMING

Prof: Nguyen, Hien

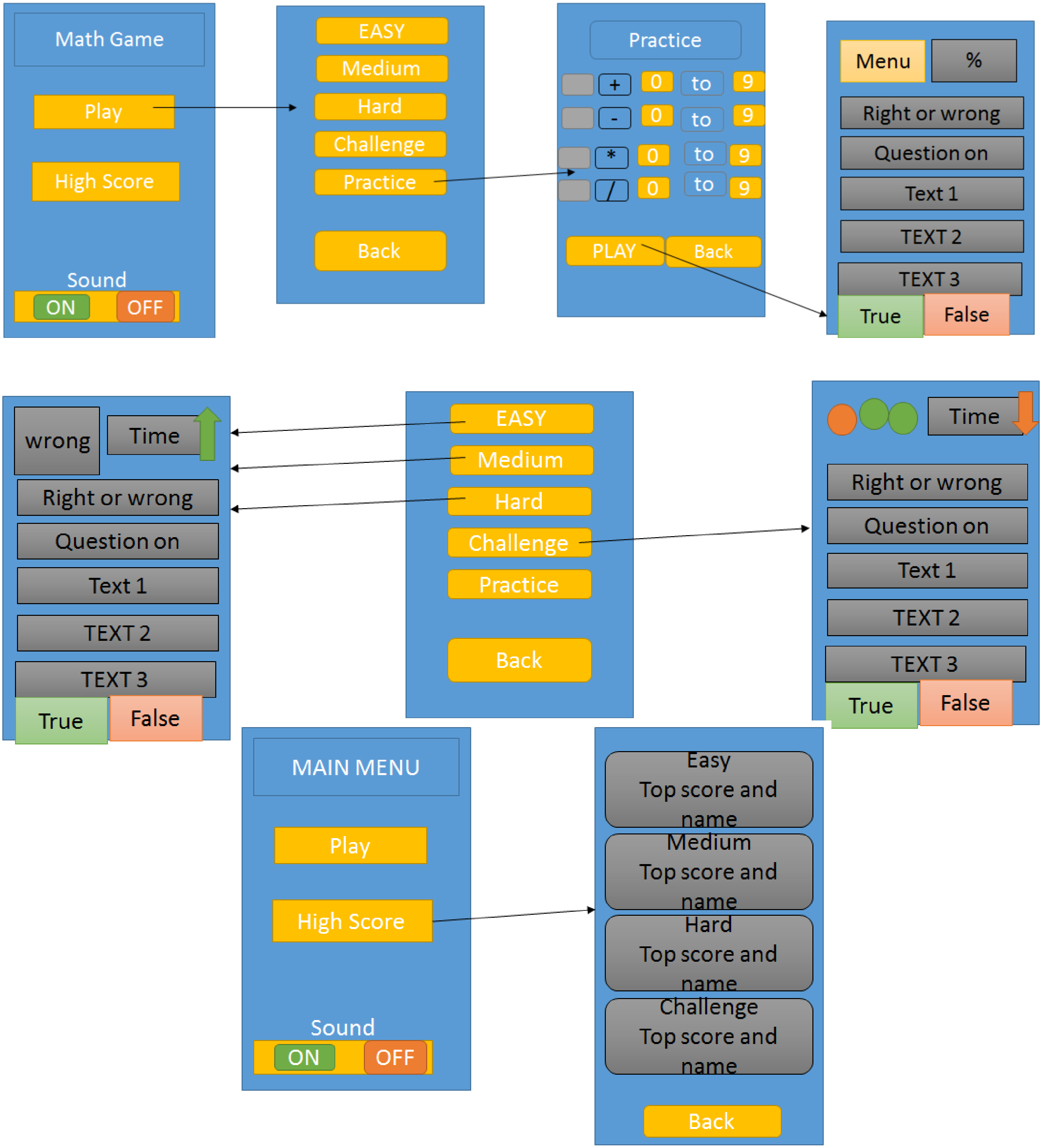
Summer 2013

Project Description

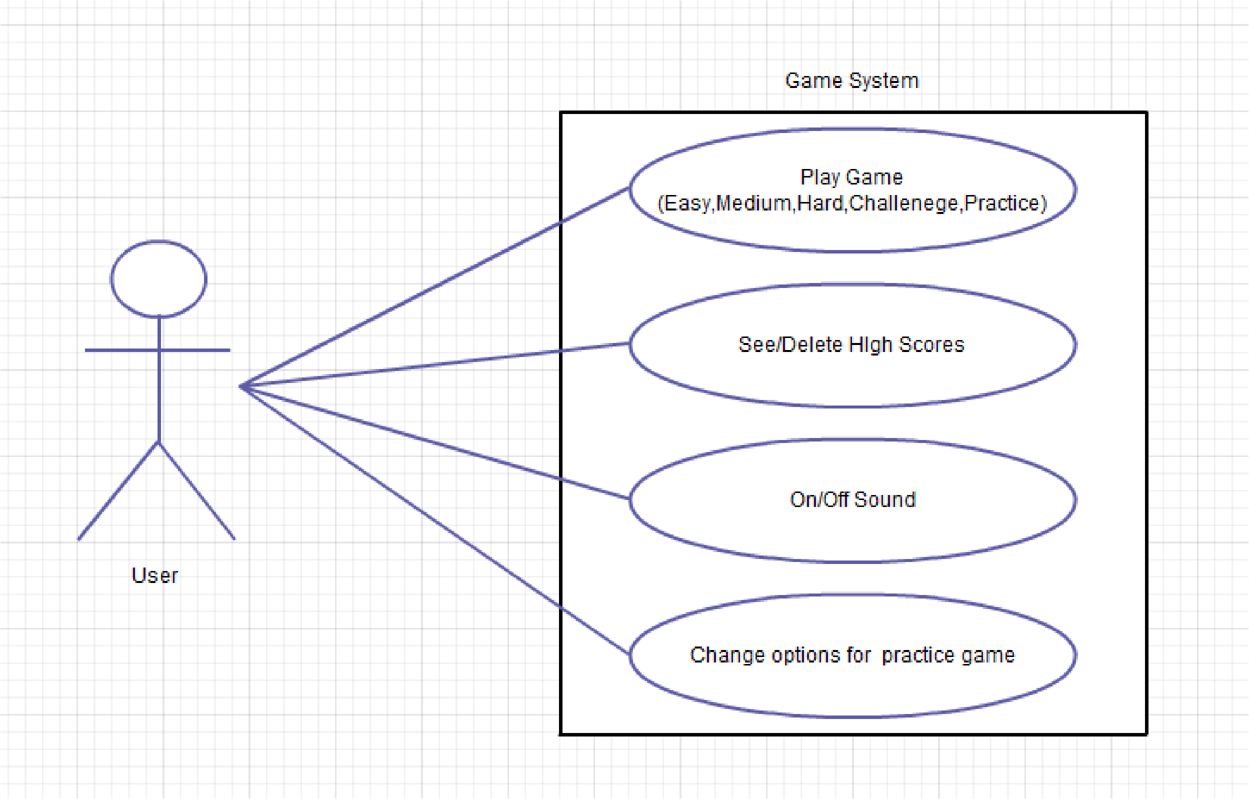
The project is an android entertainment application where the user can test and improve their speed on their basic math skills. During the game activities, the user solves multiple basic and challenging equations. The application will feature four game modes (easy, medium, hard, and challenge). There is also a practice section where the user can change the options for their practice game and go back to the default options. These options are saved for improved user interaction. There is also a high scores menu that will show the best score for each game mode and allow the user to reset the scores.

Program Flow

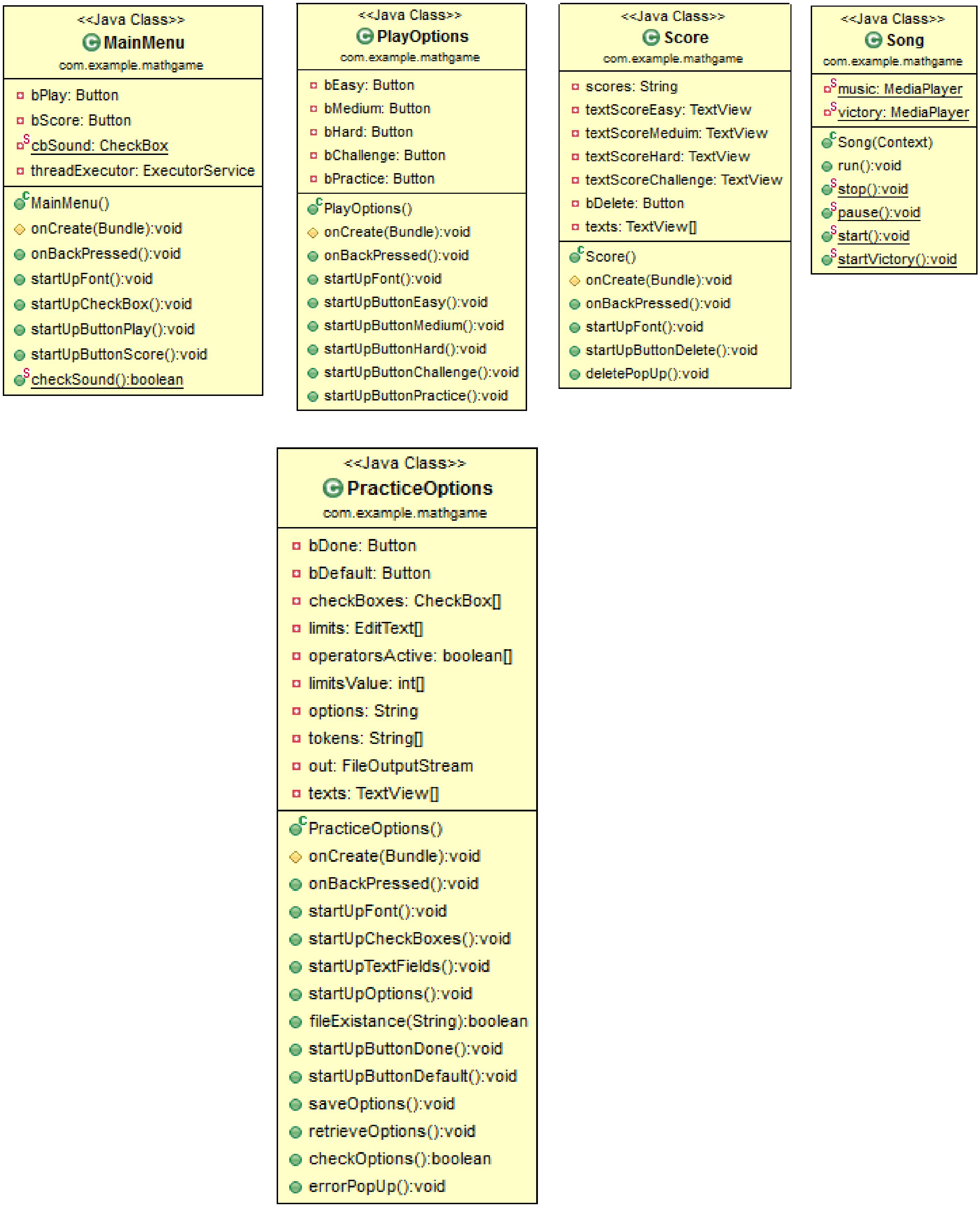
- **Start Screen/Menu**
  - PLAY
    - Easy
      - Addition, Subtraction, from 0 to 20 (20 questions)
    - Medium
      - Addition, Subtraction 0 – 30, Multiplication 0 - 12 (20 questions)
    - Hard
      - Addition Subtraction 0 – 50, Multiplication Division 0 – 12 (20 questions)
    - Practice
      - Can edit operators and length of numbers
    - Challenge
      - How far you can get with only 3 strikes (have to solve each question in 5 sec)
  - High Score Board
    - Score for easy medium hard
      - Your score : time you took to solve 20 questions + 5 penalty for wrong answers
      - Show best score for each game mode
    - Score for challenge
      - It will be by how many questions you did
    - User can delete the scores
  - Sound OFF/ON

