A picture containing text, sign

Description automatically generatedA picture containing logo

Description automatically generated*Republic of the Philippines*

**EULOGIO “AMANG” RODRIGUEZ**

**INSTITUTE OF SCIENCE AND TECHNOLOGY**

*Nagtahan, Sampaloc, Manila*

COLLEGE OF ARTS AND SCIENCES - ITE DEPARTMENT

**< DATE >**

**Name <Signature/Date>**

**7-32-2021 Marvin Terence Heroda**

**Final Project in Application Development and Emerging Technologies Title:**

**Sudoku Math Games**

**Rationale: (Background of the final project understudy)**

Sudoku is a puzzle with simple, easy to understand rules that any kid can try. The need to engage in logical thinking to fill the grid correctly plus the process of trial and error they must apply will naturally and unconsciously help to develop their problem-solving skills.

Furthermore, the challenge of solving what to a kid will seem like an easy and boring game also helps them to engage more intensely to finish it quickly and improves their concentration skills.

These benefits of Sudoku can also help them in other areas and even improve their school performance.

**Area of Investigation: (Refer to enlisted lists of topics)**

The investigation covers the utilized of the current framework however automating it to effortlessly check the participation of the understudies and educators. To make it simpler to distinguish their records and punishments. The proposed framework will be equipped for keeping up the proficiency of each understudy participation record through Number input. The proposed framework additionally has the ability of figuring punishment of understudy to begiven contingent upon his participation as respects to his/her nonappearances per occasion.