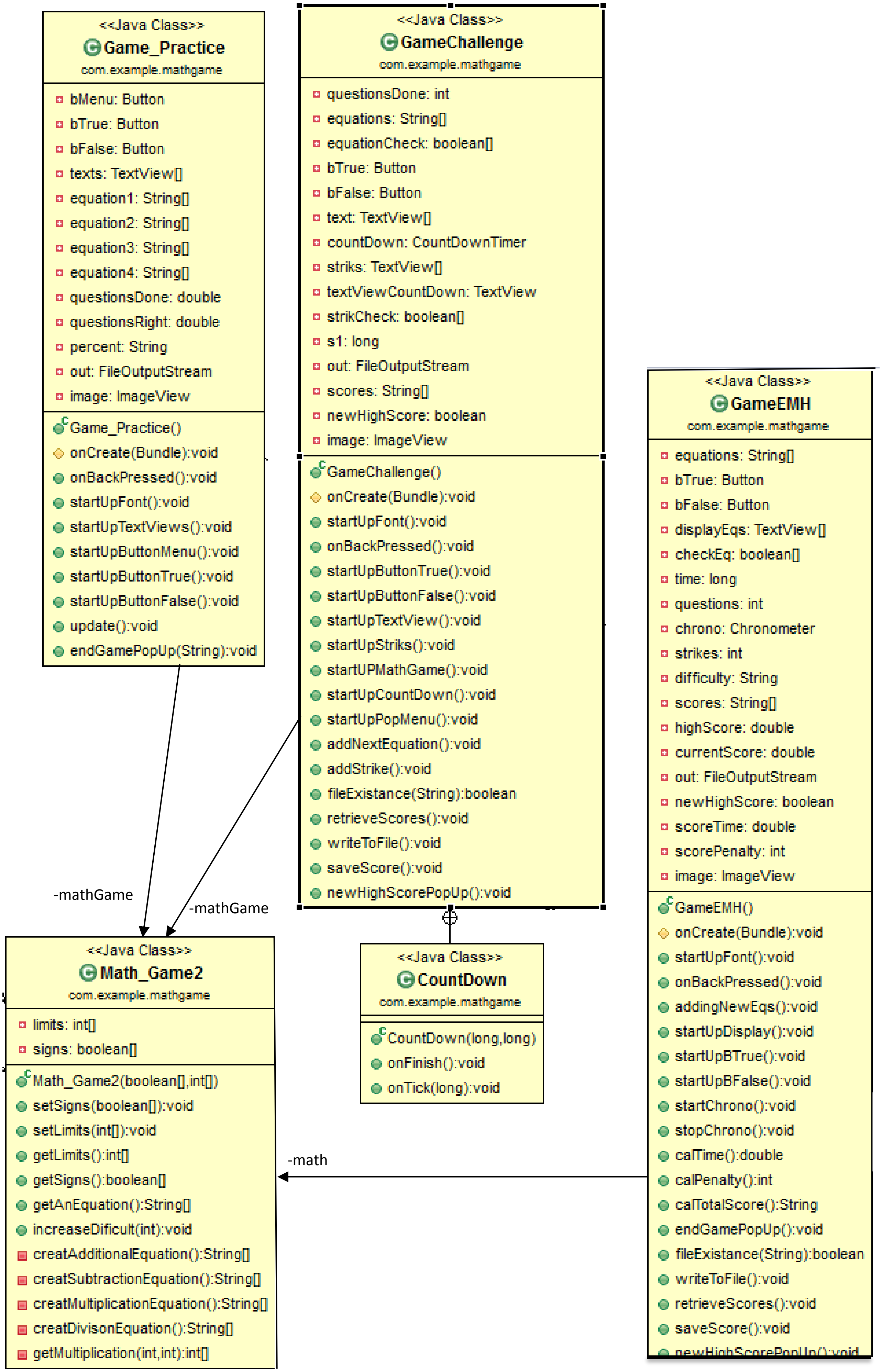

Math Game
Play
High Score
Sound
ON
OFF
EASY
Medium
Hard
Challenge
Practice
Back
Practice
+
0
to
9
-
0
to
9
*
0
to
9
/
0
to
9
PLAY
Back
Menu
%
Right or wrong
Question on
Text 1
TEXT 2
TEXT 3
True
False
wrong
Time
Right or wrong
Question on
Text 1
TEXT 2
TEXT 3
True
False
EASY
Medium
Hard
Challenge
Practice
Back
Time
Right or wrong
Question on
Text 1
TEXT 2
TEXT 3
True
False
MAIN MENU
Play
High Score
Sound
ON
OFF
Easy
Top score and name
Medium
Top score and name
Hard
Top score and name
Challenge
Top score and name
Back

UML Use Case Diagram


Game System
Play Game
(Easy,Medium,Hard,Challenege,Practice)
See/Delete HIgh Scores
On/Off Sound
Change options for practice game
User

UML Class Diagrams




<<Java Class>>
Game_Practice
com.example.mathgame
bMenu: Button
bTrue: Button
bFalse: Button
texts: TextView[]
equation1: String[]
equation2: String[]
equation3: String[]
equation4: String[]
questionsDone: double
questionsRight: double
percent: String
out: FileOutputStream
image: ImageView
Game_Practice()
onCreate(Bundle):void
onBackPressed():void
startUpFont():void
startUpTextViews():void
startUpButtonMenu():void
startUpButtonTrue():void
startUpButtonFalse():void
update():void
endGamePopUp(String):void
<<Java Class>>
GameChallenge
com.example.mathgame
questionsDone: int
equations: String[]
equationCheck: boolean[]
bTrue: Button
bFalse: Button
text: TextView[]
countDown: CountDownTimer
striks: TextView[]
textViewCountDown: TextView
strikCheck: boolean[]
s1: long
out: FileOutputStream
scores: String[]
newHighScore: boolean
image: ImageView
GameChallenge()
onCreate(Bundle):void
startUpFont():void
onBackPressed():void
startUpButtonTrue():void
startUpButtonFalse():void
startUpTextView():void
startUpStriks():void
startUPMathGame():void
startUpCountDown():void
startUpPopMenu():void
addNextEquation():void
addStrike():void
fileExistance(String):boolean
retrieveScores():void
writeToFile():void
saveScore():void
newHighScorePopUp():void
<<Java Class>>
GameEMH
com.example.mathgame
equations: String[]
bTrue: Button
bFalse: Button
displayEqs: TextView[]
checkEq: boolean[]
time: long
questions: int
chrono: Chronometer
strikes: int
difficulty: String
scores: String[]
highScore: double
currentScore: double
out: FileOutputStream
newHighScore: boolean
scoreTime: double
scorePenalty: int
image: ImageView
GameEMH()
onCreate(Bundle):void
startUpFont():void
onBackPressed():void
addingNewEqs():void
startUpDisplay():void
startUpBTrue():void
startUpBFalse():void
startChrono():void
stopChrono():void
calTime():double
calPenalty():int
calTotalScore():String
endGamePopUp():void
fileExistance(String):boolean
writeToFile():void
retrieveScores():void
saveScore():void
newHighScorePopUp():void
-mathGame
-mathGame
<<Java Class>>
Math_Game2
com.example.mathgame
limits: int[]
signs: boolean[]
Math_Game2(boolean[],int[])
setSigns(boolean[]):void
setLimits(int[]):void
getLimits():int[]
getSigns():boolean[]
getAnEquation():String[]
increaseDificult(int):void
creatAdditionalEquation():String[]
creatSubtractionEquation():String[]
creatMultiplicationEquation():String[]
creatDivisonEquation():String[]
getMultiplication(int,int):int[]
<<Java Class>>
CountDown
com.example.mathgame
CountDown(long,long)
onFinish():void
onTick(long):void
-math

MainMenu.java

```
package com.example.mathgame;

import java.util.concurrent.ExecutorService;
import java.util.concurrent.Executors;
import android.app.Activity;
import android.content.Intent;
import android.graphics.Color;
import android.graphics.Typeface;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.CheckBox;

public class MainMenu extends Activity {

      private Button bPlay,bScore;
      private CheckBox cbSound;
      private ExecutorService threadExecutor;

      @Override
      protected void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.activity_main_menu);
            startUpButtonPlay();
            startUpButtonScore();
            startUpCheckBox();
            startUpFont();
            threadExecutor = Executors.newCachedThreadPool();
            threadExecutor.execute(new Song(MainMenu.this));
            threadExecutor.shutdown();
      }

      @Override
      public void onBackPressed() {
            super.onBackPressed();
            Song.pause();
      }

      // adds a new font from the assets folder/ font
      public void startUpFont(){
            Typeface crayon_crumble = Typeface.createFromAsset(getAssets(),
"fonts/dk_crayon_crumble.ttf");
            bPlay.setTypeface(crayon_crumble);     // set the font for all the buttons to
crayon_crumble
            bScore.setTypeface(crayon_crumble);
      }

      public void startUpCheckBox(){
            cbSound = (CheckBox) findViewById(R.id.checkBoxSound);
            cbSound.setButtonDrawable(getResources().getDrawable(R.drawable.sound_on));
            cbSound.setOnClickListener(new View.OnClickListener(){

                  @Override
                  public void onClick(View v) {

                        if(cbSound.isChecked()){

      cbSound.setButtonDrawable(getResources().getDrawable(R.drawable.sound_off));
                              Song.pause();
                        }
                        else{
```

```java
    cbSound.setButtonDrawable(getResources().getDrawable(R.drawable.sound_on));
                    Song.start();
                }
            }
        });
    }

    public void startUpButtonPlay(){
        bPlay = (Button) findViewById(R.id.buttonPlay);
        bPlay.setOnClickListener(new View.OnClickListener() {

            @Override
            public void onClick(View v) {
                Intent startPlayOptions = new
Intent("com.example.mathgame.PLAYOPTIONS");
                startActivity(startPlayOptions);
            }
        });
        bPlay.setTextColor(Color.WHITE);

    }
    public void startUpButtonScore(){
        bScore = (Button) findViewById(R.id.buttonScore);
        bScore.setOnClickListener(new View.OnClickListener() {

            @Override
            public void onClick(View v) {
                Intent startScore = new Intent("android.intent.action.SCORE");
                startActivity(startScore);
            }
        });
        bScore.setTextColor(Color.WHITE);

    }
    public static boolean checkSound(){
        if(cbSound.isChecked())
            return true;
        else
            return false;
    }
}
```