**Features of the application development: (min. of 5 in bulleted format)**

* **NEW GAME**
* **CONTINUE GAME**
* **ABOUT**
* **EXIT**

**Technologies: (min. of 3 in bulleted format)**

**<Frontend/ Backend>**

* **JAVA**
* **XML**
* **AIDE**

**User’s Manual**

**XML**

**Activity\_Main.xml**

*<?xml version="1.0" encoding="utf-8"?>*

*<LinearLayout*

*xmlns:android="http://schemas.android.com/apk/res/android"*

*android:layout\_height="fill\_parent"*

*android:layout\_width="fill\_parent"*

*android:orientation="horizontal"*

*android:padding="85dip"*

*android:background="@drawable/sudoku\_1"*

*>*

*<LinearLayout*

*android:orientation="vertical"*

*android:layout\_height="wrap\_content"*

*android:layout\_width="fill\_parent"*

*android:layout\_gravity="center"*

*>*

*<TextView*

*android:layout\_height="wrap\_content"*

*android:layout\_width="wrap\_content"*

*android:layout\_gravity="center"*

*android:layout\_marginBottom="25dip"*

*android:textSize="24.5sp"*

*/>*

*<Button*

*android:id="@+id/continue\_button"*

*android:layout\_width="wrap\_content"*

*android:layout\_height="wrap\_content"*

*android:text="@string/continue\_label"*

*android:layout\_margin="5dip"*

*android:background="@drawable/btn"*

*/>*

*<Button*

*android:id="@+id/new\_button"*

*android:layout\_width="wrap\_content"*

*android:layout\_height="wrap\_content"*

*android:text="@string/new\_game\_label"*

*android:layout\_margin="5dip"*

*android:background="@drawable/btn"*

*/>*

*<Button*

*android:id="@+id/about\_button"*

*android:layout\_width="wrap\_content"*

*android:layout\_height="wrap\_content"*

*android:text="@string/about\_label"*

*android:layout\_margin="5dip"*

*android:background="@drawable/btn"*

*/>*

*<Button*

*android:id="@+id/exit\_button"*

*android:layout\_width="wrap\_content"*

*android:layout\_height="wrap\_content"*

*android:text="@string/exit\_label"*

*android:background="@drawable/btn"*

*android:layout\_margin="5dip"*

*/>*

*</LinearLayout>*

*</LinearLayout>*

**Main1.xml**

**<?xml version="1.0" encoding="utf-8"?>**

**<!--**

**! Excerpted from "Hello, Android!",**

**! published by The Pragmatic Bookshelf.**

**! Copyrights apply to this code. It may not be used to create training material,**

**! courses, books, articles, and the like. Contact us if you are in doubt.**

**! We make no guarantees that this code is fit for any purpose.**

**! Visit http://www.pragmaticprogrammer.com/titles/eband for more book information.**

**-->**

**<LinearLayout**

**xmlns:android="http://schemas.android.com/apk/res/android"**

**android:orientation="vertical"**

**android:layout\_width="fill\_parent"**

**android:layout\_height="fill\_parent"**

**android:background="#ffffff">**

**<TextView**

**android:layout\_width="fill\_parent"**

**android:layout\_height="wrap\_content"**

**android:text="@string/main\_title" />**

**<Button**

**android:layout\_width="fill\_parent"**

**android:layout\_height="wrap\_content"**

**android:text="@string/continue\_label" />**

**<Button**

**android:layout\_width="fill\_parent"**

**android:layout\_height="wrap\_content"**

**android:text="@string/new\_game\_label" />**

**<Button**

**android:layout\_width="fill\_parent"**

**android:layout\_height="wrap\_content"**

**android:text="@string/about\_label" />**

**<Button**

**android:layout\_width="fill\_parent"**

**android:layout\_height="wrap\_content"**

**android:text="@string/exit\_label" />**

**</LinearLayout>**

**About.xml**

**<?xml version="1.0" encoding="utf-8"?>**

**<!--**

**! Excerpted from "Hello, Android!",**

**! published by The Pragmatic Bookshelf.**

**! Copyrights apply to this code. It may not be used to create training material,**

**! courses, books, articles, and the like. Contact us if you are in doubt.**

**! We make no guarantees that this code is fit for any purpose.**

**! Visit http://www.pragmaticprogrammer.com/titles/eband for more book information.**

**-->**

**<ScrollView**

**xmlns:android="http://schemas.android.com/apk/res/android"**

**android:layout\_width="fill\_parent"**

**android:layout\_height="fill\_parent"**

**android:padding="10dip">**

**<TextView**

**android:id="@+id/about\_content"**

**android:layout\_width="wrap\_content"**

**android:layout\_height="wrap\_content"**

**android:text="@string/about\_text" />**

**</ScrollView>**

**Keypad.xml**

<?xml version="1.0" encoding="utf-8"?>

<!--

! Excerpted from "Hello, Android!",

! published by The Pragmatic Bookshelf.

! Copyrights apply to this code. It may not be used to create training material,

! courses, books, articles, and the like. Contact us if you are in doubt.

! We make no guarantees that this code is fit for any purpose.

! Visit http://www.pragmaticprogrammer.com/titles/eband for more book information.

-->

<TableLayout

xmlns:android="http://schemas.android.com/apk/res/android"

android:id="@+id/keypad"

android:orientation="vertical"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:stretchColumns="\*">

<TableRow>

<Button android:id="@+id/keypad\_1"

android:text="1">

</Button>

<Button android:id="@+id/keypad\_2"

android:text="2">

</Button>

<Button android:id="@+id/keypad\_3"

android:text="3">

</Button>

</TableRow>

<TableRow>

<Button android:id="@+id/keypad\_4"

android:text="4">

</Button>

<Button android:id="@+id/keypad\_5"

android:text="5">

</Button>

<Button android:id="@+id/keypad\_6"

android:text="6">

</Button>

</TableRow>

<TableRow>

<Button android:id="@+id/keypad\_7"

android:text="7">

</Button>

<Button android:id="@+id/keypad\_8"

android:text="8">

</Button>

<Button android:id="@+id/keypad\_9"

android:text="9">

</Button>

</TableRow>

</TableLayout>

**SodukoGame1.java**

package org.ace.game;

import android.app.Activity;

import android.app.AlertDialog;

import android.content.DialogInterface;

import android.content.Intent;

import android.os.Bundle;

import android.util.Log;

import android.view.Menu;

import android.view.MenuInflater;

import android.view.MenuItem;

import android.view.View;

import android.view.View.OnClickListener;

public class SudokuGame1 extends Activity implements OnClickListener {

private static final String TAG = "Sukgludflkgfdkhd";

/\*\* Called when the activity is first created. \*/

@Override

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.main);

// Set up click listeners for all the buttons

View continueButton = findViewById(R.id.continue\_button);

continueButton.setOnClickListener(this);

View newButton = findViewById(R.id.new\_button);

newButton.setOnClickListener(this);

View aboutButton = findViewById(R.id.about\_button);

aboutButton.setOnClickListener(this);

View exitButton = findViewById(R.id.exit\_button);

exitButton.setOnClickListener(this);

}

// ...