PlayOptions.java

```
package com.example.mathgame;

import android.app.Activity;
import android.content.Intent;
import android.graphics.Color;
import android.graphics.Typeface;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;

public class PlayOptions extends Activity {

      private Button bEasy, bMedium, bHard, bChallenge, bPractice;

      @Override
      protected void onCreate(Bundle playOptionsMenu) {
            super.onCreate(playOptionsMenu);
            setContentView(R.layout.play_options);
            startUpButtonEasy();
            startUpButtonMedium();
            startUpButtonHard();
            startUpButtonChallenge();
            startUpButtonPractice();
            startUpFont();
      }

      @Override
      public void onBackPressed() {
            super.onBackPressed();
            finish();
      }

      public void startUpFont(){
            Typeface crayon_crumble = Typeface.createFromAsset(getAssets(),
"fonts/dk_crayon_crumble.ttf");
            bEasy.setTypeface(crayon_crumble);
            bMedium.setTypeface(crayon_crumble);
            bHard.setTypeface(crayon_crumble);
            bChallenge.setTypeface(crayon_crumble);
            bPractice.setTypeface(crayon_crumble);
      }

      // methods StartUps
      public void startUpButtonEasy() {
            bEasy = (Button) findViewById(R.id.buttonEasy);
            bEasy.setOnClickListener(new View.OnClickListener() {

                  @Override
                  public void onClick(View v) {
                        Intent openGameE = new Intent("com.example.mathgame.GAMEEMH");
                        boolean [] check = {true,true,false,false};// addition and
subtraction only

                        int [] limits = { 0, 20, 0, 20, 0, 0, 0, 0 };// from 0 to 20
                        String difficulty = "easy";
                        openGameE.putExtra("check", check);
                        openGameE.putExtra("limits", limits);
                        openGameE.putExtra("difficulty", difficulty);
                        startActivity(openGameE);
                        finish();
                  }
            });// end of click listener method
            bEasy.setTextColor(Color.WHITE);
      }// end start up method for easy button

      public void startUpButtonMedium() {
            bMedium = (Button) findViewById(R.id.buttonMedium);
            bMedium.setOnClickListener(new View.OnClickListener() {

                  @Override
                  public void onClick(View v) {
                        Intent openGameE = new Intent("com.example.mathgame.GAMEEMH");
                        boolean [] check = {true,true,true,false};// addition,
subtraction and multiplication
                        int [] limits = { 0, 30, 0, 30, 0, 12, 0, 0 };// add and sub from
0 to 30, mul from 0 to 12
                        String difficulty = "meduim";
                        openGameE.putExtra("check", check);
                        openGameE.putExtra("limits", limits);
                        openGameE.putExtra("difficulty", difficulty);
                        startActivity(openGameE);
                        finish();
                  }
            });// end of click listener method
            bMedium.setTextColor(Color.WHITE);
      }// end start up method for Medium button

      public void startUpButtonHard() {
            bHard = (Button) findViewById(R.id.buttonHard);
            bHard.setOnClickListener(new View.OnClickListener() {

                  @Override
                  public void onClick(View v) {
                        Intent openGameE = new Intent("com.example.mathgame.GAMEEMH");
                        boolean [] check = {true,true,true,true};
                        int [] limits = { 0, 50, 0, 50, 0, 12, 0, 12};
                        String difficulty = "hard";
                        openGameE.putExtra("check", check);
                        openGameE.putExtra("limits", limits);
                        openGameE.putExtra("difficulty", difficulty);
                        startActivity(openGameE);
                        finish();
                  }
            });// end of click listener method
            bHard.setTextColor(Color.WHITE);
      }// end start up method for Hard button

      public void startUpButtonChallenge() {
            bChallenge = (Button) findViewById(R.id.buttonChallenge);
            bChallenge.setOnClickListener(new View.OnClickListener() {

                  @Override
                  public void onClick(View v) {
                        Intent startGameChallenge = new
Intent("android.intent.action.GAMECHALLENGE");
                        startActivity(startGameChallenge);
                        finish();
                  }
            });// end of click listener method
            bChallenge.setTextColor(Color.WHITE);
      }// end start up method for Challenge button

      public void startUpButtonPractice() {
            bPractice = (Button) findViewById(R.id.buttonPractice);
            bPractice.setOnClickListener(new View.OnClickListener() {

                  @Override
                  public void onClick(View v) {
                        Intent startPracticeOptions = new
Intent("android.intent.action.PRACTICEOPTIONS");
                        startActivity(startPracticeOptions);
                        finish();
                  }
            });// end of click listener method
            bPractice.setTextColor(Color.WHITE);
      }// end start up method for Practice button
}
```

GameEMH.java

```java
package com.example.mathgame;

import java.io.BufferedReader;
import java.io.File;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.InputStreamReader;
import java.text.NumberFormat;

import android.app.Activity;
import android.app.AlertDialog;
import android.app.AlertDialog.Builder;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.graphics.Color;
import android.graphics.Typeface;
import android.os.Bundle;
import android.os.SystemClock;
import android.view.View;
import android.widget.Button;
import android.widget.Chronometer;
import android.widget.ImageView;
import android.widget.TextView;

public class GameEMH extends Activity {

	private Math_Game2 mathGame;
	private String[] equations;
	private Button bTrue, bFalse;
	private TextView[] displayEqs;
	private boolean[] checkEq;
	private long time = 0;
	private int questions = 0;
	private Chronometer chrono;
	private int strikes;
	private String difficulty;
	private String[] scores;
	private double highScore;
	private double currentScore;
	private FileOutputStream out;
	private boolean newHighScore;
	private double scoreTime;
	private int scorePenalty;
	private ImageView image;

	@Override
	protected void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState);
		setContentView(R.layout.game);
		Bundle extras = getIntent().getExtras();
		mathGame = new
Math_Game2(extras.getBooleanArray("check"),extras.getIntArray("limits"));
		difficulty = extras.getString("difficulty");
		newHighScore = false;
		startUpDisplay();
		startUpBTrue();
		startUpBFalse();
		startChrono();
		startUpFont();
```

```
      }

      public void startUpFont(){
            Typeface crayon_crumble = Typeface.createFromAsset(getAssets(),
"fonts/dk_crayon_crumble.ttf");
            for (int i = 0; i < displayEqs.length; i++) {
                  displayEqs[i].setTypeface(crayon_crumble);
                  displayEqs[i].setTextColor(Color.WHITE);
            }
            bFalse.setTypeface(crayon_crumble);
            bFalse.setTextColor(Color.WHITE);
            bTrue.setTypeface(crayon_crumble);
            bTrue.setTextColor(Color.WHITE);
            chrono.setTypeface(crayon_crumble);
            chrono.setTextColor(Color.WHITE);
      }

      //when the back button is pressed on the phone
      @Override
      public void onBackPressed() {
            stopChrono();
            displayEqs[1].setText("");
        displayEqs[2].setText("");
        displayEqs[3].setText("");
        displayEqs[4].setText("");
            Builder box = new AlertDialog.Builder(this);
        box.setMessage("Are you sure you want to exit?");
        box.setCancelable(false);
        box.setPositiveButton("Yes", new DialogInterface.OnClickListener() {
            public void onClick(DialogInterface dialog, int id) {
                  Intent startPlayOptions = new
Intent("com.example.mathgame.PLAYOPTIONS");
                        startActivity(startPlayOptions);
                        finish();
            }
        });
        box.setNegativeButton("No", new DialogInterface.OnClickListener() {
            public void onClick(DialogInterface dialog, int id) {
                  startChrono();
                dialog.cancel();
                displayEqs[1].setText(equations[0]);
                displayEqs[2].setText(equations[1]);
                displayEqs[3].setText(equations[2]);
                displayEqs[4].setText(equations[3]);
            }
        });
        box.show();
      }

      public void addingNewEqs() {
            String[] n = mathGame.getAnEquation();
            questions++;
            displayEqs[0].setText(equations[0]);

            for (int i = 0; i < equations.length - 1; i++) {

                  equations[i] = equations[i + 1];
                  checkEq[i] = checkEq[i + 1];
                  displayEqs[i + 1].setText(equations[i]);
            }// end for loop

            equations[equations.length - 1] = n[0];
            checkEq[checkEq.length - 1] = ((n[1].trim().equals("0"))? false : true);
```

```
    displayEqs[displayEqs.length - 1].setText(n[0]);

}// end addingNewEqs method

public void startUpDisplay() {

    int size = 4;
    displayEqs = new TextView[size + 1];
    displayEqs[0] = (TextView) findViewById(R.id.viewLastEquation_EMH);
    displayEqs[1] = (TextView) findViewById(R.id.viewEquation1_EMH);
    displayEqs[2] = (TextView) findViewById(R.id.viewEquation2_EMH);
    displayEqs[3] = (TextView) findViewById(R.id.viewEquation3_EMH);
    displayEqs[4] = (TextView) findViewById(R.id.viewEquation4_EMH);
    image = (ImageView) findViewById(R.id.imageView_EMH);

    equations = new String[size];
    checkEq = new boolean[size];

    for (int i = 0; i < size; i++) {
        String[] n = mathGame.getAnEquation();
        equations[i] = n[0];
        checkEq[i] = ((n[1].trim().equals("0")) ? false : true);
        displayEqs[i + 1].setText(n[0]);
    }// end for loop
}// end startUpDisplay method

public void startUpBTrue() { // setting button true
    bTrue = (Button) findViewById(R.id.buttonTrue_EMH);
    bTrue.setOnClickListener(new View.OnClickListener() {
        public void onClick(View v) {
            if (!checkEq[0]) {
                strikes++;

image.setBackgroundDrawable(getResources().getDrawable(R.drawable.wrong));
            } else {

image.setBackgroundDrawable(getResources().getDrawable(R.drawable.right));
            }
            addingNewEqs();
            if(questions == 20){
                stopChrono();
                scoreTime = calTime();
                scorePenalty = calPenalty();
                calTotalScore();
                saveScore();
                if(newHighScore)
                    newHighScorePopUp();
                else
                    endGamePopUp();
            }
            if(questions == 17){
                displayEqs[4].setText("");
            }
            if(questions == 18){
                displayEqs[3].setText("");
                displayEqs[4].setText("");
            }
            if(questions == 19){
                displayEqs[2].setText("");
                displayEqs[3].setText("");
                displayEqs[4].setText("");
            }
            if(questions == 20){
                displayEqs[1].setText("");
                displayEqs[2].setText("");
                displayEqs[3].setText("");
                displayEqs[4].setText("");
            }
        }// end method onCLick
    });// end of method setOnClickListner from class Button
} // end of public method set up true button

public void startUpBFalse() {// setting button false
    bFalse = (Button) findViewById(R.id.buttonFalse_EMH);
    bFalse.setOnClickListener(new View.OnClickListener() {
        public void onClick(View v) {
            if (checkEq[0]) {
                strikes++;

image.setBackgroundDrawable(getResources().getDrawable(R.drawable.wrong));
            } else {

image.setBackgroundDrawable(getResources().getDrawable(R.drawable.right));
            }
            addingNewEqs();
            if(questions == 20){
                stopChrono();
                scoreTime = calTime();
                scorePenalty = calPenalty();
                calTotalScore();
                saveScore();
                if(newHighScore)
                    newHighScorePopUp();
                else
                    endGamePopUp();
            }
            if(questions == 17){
                displayEqs[4].setText("");
            }
            if(questions == 18){
                displayEqs[3].setText("");
                displayEqs[4].setText("");
            }
            if(questions == 19){
                displayEqs[2].setText("");
                displayEqs[3].setText("");
                displayEqs[4].setText("");
            }
            if(questions == 20){
                displayEqs[1].setText("");
                displayEqs[2].setText("");
                displayEqs[3].setText("");
                displayEqs[4].setText("");
            }
        }// end method onCLick
    });// end of method setOnClickListner from class Button
} // end of public method set up true button

public void startChrono(){
    chrono = (Chronometer)findViewById(R.id.timerEMH);
    chrono.setBase(SystemClock.elapsedRealtime()+time);
    chrono.start();
}

public void stopChrono(){
    time = chrono.getBase() - SystemClock.elapsedRealtime();
```

```
            chrono.stop();
      }
      //calculates the time
      public double calTime(){
            double result;
            result= (SystemClock.elapsedRealtime() - chrono.getBase());
            double endResult = (double)result/1000;

            return endResult;
      }
      //calculates the penalty according to the strikes
      public int calPenalty(){
            int penalty = (strikes * 5);
            return penalty;
      }
      //calcs the total score
      public String calTotalScore(){
            double totalScore = scoreTime + scorePenalty;
            //formats the total score 3 numbers after the dot
            NumberFormat nf = NumberFormat.getInstance();
            nf.setMaximumFractionDigits(3);
            nf.setMaximumFractionDigits(3);
            String output = nf.format(totalScore);
            currentScore = Double.parseDouble(output);
            return output;
      }//end calTotalScore

      public void endGamePopUp(){
                  Builder dialogbox = new AlertDialog.Builder(this);
                  dialogbox.setCancelable(false);
                  dialogbox.setMessage((20 - strikes) + " right out of 20, " + strikes +
" wrong, in " + scoreTime +
                              " seconds + " + scorePenalty + " penalty = " +
calTotalScore() + " seconds." +
                              "\n Do you want to continue playing?");
                  dialogbox.setNegativeButton("no", new DialogInterface.OnClickListener()
{

                        @Override
                        public void onClick(DialogInterface dialog, int which) {
                              Intent startPlayOptions = new
Intent("com.example.mathgame.PLAYOPTIONS");
                              startActivity(startPlayOptions);
                              finish();
                        }
                  });
                  dialogbox.setPositiveButton("yes", new
DialogInterface.OnClickListener() {

                        @Override
                        public void onClick(DialogInterface dialog, int which) {
                              Intent startGameEMH = getIntent();
                              startActivity(startGameEMH);
                              finish();
                        }
                  });
                  dialogbox.show();
      }//end endGamePopUp method

      public boolean fileExistance(String fname){
             File file = getBaseContext().getFileStreamPath(fname);
             if(file.exists()){
                 return true;
```

```
       }
       else{
           return false;
       }
}

public void writeToFile(){
      try{
            String output = scores[0] + " " + scores[1] + " " + scores[2]
                  + " " + scores[3];
            out = openFileOutput("scores", Context.MODE_PRIVATE);
            out.write(output.getBytes());
            out.close();
      }catch(Exception e){
      }
}

public void retrieveScores() throws Exception{
      FileInputStream in = openFileInput("scores");
      InputStreamReader isr = new InputStreamReader(in);
      BufferedReader bufferedReader = new BufferedReader(isr);
      String receiveString = "";
      StringBuilder stringBuilder = new StringBuilder();

      while ( (receiveString = bufferedReader.readLine()) != null ) {
      stringBuilder.append(receiveString);
  }
  in.close();

  scores = stringBuilder.toString().split(" ");
}

public void saveScore(){
      try {
            if(fileExistance("scores")){
                  retrieveScores();
                  if(difficulty.equals("easy")){
                        if(scores[0].equals("#")){
                              scores[0] = currentScore + "";
                              writeToFile();
                        }
                        else{
                              highScore = Double.parseDouble(scores[0]);
                              if(currentScore < highScore){
                              scores[0] = currentScore + "";
                              newHighScore = true;
                              writeToFile();
                        }
                        }
            }
                  else if(difficulty.equals("meduim")){
                        if(scores[1].equals("#")){
                              scores[1] = currentScore + "";
                              writeToFile();
                        }
                        else{
                              highScore = Double.parseDouble(scores[1]);
                              if(currentScore < highScore){
                              scores[1] = currentScore + "";
                              newHighScore = true;
                              writeToFile();
                        }
                        }
```

```
                }
                else{
                    if(scores[2].equals("#")){
                        scores[2] = currentScore + "";
                        writeToFile();
                    }
                    else{
                        highScore = Double.parseDouble(scores[2]);
                        if(currentScore < highScore){
                        scores[2] = currentScore + "";
                        newHighScore = true;
                        writeToFile();
                    }
                    }
                }
            }
            else{
                if(difficulty.equals("easy")){
                    scores = new String[4];
                scores[0] = currentScore + "";
                scores[1] = "#";
                scores[2] = "#";
                scores[3] = "#";
                writeToFile();
            }
                else if(difficulty.equals("meduim")){
                    scores = new String[4];
                scores[0] = "#";
                scores[1] = currentScore + "";
                scores[2] = "#";
                scores[3] = "#";
                writeToFile();
                }
                else{
                    scores = new String[4];
                scores[0] = "#";
                scores[1] = "#";
                scores[2] = currentScore + "";
                scores[3] = "#";
                writeToFile();
                }
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    public void newHighScorePopUp(){
        Builder dialogbox = new AlertDialog.Builder(this);
        dialogbox.setCancelable(false);
        dialogbox.setMessage("Congratulations, you just got a new high score!!!!\n" +
(20 - strikes) +
                " right, " + strikes + " wrong, in " + scoreTime + " seconds + "
+ scorePenalty +
                " penalty = " + calTotalScore()+ " seconds."+ "\n Do you want to
continue playing?");
        dialogbox.setNegativeButton("no", new DialogInterface.OnClickListener() {

            @Override
            public void onClick(DialogInterface dialog, int which) {
                finish();
            }
        });
```

```
            dialogbox.setPositiveButton("yes", new DialogInterface.OnClickListener() {

                  @Override
                  public void onClick(DialogInterface dialog, int which) {
                        Intent startGameEMH = getIntent();
                        startActivity(startGameEMH);
                        finish();
                  }
            });
            dialogbox.show();
            if(!MainMenu.checkSound()){
                  Song.startVictory();
            }
      }
}
```