public void onClick(View v) {

switch (v.getId()) {

case R.id.about\_button:

Intent i = new Intent(this, About.class);

startActivity(i);

break;

// More buttons go here (if any) ...

case R.id.new\_button:

openNewGameDialog();

break;

case R.id.exit\_button:

finish();

break;

}

}

@Override

public boolean onCreateOptionsMenu(Menu menu) {

super.onCreateOptionsMenu(menu);

MenuInflater inflater = getMenuInflater();

inflater.inflate(R.menu.menu, menu);

return true;

}

@Override

public boolean onOptionsItemSelected(MenuItem item) {

switch (item.getItemId()) {

case R.id.settings:

startActivity(new Intent(this, Prefs.class));

return true;

// More items go here (if any) ...

}

return false;

}

/\*\* Ask the user what difficulty level they want \*/

private void openNewGameDialog() {

new AlertDialog.Builder(this)

.setTitle(R.string.new\_game\_title)

.setItems(R.array.difficulty,

new DialogInterface.OnClickListener() {

public void onClick(DialogInterface dialoginterface,

int i) {

startGame(i);

}

})

.show();

}

/\*\* Start a new game with the given difficulty level \*/

private void startGame(int i) {

Log.d(TAG, "clicked on " + i);

Intent intent = new Intent(SudokuGame1.this, Game.class);

intent.putExtra(Game.KEY\_DIFFICULTY, i);

startActivity(intent);

}

}

**PuzzleView1.java**

package org.ace.game;

import android.content.Context;

import android.graphics.Canvas;

import android.graphics.Paint;

import android.graphics.Rect;

import android.graphics.Paint.FontMetrics;

import android.graphics.Paint.Style;

import android.util.Log;

import android.view.KeyEvent;

import android.view.MotionEvent;

import android.view.View;

import android.view.animation.AnimationUtils;

public class PuzzleView extends View

{

private static final String TAG = "Sudoku";

private float width; // width of one tile

private float height; // height of one tile

private int selX; // X index of selection

private int selY; // Y index of selection

private final Rect selRect = new Rect();

private final Game game;

public PuzzleView(Context context)

{

super(context);

this.game = (Game) context;

setFocusable(true);

setFocusableInTouchMode(true);

}

protected void onSizeChanged(int w, int h, int oldw, int oldh)

{

width = w / 9f;

height = h / 9f;

getRect(selX, selY, selRect);

Log.d(TAG, "onSizeChanged: width " + width + ", height "

+ height);

super.onSizeChanged(w, h, oldw, oldh);

}

protected void onDraw(Canvas canvas)

{

// Draw the background...

Paint background = new Paint();

background.setColor(getResources().getColor(

R.color.puzzle\_background));

canvas.drawRect(0, 0, getWidth(), getHeight(), background);

// Draw the board...

// Define colors for the grid lines

Paint dark = new Paint();

dark.setColor(getResources().getColor(R.color.puzzle\_dark));

Paint hilite = new Paint();

hilite.setColor(getResources().getColor(R.color.puzzle\_hilite));

Paint light = new Paint();

light.setColor(getResources().getColor(R.color.puzzle\_light));

// Draw the minor grid lines

for (int i = 0; i < 9; i++) {

canvas.drawLine(0, i \* height, getWidth(), i \* height,

light);

canvas.drawLine(0, i \* height + 1, getWidth(), i \* height

+ 1, hilite);

canvas.drawLine(i \* width, 0, i \* width, getHeight(),

light);

canvas.drawLine(i \* width + 1, 0, i \* width + 1,

getHeight(), hilite);

}

// Draw the major grid lines

for (int i = 0; i < 9; i++)

{

if (i % 3 != 0)

continue;

canvas.drawLine(0, i \* height, getWidth(), i \* height,

dark);

canvas.drawLine(0, i \* height + 1, getWidth(), i \* height

+ 1, hilite);

canvas.drawLine(i \* width, 0, i \* width, getHeight(), dark);

canvas.drawLine(i \* width + 1, 0, i \* width + 1,

getHeight(), hilite);

}

// Draw the numbers...

// Define color and style for numbers

Paint foreground = new Paint(Paint.ANTI\_ALIAS\_FLAG);

foreground.setColor(getResources().getColor(

R.color.puzzle\_foreground));

foreground.setStyle(Style.FILL);

foreground.setTextSize(height \* 0.75f);

foreground.setTextScaleX(width / height);

foreground.setTextAlign(Paint.Align.CENTER);

// Draw the number in the center of the tile

FontMetrics fm = foreground.getFontMetrics();

// Centering in X: use alignment (and X at midpoint)

float x = width / 2;

// Centering in Y: measure ascent/descent first

float y = height / 2 - (fm.ascent + fm.descent) / 2;

for (int i = 0; i < 9; i++) {

for (int j = 0; j < 9; j++) {

canvas.drawText(this.game.getTileString(i, j), i

\* width + x, j \* height + y, foreground);

}

}

// Draw the hints...

// Pick a hint color based on #moves left

Paint hint = new Paint();

int c[] = { getResources().getColor(R.color.puzzle\_hint\_0),

getResources().getColor(R.color.puzzle\_hint\_1),

getResources().getColor(R.color.puzzle\_hint\_2), };

Rect r = new Rect();

for (int i = 0; i < 9; i++) {

for (int j = 0; j < 9; j++) {

int movesleft = 9 - game.getUsedTiles(i, j).length;

if (movesleft < c.length) {

getRect(i, j, r);

hint.setColor(c[movesleft]);

canvas.drawRect(r, hint);

}

}

}

// Draw the selection...

Log.d(TAG, "selRect=" + selRect);

Paint selected = new Paint();

selected.setColor(getResources().getColor(

R.color.puzzle\_selected));

canvas.drawRect(selRect, selected);

}

public boolean onTouchEvent(MotionEvent event) {

if (event.getAction() != MotionEvent.ACTION\_DOWN)

return super.onTouchEvent(event);

select((int) (event.getX() / width),

(int) (event.getY() / height));

game.showKeypadOrError(selX, selY);

Log.d(TAG, "onTouchEvent: x " + selX + ", y " + selY);

return true;

}

public boolean onKeyDown(int keyCode, KeyEvent event) {

Log.d(TAG, "onKeyDown: keycode=" + keyCode + ", event="

+ event);

switch (keyCode) {

case KeyEvent.KEYCODE\_DPAD\_UP:

select(selX, selY - 1);

break;

case KeyEvent.KEYCODE\_DPAD\_DOWN:

select(selX, selY + 1);

break;

case KeyEvent.KEYCODE\_DPAD\_LEFT:

select(selX - 1, selY);

break;

case KeyEvent.KEYCODE\_DPAD\_RIGHT:

select(selX + 1, selY);

break;

case KeyEvent.KEYCODE\_0:

case KeyEvent.KEYCODE\_SPACE: setSelectedTile(0); break;

case KeyEvent.KEYCODE\_1: setSelectedTile(1); break;

case KeyEvent.KEYCODE\_2: setSelectedTile(2); break;

case KeyEvent.KEYCODE\_3: setSelectedTile(3); break;

case KeyEvent.KEYCODE\_4: setSelectedTile(4); break;

case KeyEvent.KEYCODE\_5: setSelectedTile(5); break;

case KeyEvent.KEYCODE\_6: setSelectedTile(6); break;

case KeyEvent.KEYCODE\_7: setSelectedTile(7); break;

case KeyEvent.KEYCODE\_8: setSelectedTile(8); break;

case KeyEvent.KEYCODE\_9: setSelectedTile(9); break;

case KeyEvent.KEYCODE\_ENTER:

case KeyEvent.KEYCODE\_DPAD\_CENTER:

game.showKeypadOrError(selX, selY);

break;

default:

return super.onKeyDown(keyCode, event);

}

return true;

}

public void setSelectedTile(int tile) {

if (game.setTileIfValid(selX, selY, tile)) {

invalidate();// may change hints

} else {

// Number is not valid for this tile

Log.d(TAG, "setSelectedTile: invalid: " + tile);

startAnimation(AnimationUtils.loadAnimation(game,

R.anim.shake));

}

}

private void select(int x, int y) {

invalidate(selRect);

selX = Math.min(Math.max(x, 0), 8);

selY = Math.min(Math.max(y, 0), 8);

getRect(selX, selY, selRect);

invalidate(selRect);

}

private void getRect(int x, int y, Rect rect) {

rect.set((int) (x \* width), (int) (y \* height), (int) (x

\* width + width), (int) (y \* height + height));