GameChallenge.java

```
package com.example.mathgame;

import java.io.BufferedReader;
import java.io.File;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.InputStreamReader;

import android.app.Activity;
import android.app.AlertDialog;
import android.app.AlertDialog.Builder;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.graphics.Color;
import android.graphics.Typeface;
import android.os.Bundle;
import android.os.CountDownTimer;
import android.view.View;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.TextView;

public class GameChallenge extends Activity{
      private int questionsDone;
      private String[] equations;
      private boolean[] equationCheck;
      private Button bTrue, bFalse;
      private TextView[] text;
      private CountDownTimer countDown;
      private TextView[] striks;
      private TextView textViewCountDown;
      private boolean[] strikCheck;
      private Math_Game2 mathGame;
      private long s1;
      private FileOutputStream out;
      private String[] scores;
      private boolean newHighScore;
      private ImageView image;

      @Override
      protected void onCreate(Bundle gameChallengeBundle) {
            super.onCreate(gameChallengeBundle);
            this.setContentView(R.layout.game_challenge);
            questionsDone = 0;
            newHighScore = false;
            startUPMathGame();
            startUpTextView();
            startUpButtonFalse();
            startUpButtonTrue();
            startUpStriks();
            startUpFont();
      }

      public void startUpFont(){
            Typeface crayon_crumble = Typeface.createFromAsset(getAssets(),
"fonts/dk_crayon_crumble.ttf");
            bTrue.setTypeface(crayon_crumble);      // set the font for all the buttons to
crayon_crumble
            bTrue.setTextColor(Color.WHITE);
            bFalse.setTypeface(crayon_crumble);
            bFalse.setTextColor(Color.WHITE);
            textViewCountDown.setTypeface(crayon_crumble);
            textViewCountDown.setTextColor(Color.WHITE);
            for (int i = 0; i < text.length; i++) {
                  text[i].setTypeface(crayon_crumble);
                  text[i].setTextColor(Color.WHITE);
            }
      }

      //when the back button is pressed on the phone
      @Override
      public void onBackPressed() {
            if(countDown != null)
                  countDown.cancel();
            text[1].setText("");
            text[2].setText("");
            text[3].setText("");
            text[4].setText("");
            Builder box = new AlertDialog.Builder(this);
        box.setMessage("Are you sure you want to exit?");
        box.setCancelable(false);
        box.setPositiveButton("Yes", new DialogInterface.OnClickListener() {
            public void onClick(DialogInterface dialog, int id) {
                  Intent startPlayOptions = new
Intent("com.example.mathgame.PLAYOPTIONS");
                        startActivity(startPlayOptions);
                        finish();
            }
        });
        box.setNegativeButton("No", new DialogInterface.OnClickListener() {
            public void onClick(DialogInterface dialog, int id) {
                  if(countDown != null){
                        countDown = new CountDown(s1,100);
                  countDown.start();
                    dialog.cancel();
                    text[1].setText(equations[0]);
                        text[2].setText(equations[1]);
                        text[3].setText(equations[2]);
                        text[4].setText(equations[3]);
                  }
                  else{
                        text[1].setText(equations[0]);
                        text[2].setText(equations[1]);
                        text[3].setText(equations[2]);
                        text[4].setText(equations[3]);
                  }
            }
        });
       box.show();
      }
      //start up method
      public void startUpButtonTrue(){
            bTrue = (Button) findViewById(R.id.buttonTrue_Challenge);
            bTrue.setOnClickListener(new View.OnClickListener() {

                  @Override
                  public void onClick(View v) {
                        if(!equationCheck[0]){
                              try {
                                    addStrike();
                              } catch (Exception e) {
                                    e.printStackTrace();
                              }

      image.setBackgroundDrawable(getResources().getDrawable(R.drawable.wrong));
                        } else {

      image.setBackgroundDrawable(getResources().getDrawable(R.drawable.right));
                        }
                  addNextEquation();
                  }
            });//end of method clickListener
      }//end of method StartUpButtonTrue
      public void startUpButtonFalse(){
            bFalse = (Button) findViewById(R.id.buttonFalse_Challenge);
            bFalse.setOnClickListener(new View.OnClickListener() {

                  @Override
                  public void onClick(View v) {
                        if(equationCheck[0]){
                              try {
                                    addStrike();
                              } catch (Exception e) {
                                    e.printStackTrace();
                              }

      image.setBackgroundDrawable(getResources().getDrawable(R.drawable.wrong));
                        } else {

      image.setBackgroundDrawable(getResources().getDrawable(R.drawable.right));
                        }
                  addNextEquation();
                  }
            });//end of method clickListener
      }//end of method StartUpButtonFalse
      public void startUpTextView(){
            int size = 4;

            text = new TextView[size + 1];
            text[0] = (TextView) findViewById(R.id.viewLastEquation_Challenge);
            text[1] = (TextView) findViewById(R.id.viewEquation1_Challenge);

            text[2] = (TextView) findViewById(R.id.viewEquation2_Challenge);
            text[3] = (TextView) findViewById(R.id.viewEquation3_Challenge);
            text[4] = (TextView) findViewById(R.id.viewEquation4_Challenge);

            image = (ImageView) findViewById(R.id.imageView_Challenge);

            equations = new String[size];
            equationCheck = new boolean[size];

            for (int i = 0; i < size; i++) {
                  String[] n = mathGame.getAnEquation();
                  equations[i] = n[0];
                  equationCheck[i] = ((n[1].trim().equals("0"))? false : true);
                  text[i + 1].setText(n[0]);
            }//end for loop

            textViewCountDown = (TextView) findViewById(R.id.countDownChallenge);
            textViewCountDown.setText("5.00");

      }//end method StartUpTextView
      public void startUpStriks(){
            striks = new TextView[3];
            strikCheck = new boolean[3];
```

```
        striks[0] = (TextView) findViewById(R.id.textViewStrick1);
        striks[1] = (TextView) findViewById(R.id.textViewStrick2);
        striks[2] = (TextView) findViewById(R.id.textViewStrick3);

        for (int i = 0; i < striks.length; i++) {
            strikCheck[i] = true;
        }//end for loop
    }//end method startUpStriks

    public void startUPMathGame(){
        boolean[] check = {true,true,true,true};
        int[] num = {0,10,0,10,0,10,0,10};
        mathGame = new Math_Game2(check,num);
    }//end method startUpMathGame

    public void startUpCountDown(){
        if(strikCheck[2]){
            countDown = new CountDown(5000,50);
            countDown.start();
        }
    }

    public void startUpPopMenu(){

        textViewCountDown.setText("0.00");
        Builder dialogBox = new AlertDialog.Builder(this);
        dialogBox.setTitle("Game Over");
        dialogBox.setMessage("You solved " + (questionsDone - 3)
                + " questions\n Do you want to continue? ");
        dialogBox.setCancelable(false);
        dialogBox.setNegativeButton("No",
                new DialogInterface.OnClickListener() {
                    @Override
                    public void onClick(DialogInterface dialog, int which) {
                        try {
                            writeToFile();
                        } catch (Exception e) {
                            e.printStackTrace();
                        }
                        Intent startPlayOptions = new
Intent("com.example.mathgame.PLAYOPTIONS");
                        startActivity(startPlayOptions);
                        finish();
                    }
                });
        dialogBox.setPositiveButton("Yes", new DialogInterface.OnClickListener() {
            //restarts game challenge when user want to continue
            @Override
            public void onClick(DialogInterface dialog, int which) {
                try {
                    writeToFile();
                } catch (Exception e) {
                    e.printStackTrace();
                }
                Intent startGameChallenge = getIntent();
                startActivity(startGameChallenge);
                finish();
            }
        });
        dialogBox.show();
    }
    //making dialog box
    //other public methods
    public void addNextEquation(){
        String[] n = mathGame.getAnEquation();
        mathGame.increaseDificult(questionsDone++);

        text[0].setText(equations[0]);

        for (int i = 0; i < equations.length - 1; i++) {// move all the equation by 1 
up
            equations[i] = equations[i + 1];
            equationCheck[i] = equationCheck[i + 1];
            text[i + 1].setText(equations[i]);
        }// end for loop

        equations[equations.length - 1] = n[0];
        equationCheck[equationCheck.length - 1] = ((n[1].trim().equals("0"))? false : 
true);
        text[text.length - 1].setText(n[0]);
        if(countDown != null)
            countDown.cancel();
        startUpCountDown();
    }
    public void addStrike() throws Exception{

        for (int i = 0; i < strikCheck.length; i++) {
            if(strikCheck[i]){

                strikCheck[i] = false;

    striks[i].setBackgroundDrawable(getResources().getDrawable(R.drawable.red_apple));
                if(i == 2){
                    saveScore();
                    if(newHighScore)
                        newHighScorePopUp();
                    else
                        startUpPopMenu();
                }
                break;
            }// end if
        }//end for loop
    }//end method strike

    //inner class
    public class CountDown extends CountDownTimer{

        public CountDown(long millisInFuture, long countDownInterval) {
            super(millisInFuture, countDownInterval);
        }

        @Override
        public void onFinish() {
            try {
                addStrike();
            } catch (Exception e) {
                e.printStackTrace();
            }
            addNextEquation();

    image.setBackgroundDrawable(getResources().getDrawable(R.drawable.wrong));
        }

        @Override
```

```
            public void onTick(long millisUntilFinished) {
                  s1=millisUntilFinished;
                  textViewCountDown.setText("" + String.format("%.2f",
millisUntilFinished/1000.0  ));
            }
      }

      public boolean fileExistance(String fname){
             File file = getBaseContext().getFileStreamPath(fname);
             if(file.exists()){
                 return true;
             }
             else{
                 return false;
             }
      }

      public void retrieveScores() throws Exception{
            FileInputStream in = openFileInput("scores");
            InputStreamReader isr = new InputStreamReader(in);
            BufferedReader bufferedReader = new BufferedReader(isr);
            String receiveString = "";
            StringBuilder stringBuilder = new StringBuilder();

            while ( (receiveString = bufferedReader.readLine()) != null ) {
            stringBuilder.append(receiveString);
        }
        in.close();
        scores = stringBuilder.toString().split(" ");
      }

      public void writeToFile(){
            try{
                  String output = scores[0] + " " + scores[1] + " " + scores[2]
                        + " " + scores[3];
                  out = openFileOutput("scores", Context.MODE_PRIVATE);
                  out.write(output.getBytes());
                  out.close();
            }catch(Exception e){
            }
      }

      public void saveScore() throws Exception{
            if(fileExistance("scores")){
                  retrieveScores();
                  if(scores[3].equals("#")){
                        scores[3] = (questionsDone - 2) + "";
                  }else{
                        int highScore = Integer.parseInt(scores[3]);
                        if((questionsDone - 2) > highScore){
                              scores[3] = (questionsDone - 2) + "";
                              newHighScore = true;
                        }
                  }
            }else{
                  scores = new String[4];
                  scores[0] = "#";
            scores[1] = "#";
            scores[2] = "#";
            scores[3] = (questionsDone - 2) + "";
            }
            questionsDone++;
      }
```

```java
    public void newHighScorePopUp(){
        textViewCountDown.setText("0.00");
        Builder dialogBox = new AlertDialog.Builder(this);
        dialogBox.setTitle("Game Over");
        dialogBox.setMessage("Congratulations, you just got a new high score in
Challenge mode!!!!\n" +
                "You solved " + (questionsDone - 3)
                + " questions\n Do you want to continue? ");
        dialogBox.setCancelable(false);
        dialogBox.setNegativeButton("No",
                new DialogInterface.OnClickListener() {
                    @Override
                    public void onClick(DialogInterface dialog, int which) {
                        try {
                            writeToFile();
                        } catch (Exception e) {
                            e.printStackTrace();
                        }
                        finish();
                    }
                });
        dialogBox.setPositiveButton("Yes", new DialogInterface.OnClickListener() {
            //restarts game challenge when user want to continue
            @Override
            public void onClick(DialogInterface dialog, int which) {
                try {
                    writeToFile();
                } catch (Exception e) {
                    e.printStackTrace();
                }
                Intent startGameChallenge = getIntent();
                startActivity(startGameChallenge);
                finish();
            }
        });
        dialogBox.show();
        if(!MainMenu.checkSound()){
            Song.startVictory();
        }
    }
}
```

GamePractice.java

```
package com.example.mathgame;

import android.app.Activity;
import android.content.Intent;
import android.graphics.Color;
import android.graphics.Typeface;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.TextView;

public class Game_Practice extends Activity {

      private Button bMenu, bTrue, bFalse;
      private TextView[] texts;
       //equation1, equation2, equation3, equation4, lastEquation, percentRight
      private Math_Game2 math;
      private String[] equation1;
      private String[] equation2;
      private String[] equation3;
      private String[] equation4;
      private double questionsDone;
      private double questionsRight;
      private String percent;
      private ImageView image;

      @Override
      protected void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.game_practice);
            Bundle extras = getIntent().getExtras();
            math = new Math_Game2(extras.getBooleanArray("operatorsActive"),
extras.getIntArray("limitsValue"));
            questionsDone = 0;
            questionsRight = 0;

            startUpTextViews();
            startUpButtonMenu();
            startUpButtonTrue();
            startUpButtonFalse();
            startUpFont();
      }

      @Override
      public void onBackPressed() {
            super.onBackPressed();
            Intent startPracticeOptions = new
Intent("android.intent.action.PRACTICEOPTIONS");
            startActivity(startPracticeOptions);
            finish();
      }

      public void startUpFont(){
            Typeface crayon_crumble = Typeface.createFromAsset(getAssets(),
"fonts/dk_crayon_crumble.ttf");
            for (int i = 0; i < texts.length; i++) {
                  texts[i].setTypeface(crayon_crumble);
                  texts[i].setTextColor(Color.WHITE);
            }
```

```
        bFalse.setTypeface(crayon_crumble);
        bFalse.setTextColor(Color.WHITE);
        bTrue.setTypeface(crayon_crumble);
        bTrue.setTextColor(Color.WHITE);
        bMenu.setTypeface(crayon_crumble);
        bMenu.setTextColor(Color.WHITE);
    }

    public void startUpTextViews(){
        texts = new TextView[6];
        texts[0] = (TextView) findViewById(R.id.viewEquation1_Practice);
        texts[1] = (TextView) findViewById(R.id.viewEquation2_Practice);
        texts[2] = (TextView) findViewById(R.id.viewEquation3_Practice);
        texts[3] = (TextView) findViewById(R.id.viewEquation4_Practice);
        texts[4] = (TextView) findViewById(R.id.viewLastEquation_Practice);
        texts[5] = (TextView) findViewById(R.id.viewPercentRight_Practice);

        equation1 = math.getAnEquation();
        equation2 = math.getAnEquation();
        equation3 = math.getAnEquation();
        equation4 = math.getAnEquation();
        texts[0].setText(equation1[0]);
        texts[1].setText(equation2[0]);
        texts[2].setText(equation3[0]);
        texts[3].setText(equation4[0]);

        image = (ImageView) findViewById(R.id.imageView_Practice);
    }

    public void startUpButtonMenu(){
        bMenu = (Button) findViewById(R.id.buttonMenu_Practice);
        bMenu.setOnClickListener(new View.OnClickListener() {

            @Override
            public void onClick(View v) {
                Intent intent = new Intent(getApplicationContext(),
MainMenu.class);
                intent.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
                startActivity(intent);
            }
        });
    }

    public void startUpButtonTrue(){
        bTrue = (Button) findViewById(R.id.buttonTrue_Practice);
        bTrue.setOnClickListener(new View.OnClickListener() {

            @Override
            public void onClick(View v) {
                texts[4].setText(equation1[0]);
                if(equation1[1].equals("1")){

    image.setBackgroundDrawable(getResources().getDrawable(R.drawable.right));
                    questionsRight++;
                }
                else

    image.setBackgroundDrawable(getResources().getDrawable(R.drawable.wrong));
                update();
            }
        });
    }
    public void startUpButtonFalse(){
        bFalse = (Button) findViewById(R.id.buttonFalse_Practice);
        bFalse.setOnClickListener(new View.OnClickListener() {

            @Override
            public void onClick(View v) {
                texts[4].setText(equation1[0]);
                if(equation1[1].equals("0")){

    image.setBackgroundDrawable(getResources().getDrawable(R.drawable.right));
                    questionsRight++;
                }
                else

    image.setBackgroundDrawable(getResources().getDrawable(R.drawable.wrong));
                update();
            }
        });
    }

    public void update(){
        questionsDone++;
        texts[0].setText(equation2[0]);
        texts[1].setText(equation3[0]);
        texts[2].setText(equation4[0]);
        equation1 = equation2;
        equation2 = equation3;
        equation3 = equation4;
        equation4 = math.getAnEquation();
        texts[3].setText(equation4[0]);
        double percentNum = (questionsRight/questionsDone) * 100;
        percentNum = (double)Math.round(percentNum*10)/10;
        percent = percentNum + "%";
        texts[5].setText(percent);
    }

}
```