}

}

**Prefs.java**

package org.ace.game;

import android.os.Bundle;

import android.preference.PreferenceActivity;

public class Prefs extends PreferenceActivity {

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

addPreferencesFromResource(R.xml.settings);

}

}

**Keypads.java**

package org.ace.game;

import android.app.Dialog;

import android.content.Context;

import android.os.Bundle;

import android.view.KeyEvent;

import android.view.View;

public class Keypad extends Dialog

{

protected static final String TAG = "Sudoku";

private final View keys[] = new View[9];

private View keypad;

private final int useds[];

private final PuzzleView puzzleView;

public Keypad(Context context, int useds[], PuzzleView puzzleView)

{

super(context);

this.useds = useds;

this.puzzleView = puzzleView;

}

protected void onCreate(Bundle savedInstanceState)

{

super.onCreate(savedInstanceState);

setTitle(R.string.keypad\_title);

setContentView(R.layout.keypad);

findViews();for (int element : useds)

{

if (element != 0)

keys[element - 1].setVisibility(View.INVISIBLE);

}

setListeners();

}

public boolean onKeyDown(int keyCode, KeyEvent event)

{

int tile = 0;

switch (keyCode)

{

case KeyEvent.KEYCODE\_0:

case KeyEvent.KEYCODE\_SPACE: tile = 0; break;

case KeyEvent.KEYCODE\_1: tile = 1; break;

case KeyEvent.KEYCODE\_2: tile = 2; break;

case KeyEvent.KEYCODE\_3: tile = 3; break;

case KeyEvent.KEYCODE\_4: tile = 4; break;

case KeyEvent.KEYCODE\_5: tile = 5; break;

case KeyEvent.KEYCODE\_6: tile = 6; break;

case KeyEvent.KEYCODE\_7: tile = 7; break;

case KeyEvent.KEYCODE\_8: tile = 8; break;

case KeyEvent.KEYCODE\_9: tile = 9; break;

default:

return super.onKeyDown(keyCode, event);

}

if (isValid(tile))

{

returnResult(tile);

}

return true;

}

private void returnResult(int tile)

{

puzzleView.setSelectedTile(tile);

dismiss();

}

private boolean isValid(int tile)

{

for (int t : useds)

{

if (tile == t)

return false;

}

return true;

}

private void findViews()

{

keypad = findViewById(R.id.keypad);

keys[0] = findViewById(R.id.keypad\_1);

keys[1] = findViewById(R.id.keypad\_2);

keys[2] = findViewById(R.id.keypad\_3);

keys[3] = findViewById(R.id.keypad\_4);

keys[4] = findViewById(R.id.keypad\_5);

keys[5] = findViewById(R.id.keypad\_6);

keys[6] = findViewById(R.id.keypad\_7);

keys[7] = findViewById(R.id.keypad\_8);

keys[8] = findViewById(R.id.keypad\_9);

}

private void setListeners()

{

for (int i = 0; i < keys.length; i++)

{

final int t = i + 1;

keys[i].setOnClickListener(new View.OnClickListener()

{

public void onClick(View v)

{

returnResult(t);

}

});

}

keypad.setOnClickListener(new View.OnClickListener()

{

public void onClick(View v)

{

returnResult(0);

}

});

}

}

**Game.java**

package org.ace.game;

import android.app.Activity;

import android.app.Dialog;

import android.os.Bundle;

import android.util.Log;

import android.view.Gravity;

import android.widget.Toast;

public class Game extends Activity {

private static final String TAG = "Sudoku";

public static final String KEY\_DIFFICULTY =

"org.example.sudoku.difficulty";

public static final int DIFFICULTY\_EASY = 0;

public static final int DIFFICULTY\_MEDIUM = 1;

public static final int DIFFICULTY\_HARD = 2;

private int puzzle[] = new int[9 \* 9];

private final String easyPuzzle =

"360000000004230800000004200" +

"070460003820000014500013020" +

"001900000007048300000000045";

private final String mediumPuzzle =

"650000070000506000014000005" +

"007009000002314700000700800" +

"500000630000201000030000097";

private final String hardPuzzle =

"009000000080605020501078000" +

"000000700706040102004000000" +

"000720903090301080000000600";

private PuzzleView puzzleView;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

Log.d(TAG, "onCreate");

int diff = getIntent().getIntExtra(KEY\_DIFFICULTY,

DIFFICULTY\_EASY);

puzzle = getPuzzle(diff);

calculateUsedTiles();

puzzleView = new PuzzleView(this);

setContentView(puzzleView);

puzzleView.requestFocus();

}

// ...