Math_Game2.java

```java
package com.example.mathgame;

public class Math_Game2 {
      private int[] limits; // pos 0: addMin, pos 1: addMax, pos 2: subMin, pos 3: subMax
                                      // pos 4: mulMin, pos 5: mulMax, pos 6: divMin, pos
7: divMax
      private boolean[] signs; // pos 0 : +, pos 1 : -, pos 2: *, pos 3: /;

      // constructors
      public Math_Game2(boolean[] signs, int[] limits) {
            setSigns(signs);
            setLimits(limits);
      }

      // setters
      public void setSigns(boolean[] signs) {
            this.signs = signs;
      }

      public void setLimits(int[] limits) {
            this.limits = limits;
      }

      // getters
      public int[] getLimits() {
            return limits;
      }

      public boolean[] getSigns() {
            return signs;
      }

      // methods
      public String[] getAnEquation() {
            // pos 0: equation , pos 1: 0 is false___ 1 is true
            while (true) {
                  switch ((int) (Math.random() * (4))) {
                  case 0:
                        if (signs[0])
                              return creatAdditionalEquation();
                        else
                              continue;
                  case 1:
                        if (signs[1])
                              return creatSubtractionEquation();
                        else
                              continue;
                  case 2:
                        if (signs[2])
                              return creatMultiplicationEquation();
                        else
                              continue;
                  case 3:
                        if (signs[3])
                              return creatDivisonEquation();
                        else
                              continue;
                  }// end of switch
            }// end of while loop
      }// end of getAnEquation method
      public void increaseDificult(int num){
            for (int i = 0; i < limits.length; i++) {
                  if(i < 4 ){
                        if(i%2 == 1) // check if is a max
                              limits[i]++;
                        else if(i%2 == 0 && num%5 == 0 && num != 0) // check if is a min
                              limits[i]++;
                  } else if(i%2 == 1 && num%4 == 0 && num != 0)
                        limits[i]++;
                  else if(i%2 == 0 && num%9 == 0 && num != 0)
                        limits[i]++;
            }
      }
      // private methods
      private String[] creatAdditionalEquation() {
            int num1 = (int) (limits[0] + Math.random() * (limits[1] + 1 - limits[0]));
// make

                  // Number 1

            int num2 = (int) (limits[0] + Math.random() * (limits[1] + 1 - limits[0]));
// make

                  // number 2
            int equals = num1 + num2;  // get the sum
            boolean check = true;  // is going to say if the equation is true or false

            if (Math.random() < 0.5) { // is going to set the equation false if the
                                                      // number is less than 0.5

                  check = false;    // set check to false
                  int equals2 = equals;  // we have to change the equal
                  while (equals == equals2) {

                        if (((num1 == 1) || (num2 == 1)) && (Math.random() < 0.6)){ // if
for getting a false equation when the result is 1
                                    if (num1 == 1) {
                                          equals2 = num2;
                                          continue;
                                    } else if (num2 == 1) {
                                          equals2 = num1;
                                          continue;
                              }
                        }else {
                                    if (limits[1] - limits[0] > 5) {
                                          int change = (int)((limits[1] - limits[0]) *
0.2);

                                          if ((num1 > change || num2 > change)
                                                      && Math.random() < 0.5)
                                                equals2 = num2 + num1 - (int)(1 + ( 1
+change)*Math.random());  // subtract change
                                          else
                                                equals2 = num2 + num1 + (int)(1 + ( 1
+change)*Math.random()); //add change

                                    } else {

                                          equals2 = (int) (limits[0] + Math.random()
                                                      * (2 * limits[1] + 1));
                                    }
                              } // end of else
                  } // end of while loop
                  equals = equals2;

            } // end of if
            String[] equation = { num1 + " + " + num2 + " = " + equals,
                        (check) ? 1 + "" : 0 + "" };  // one means true, 0 means false
            return equation;
      }

      private String[] creatSubtractionEquation() {
            int num1 = (int) (limits[2] + Math.random() * (limits[3] + 1 - limits[2]));
// make

                  // number 1
            int num2 = (int) (limits[2] + Math.random() * (limits[3] + 1 - limits[2]));
// make

                  // number 2
            if (num1 < num2) {
                  int switchh = num1;
                  num1 = num2;
                  num2 = switchh;
            }

            int equals = num1 - num2;
            boolean check = true;

            if (Math.random() < 0.5) { // is going to set the equation false if the
                                                      // number is less than 0.5
                  check = false;
                  int equals2 = equals;
                  while (equals == equals2) {

                        if (((num1 == 1) || (num2 == 1)) && (Math.random() < 0.6)){ // if
for getting a false equation when the result is 1
                                    if (num1 == 1) {
                                          equals2 = num2;
                                          continue;
                                    } else if (num2 == 1) {
                                          equals2 = num1;
                                          continue;
                              }
                        }else {
                                    if (limits[3] - limits[2] > 5) {
                                          int change = (int)((limits[3] - limits[2]) *
0.2);

                                          if (( (num1 - num2) > change)
                                                      && Math.random() < 0.5)
                                                equals2 = num1 - num2 - (int)(1 + ( 1
+change)*Math.random());  // subtract change
                                          else
                                                equals2 = num1 - num2 + (int)(1 + ( 1
+change)*Math.random()); //add change

                                    } else {

                                          equals2 = (int) (limits[2] + Math.random()
                                                      * (limits[3] + 1));
                                    }
                              } // end of else

                  } // end of while loop
                  equals = equals2;

            } // end of if
            String[] equation = { num1 + " - " + num2 + " = " + equals,
                        (check) ? 1 + "" : 0 + "" };
            return equation;
      }

      private String[] creatMultiplicationEquation() {
            int[] num = getMultiplication(limits[4], limits[5]);
            boolean check = true;

            if (Math.random() < 0.5) { // is going to set the equation false if the
                                                      // number is less than 0.5
                  check = false;
                  int equals2 = num[2];
                  while (num[2] == equals2) {

                        if (((num[0] == 1) || (num[1] == 1)) && Math.random() < 0.6){
                                    equals2 = (num[0] > num[1]) ? num[0] + 1 : num[1] +
1;

                                    continue;

                              } else if ((num[0] == 0) || (num[1] == 0)) {
                                    equals2 = (num[0] > num[1]) ? num[0] : num[1];
                                    continue;

                              } else if (limits[5] - limits[4] > 5) {
                                    int change = (int)((limits[5] - limits[4]) * 0.2);

                                    if(num[2] > change && Math.random() < 0.5){
                                          equals2 = (int) (num[2] - (1 + (change +
1)*Math.random()));
                                    }
                                    else equals2 = (int) (num[2] + (1 + (change +
1)*Math.random()));
                              } else {

                                    equals2 = (int) (limits[4] + Math.random()
                                                * (limits[5] * limits[5] + 1));
                                    continue;
                              }
                  } // end of while loop
                  num[2] = equals2;

            } // end of if

            String[] equation = { num[0] + " x " + num[1] + " = " + num[2],
                        (check) ? 1 + "" : 0 + "" };
            return equation;
      }

      private String[] creatDivisonEquation() {
            int[] num = getMultiplication(limits[6], limits[7]);

            {// make switch
                  int switchh;// this variable can only be use inside the switch
                  if (num[1] == 0) { // numbers can't be divided by 0
                        switchh = num[0];
                        num[0] = num[1];
                        num[1] = switchh;
                  }
                  switchh = num[0];
```

```
                num[0] = num[2];
                num[2] = switchh;
            }// end of switch
            boolean check = true;
            if (Math.random() < 0.5) { // is going to set the equation false if the
                                          // number is less than 0.5
                check = false;
                int equals2 = num[2];
                while (num[2] == equals2) {
                    if (num[0] == 1) {
                        equals2 = (Math.random() < 0.5) ? num[1] - 1 : num[1] + 1;
                        continue;
                    } else if (num[0] == 0) {
                        equals2 = num[1];
                        continue;
                    }
                    if (num[1] == 1) {
                        equals2 = (Math.random() < 0.5) ? num[0] - 1 : num[0] + 1;
                        continue;
                    } else if (num[1] == 0) {
                        equals2 = num[0];
                        continue;
                    }

                    if (limits[7] - limits[6] > 5){
                        int change = (int)((limits[5] - limits[4]) * 0.2);
                        if(num[2] > change && Math.random() < 0.5)
                            equals2 = (int) (num[2] - (1 + (change +
1)*Math.random()));

                            else equals2 = (int) (num[2] + (1 + (change +
1)*Math.random()));
                    }
                    else equals2 = (int) (limits[6] + Math.random() * (limits[7] +
1));
                } // end of while loop
                num[2] = equals2;

            } // end of if
            String[] equation = { num[0] + " / " + num[1] + " = " + num[2],
                    (check) ? 1 + "" : 0 + "" };
            return equation;
        }

        private int[] getMultiplication(int min, int max) {
            int num1, num2;
            do {
             num1 = (int) (min + Math.random() * (max + 1 - min));
             num2 = (int) (min + Math.random() * (max + 1 - min));
            }while (num1 == 0 && num2 == 0); // the two numbers cannot be 0

            int equals = num1 * num2;
            int[] num = { num1, num2, equals };
            return num;
        }
}
```

PracticeOptions.java

```
package com.example.mathgame;

import java.io.BufferedReader;
import java.io.File;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.InputStreamReader;
import android.app.Activity;
import android.app.AlertDialog;
import android.app.AlertDialog.Builder;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.graphics.Color;
import android.graphics.Typeface;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.CheckBox;
import android.widget.EditText;
import android.widget.TextView;

public class PracticeOptions extends Activity{

      private Button bDone, bDefault;
      private CheckBox[] checkBoxes; //pos0: Add, pos1: Sub, pos2: Mul, pos3: Div
      private EditText[] limits; //pos0: AddMin, pos1: AddMax, pos2: SubMin, pos3: SubMax
                                             //pos4: MulMin, pos5: MulMax, pos6: DivMin,
pos7: DivMax
      private boolean[] operatorsActive;
      private int[] limitsValue;
      private String options;
      private String[] tokens;
      private FileOutputStream out;
      private TextView[] texts; // pos0: title, pos1-4: from1-4, pos5-8: to1-4

      @Override
      protected void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.practice_options);
            startUpCheckBoxes();
            startUpTextFields();
            startUpOptions();
            startUpButtonDone();
            startUpButtonDefault();

            texts = new TextView[9];
            texts[0] = (TextView) findViewById(R.id.textViewOptions);
            texts[1] = (TextView) findViewById(R.id.textViewFrom1);
            texts[2] = (TextView) findViewById(R.id.textViewFrom2);
            texts[3] = (TextView) findViewById(R.id.textViewFrom3);
            texts[4] = (TextView) findViewById(R.id.textViewFrom4);
            texts[5] = (TextView) findViewById(R.id.textViewTo1);
            texts[6] = (TextView) findViewById(R.id.textViewTo2);
            texts[7] = (TextView) findViewById(R.id.textViewTo3);
            texts[8] = (TextView) findViewById(R.id.textViewTo4);

            startUpFont();
      }
```

```
    @Override
    public void onBackPressed() {
        super.onBackPressed();
        Intent startPlayOptions = new Intent("com.example.mathgame.PLAYOPTIONS");
        startActivity(startPlayOptions);
        finish();
    }

    public void startUpFont(){
        Typeface crayon_crumble = Typeface.createFromAsset(getAssets(),
"fonts/dk_crayon_crumble.ttf");
        bDone.setTypeface(crayon_crumble);      // set the font for all the buttons to
crayon_crumble
        bDone.setTextColor(Color.WHITE);
        bDefault.setTypeface(crayon_crumble);
        bDefault.setTextColor(Color.WHITE);

        for (int i = 0; i < checkBoxes.length; i++) {
            checkBoxes[i].setTypeface(crayon_crumble);
            checkBoxes[i].setTextColor(Color.WHITE);
        }
        for (int i = 0; i < limits.length; i++) {
            limits[i].setTypeface(crayon_crumble);
            limits[i].setBackgroundColor(Color.TRANSPARENT);
            limits[i].setTextColor(Color.WHITE);
        }

        for (int i = 0; i < texts.length; i++) {
            texts[i].setTypeface(crayon_crumble);
            texts[i].setTextColor(Color.WHITE);
        }
    }

    public void startUpCheckBoxes(){
        checkBoxes = new CheckBox[4];
        checkBoxes[0] = (CheckBox) findViewById(R.id.checkBoxAdd);
        checkBoxes[1] = (CheckBox) findViewById(R.id.checkBoxSub);
        checkBoxes[2] = (CheckBox) findViewById(R.id.checkBoxMul);
        checkBoxes[3] = (CheckBox) findViewById(R.id.checkBoxDiv);
    }

    public void startUpTextFields(){
        limits = new EditText[8];
        limits[0] = (EditText) findViewById(R.id.editAddMin);
        limits[1] = (EditText) findViewById(R.id.editAddMax);
        limits[2] = (EditText) findViewById(R.id.editSubMin);
        limits[3] = (EditText) findViewById(R.id.editSubMax);
        limits[4] = (EditText) findViewById(R.id.editMulMin);
        limits[5] = (EditText) findViewById(R.id.editMulMax);
        limits[6] = (EditText) findViewById(R.id.editDivMin);
        limits[7] = (EditText) findViewById(R.id.editDivMax);
    }

    public void startUpOptions(){
        if(fileExistance("practiceOptions")){
            retrieveOptions();

        checkBoxes[0].setChecked((tokens[0].equals("true")) ? true : false);
            checkBoxes[1].setChecked((tokens[1].equals("true")) ? true : false);
            checkBoxes[2].setChecked((tokens[2].equals("true")) ? true : false);
            checkBoxes[3].setChecked((tokens[3].equals("true")) ? true : false);
            limits[0].setText(tokens[4]);
            limits[1].setText(tokens[5]);
```

```
                  limits[2].setText(tokens[6]);
                  limits[3].setText(tokens[7]);
                  limits[4].setText(tokens[8]);
                  limits[5].setText(tokens[9]);
                  limits[6].setText(tokens[10]);
                  limits[7].setText(tokens[11]);

            }
            else{
                  checkBoxes[0].setChecked(true);
                  checkBoxes[1].setChecked(true);
                  checkBoxes[2].setChecked(true);
                  checkBoxes[3].setChecked(true);
                  limits[0].setText("0");
                  limits[1].setText("20");
                  limits[2].setText("0");
                  limits[3].setText("20");
                  limits[4].setText("0");
                  limits[5].setText("12");
                  limits[6].setText("0");
                  limits[7].setText("12");
            }
      }

      public boolean fileExistance(String fname){
             File file = getBaseContext().getFileStreamPath(fname);
             if(file.exists()){
                 return true;
             }
             else{
                 return false;
             }
      }

      public void startUpButtonDone(){
            bDone = (Button) findViewById(R.id.buttonDone);
            bDone.setOnClickListener(new View.OnClickListener() {

                  @Override
                  public void onClick(View v) {
                        if(checkOptions()){
                              errorPopUp();
                        }else{
                              operatorsActive = new boolean[4];
                              operatorsActive[0] = checkBoxes[0].isChecked();
                              operatorsActive[1] = checkBoxes[1].isChecked();
                              operatorsActive[2] = checkBoxes[2].isChecked();
                              operatorsActive[3] = checkBoxes[3].isChecked();

                              limitsValue = new int[8];
                              limitsValue[0] =
Integer.parseInt(limits[0].getText().toString());
                              limitsValue[1] =
Integer.parseInt(limits[1].getText().toString());
                              limitsValue[2] =
Integer.parseInt(limits[2].getText().toString());
                              limitsValue[3] =
Integer.parseInt(limits[3].getText().toString());
                              limitsValue[4] =
Integer.parseInt(limits[4].getText().toString());
                              limitsValue[5] =
Integer.parseInt(limits[5].getText().toString());
```

```
                              limitsValue[6] = 
Integer.parseInt(limits[6].getText().toString());
                              limitsValue[7] = 
Integer.parseInt(limits[7].getText().toString());

                              saveOptions();

                              Intent startGamePractice = new 
Intent("android.intent.action.GAMEPRACTICE");
                              startGamePractice.putExtra("operatorsActive", 
operatorsActive);
                              startGamePractice.putExtra("limitsValue", limitsValue);
                              startActivity(startGamePractice);
                              finish();
                        }
                  }
            });
      }

      public void startUpButtonDefault(){
            bDefault = (Button) findViewById(R.id.buttonDefault);
            bDefault.setOnClickListener(new View.OnClickListener() {

                  @Override
                  public void onClick(View v) {
                        checkBoxes[0].setChecked(true);
                        checkBoxes[1].setChecked(true);
                        checkBoxes[2].setChecked(true);
                        checkBoxes[3].setChecked(true);
                        limits[0].setText("0");
                        limits[1].setText("20");
                        limits[2].setText("0");
                        limits[3].setText("20");
                        limits[4].setText("0");
                        limits[5].setText("12");
                        limits[6].setText("0");
                        limits[7].setText("12");
                  }
            });
      }

      public void saveOptions(){
            try {
                  out = openFileOutput("practiceOptions", Context.MODE_PRIVATE);
                  options = ((operatorsActive[0]) ? "true" : "false") + " " +
                              ((operatorsActive[1]) ? "true" : "false") + " " +
                              ((operatorsActive[2]) ? "true" : "false") + " " +
                              ((operatorsActive[3]) ? "true" : "false") + " " +
                              limitsValue[0] + " " + limitsValue[1] + " " +
                              limitsValue[2] + " " + limitsValue[3] + " " +
                              limitsValue[4] + " " + limitsValue[5] + " " +
                              limitsValue[6] + " " + limitsValue[7];
                  out.write(options.getBytes());
                  out.close();
            } catch (Exception e) {
                  e.printStackTrace();
            }
      }

      public void retrieveOptions(){

            try {
                  FileInputStream in = openFileInput("practiceOptions");
                  InputStreamReader isr = new InputStreamReader(in);
                  BufferedReader bufferedReader = new BufferedReader(isr);
                  String receiveString = "";
                  StringBuilder stringBuilder = new StringBuilder();

                  while ( (receiveString = bufferedReader.readLine()) != null ) {
                stringBuilder.append(receiveString);
            }
                  in.close();

                  options = stringBuilder.toString();
                  tokens = options.split(" ");

            } catch (Exception e) {
                  e.printStackTrace();
            }
      }

      public boolean checkOptions(){
            int min;
            int max;
            int checkBoxesChecked = 0;
            int temp = 0;
            for (int i = 0; i < limits.length; i++) {
                  if(limits[i].getText().toString().matches(""))
                        limits[i].setText("0");
            }
            for (int i = 0; i < 4; i++) {
                  if(checkBoxes[i].isChecked()){
                        checkBoxesChecked++;
                  }
            }
            if(checkBoxesChecked == 0)
                  return true;
            for (int i = 0; i < checkBoxes.length; i++) {
                  if(checkBoxes[i].isChecked()){
                        min = Integer.parseInt(limits[temp].getText().toString());
                        max = Integer.parseInt(limits[temp+1].getText().toString());
                        if(i == 0 || i == 1){
                              if(min > 999 || max > 999)
                                    return true;
                        }
                        if(i == 2 || i == 3){
                              if(min > 999 || max > 999)
                                    return true;
                        }
                        if(min == 0 && max == 0)
                              limits[temp+1].setText("1");
                        if(min > max){
                              limits[temp].setText(max + "");
                              limits[temp+1].setText(min + "");
                        }
                  }
                  else{
                        if(limits[temp].getText().toString().equals("0") &&
limits[temp+1].getText().toString().equals("0"))
                              limits[temp+1].setText("1");
                  }
                  temp += 2;
            }
            return false;
      }
```

```
      public void errorPopUp(){
            Builder box = new AlertDialog.Builder(this);
        box.setMessage("You must check at least one operator. Minimum and maximum" +
                  " for selected operator cannot exceed 999 and cannot be 0, 0.");
        box.setCancelable(false);
        box.setPositiveButton("OK", new DialogInterface.OnClickListener() {
            public void onClick(DialogInterface dialog, int id) {
            }
        });
        box.show();
      }
}
```