/\*\* Given a difficulty level, come up with a new puzzle \*/

private int[] getPuzzle(int diff) {

String puz;

// TODO: Continue last game

switch (diff) {

case DIFFICULTY\_HARD:

puz = hardPuzzle;

break;

case DIFFICULTY\_MEDIUM:

puz = mediumPuzzle;

break;

case DIFFICULTY\_EASY:

default:

puz = easyPuzzle;

break;

}

return fromPuzzleString(puz);

}

/\*\* Convert an array into a puzzle string \*/

static private String toPuzzleString(int[] puz) {

StringBuilder buf = new StringBuilder();

for (int element : puz) {

buf.append(element);

}

return buf.toString();

}

/\*\* Convert a puzzle string into an array \*/

static protected int[] fromPuzzleString(String string) {

int[] puz = new int[string.length()];

for (int i = 0; i < puz.length; i++) {

puz[i] = string.charAt(i) - '0';

}

return puz;

}

/\*\* Return the tile at the given coordinates \*/

private int getTile(int x, int y) {

return puzzle[y \* 9 + x];

}

/\*\* Change the tile at the given coordinates \*/

private void setTile(int x, int y, int value) {

puzzle[y \* 9 + x] = value;

}

/\*\* Return a string for the tile at the given coordinates \*/

protected String getTileString(int x, int y) {

int v = getTile(x, y);

if (v == 0)

return "";

else

return String.valueOf(v);

}

/\*\* Change the tile only if it's a valid move \*/

protected boolean setTileIfValid(int x, int y, int value) {

int tiles[] = getUsedTiles(x, y);

if (value != 0) {

for (int tile : tiles) {

if (tile == value)

return false;

}

}

setTile(x, y, value);

calculateUsedTiles();

return true;

}

/\*\* Open the keypad if there are any valid moves \*/

protected void showKeypadOrError(int x, int y) {

int tiles[] = getUsedTiles(x, y);

if (tiles.length == 9) {

Toast toast = Toast.makeText(this,

R.string.no\_moves\_label, Toast.LENGTH\_SHORT);

toast.setGravity(Gravity.CENTER, 0, 0);

toast.show();

} else {

Log.d(TAG, "showKeypad: used=" + toPuzzleString(tiles));

Dialog v = new Keypad(this, tiles, puzzleView);

v.show();

}

}

/\*\* Cache of used tiles \*/

private final int used[][][] = new int[9][9][];

/\*\* Return cached used tiles visible from the given coords \*/

protected int[] getUsedTiles(int x, int y) {

return used[x][y];

}

/\*\* Compute the two dimensional array of used tiles \*/

private void calculateUsedTiles() {

for (int x = 0; x < 9; x++) {

for (int y = 0; y < 9; y++) {

used[x][y] = calculateUsedTiles(x, y);

// Log.d(TAG, "used[" + x + "][" + y + "] = "

// + toPuzzleString(used[x][y]));

}

}

}

/\*\* Compute the used tiles visible from this position \*/

private int[] calculateUsedTiles(int x, int y) {

int c[] = new int[9];

// horizontal

for (int i = 0; i < 9; i++) {

if (i == y)

continue;

int t = getTile(x, i);

if (t != 0)

c[t - 1] = t;

}

// vertical

for (int i = 0; i < 9; i++) {

if (i == x)

continue;

int t = getTile(i, y);

if (t != 0)

c[t - 1] = t;

}

// same cell block

int startx = (x / 3) \* 3;

int starty = (y / 3) \* 3;

for (int i = startx; i < startx + 3; i++) {

for (int j = starty; j < starty + 3; j++) {

if (i == x && j == y)

continue;

int t = getTile(i, j);

if (t != 0)

c[t - 1] = t;

}

}

// compress

int nused = 0;

for (int t : c) {

if (t != 0)

nused++;

}

int c1[] = new int[nused];

nused = 0;

for (int t : c) {

if (t != 0)

c1[nused++] = t;

}

return c1;

}

}

**About.java**

package org.ace.game;

import android.app.Activity;

import android.os.Bundle;

public class About extends Activity {

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.about);

}

}

***OUTPUT***