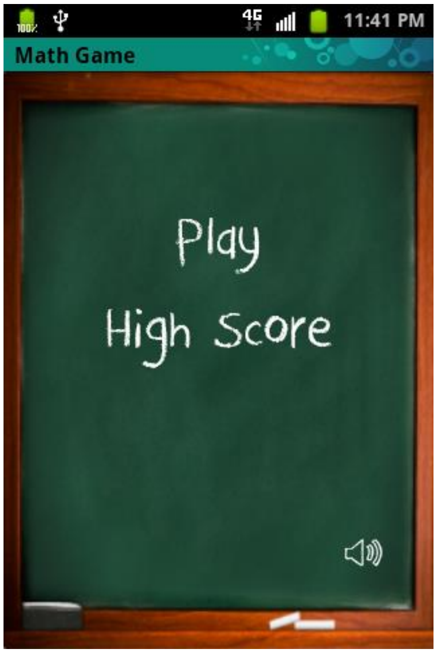
Score.java

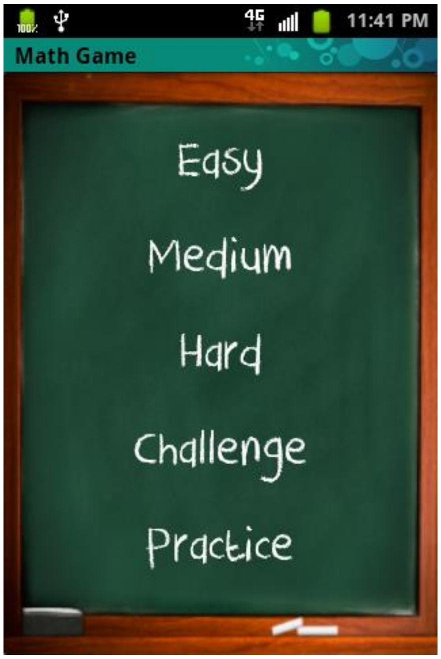
```java
package com.example.mathgame;

import java.io.BufferedReader;
import java.io.File;
import java.io.FileInputStream;
import java.io.InputStreamReader;
import android.app.Activity;
import android.app.AlertDialog;
import android.app.AlertDialog.Builder;
import android.content.DialogInterface;
import android.content.Intent;
import android.graphics.Color;
import android.graphics.Typeface;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;

public class Score extends Activity{

      private String scores;
      private TextView textScoreEasy;
      private TextView textScoreMeduim;
      private TextView textScoreHard;
      private TextView textScoreChallenge;
      private Button bDelete;
      private TextView[] texts; //pos0: title, pos1:easy, pos2:medium, pos3:hard, pos4:
challenge

      @Override
      protected void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.score);
            textScoreEasy = (TextView) findViewById(R.id.textViewScoreEasy);
            textScoreMeduim = (TextView) findViewById(R.id.textViewScoreMeduim);
            textScoreHard = (TextView) findViewById(R.id.textViewScoreHard);
            textScoreChallenge = (TextView) findViewById(R.id.textViewScoreChallenge);
            startUpButtonDelete();
            try {
                  FileInputStream in = openFileInput("scores");
                  InputStreamReader isr = new InputStreamReader(in);
                  BufferedReader bufferedReader = new BufferedReader(isr);
                  String receiveString = "";
                  StringBuilder stringBuilder = new StringBuilder();

                  while ( (receiveString = bufferedReader.readLine()) != null ) {
                stringBuilder.append(receiveString);
            }
            in.close();

            scores = stringBuilder.toString();
            String[] tokens = scores.split(" ");
            textScoreEasy.setText((tokens[0].equals("#"))? "Empty": tokens[0]);
            textScoreMeduim.setText((tokens[1].equals("#"))? "Empty": tokens[1]);
            textScoreHard.setText((tokens[2].equals("#"))? "Empty": tokens[2]);
            textScoreChallenge.setText((tokens[3].equals("#"))? "Empty": tokens[3]);
            } catch (Exception e) {
                  e.printStackTrace();
            }
            texts = new TextView[5];
            texts[0] = (TextView) findViewById(R.id.textViewHighScores);
            texts[1] = (TextView) findViewById(R.id.textViewEasy);
            texts[2] = (TextView) findViewById(R.id.textViewMeduim);
            texts[3] = (TextView) findViewById(R.id.textViewHard);
            texts[4] = (TextView) findViewById(R.id.textViewChallenge);
            startUpFont();
      }

      @Override
      public void onBackPressed() {
            super.onBackPressed();
            finish();
      }

      public void startUpFont(){
            Typeface crayon_crumble = Typeface.createFromAsset(getAssets(),
"fonts/dk_crayon_crumble.ttf");
            bDelete.setTypeface(crayon_crumble);     // set the font for all the buttons
to crayon_crumble
            bDelete.setTextColor(Color.WHITE);
            textScoreEasy.setTypeface(crayon_crumble);
            textScoreEasy.setTextColor(Color.WHITE);
            textScoreMeduim.setTypeface(crayon_crumble);
            textScoreMeduim.setTextColor(Color.WHITE);
            textScoreHard.setTypeface(crayon_crumble);
            textScoreHard.setTextColor(Color.WHITE);
            textScoreChallenge.setTypeface(crayon_crumble);
            textScoreChallenge.setTextColor(Color.WHITE);
            for (int i = 0; i < texts.length; i++) {
                  texts[i].setTypeface(crayon_crumble);
                  texts[i].setTextColor(Color.WHITE);
            }
      }

      public void startUpButtonDelete() {
            bDelete = (Button) findViewById(R.id.buttonDelete);
            bDelete.setOnClickListener(new View.OnClickListener() {
                  public void onClick(View v) {
                        deletePopUp();
                  }
            });
      }

      public void deletePopUp(){
                  Builder dialogbox = new AlertDialog.Builder(this);
                  dialogbox.setCancelable(false);
                  dialogbox.setMessage("Are you sure you want to delete the high
scores?");
                  dialogbox.setNegativeButton("no", new DialogInterface.OnClickListener()
{

                        @Override
                        public void onClick(DialogInterface dialog, int which) {
                        }
                  });
                  dialogbox.setPositiveButton("yes", new
DialogInterface.OnClickListener() {

                        @Override
                        public void onClick(DialogInterface dialog, int which) {
                              File dir = getFilesDir();
                              File file = new File(dir, "scores");
                              file.delete();
                              Intent restart = getIntent();
                              startActivity(restart);
                              finish();
                         }
                    });
                    dialogbox.show();
          }
}
```

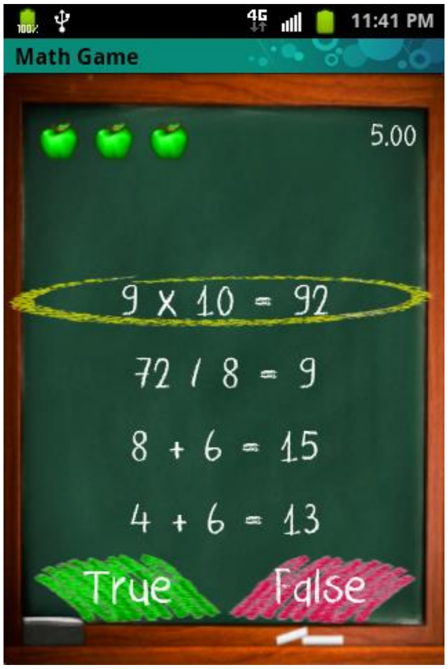
Song.java

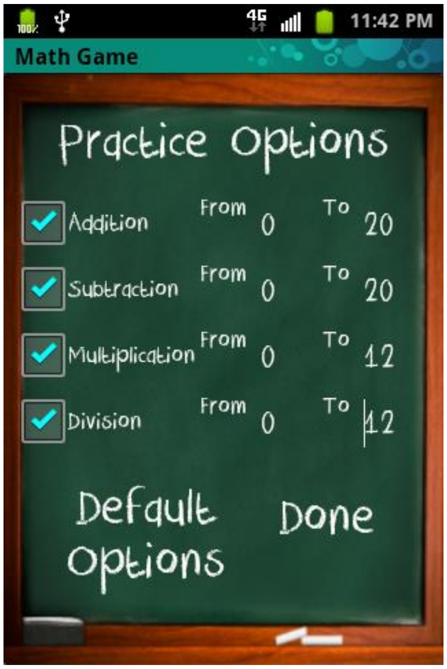
```java
package com.example.mathgame;

import android.content.Context;
import android.media.MediaPlayer;

public class Song implements Runnable{

	private static MediaPlayer music;
	private static MediaPlayer victory;

	public Song(Context context){
		music = MediaPlayer.create(context, R.raw.happy_instrumental);
		victory = MediaPlayer.create(context, R.raw.flawless_victory);
	}

	@Override
	public void run() {
		music.start();
		music.setLooping(true);
	}

	public static void stop(){
		music.stop();
	}

	public static void pause(){
		music.pause();
	}

	public static void start(){
		music.start();
	}

	public static void startVictory(){
		victory.setVolume(25, 25);
		victory.start();
	}
}
```

Finished Images









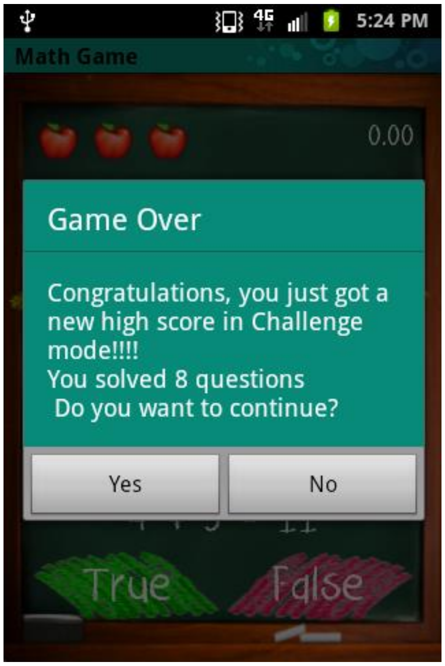
Practice Options

☑ Addition From 0 To 20

☑ Subtraction From 0 To 20

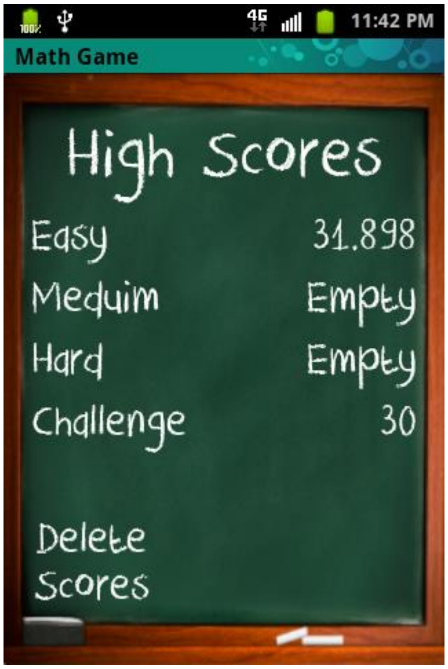
☑ Multiplication From 0 To 12

☑ Division From 0 To 12



Game Over

Congratulations, you just got a new high score in Challenge mode!!!!
You solved 8 questions
Do you want to continue?



High Scores

Easy	31.898
Meduim	Empty
Hard	Empty
Challenge	30

Total Programming Project Hours

Week	Hours	Members	Total Hours:	132
Week 9	7	JV, JC, FA		
Week 10	10	JV, JC, FA		
Week 11	9	JV, JC, FA		
Week 12	5	JV, JC, FA		
Week 7	9	JV		
Week 9	6	JV		
Week 10	7	JV		
Week 11	11	JV		
Week 12	3	JV		
Week 9	10	JC		
Week 10	15	JC		
Week 11	12	JC		
Week 12	4	JC		
Week 9	12	FA		
Week 10	7	FA		
Week 12	5	FA		