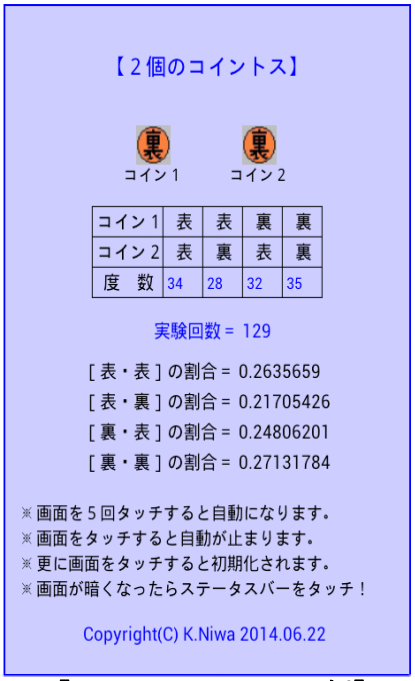
```
//---------------------------------
//
//          ２個のコイントス
//               Ver7
//   Copyright(C) K.Niwa 2014.06.22
//
//---------------------------------
```

【エミュレータ画面例】



【スマートフォン画面例】
（Android 4.0）

［アプリの概要］

２個のコインを同時に投げたときの表裏の出方が観察できます。
２個とも表である割合はどれだけでしょうか。
２個のコインを同時に投げる実験を何度もおこなった場合、実験回数を多くすると、２個とも表である割合は、実験回数が少ないときに比べてどのようなことが言えるのでしょうか。
また、２個とも表である確率、２個とも裏である確率、１個が表で１個が裏である確率をそれぞれ数学的に求めてみましょう。

［１］Cointosu.java

```
package jp.seitoku.cointosu;

import android.content.Context;
import android.content.res.Resources;      //画像用
import android.graphics.*;
import android.util.AttributeSet;
import android.view.*;

public class Cointosu extends View {

        private Bitmap bitmap1 = null;
        private Bitmap bitmap2 = null;

        int ct=0;                                          //実験回数カウンター
        int r1,r2;                                         //コイン１、コイン２　のそれぞれの表裏の識別子(乱
数)
        int d1=0,d2=0,d3=0,d4=0;  //[表表][表裏][裏表][裏裏]のカウンター
        int flag=0;                                        //自動識別子
        int syoki=0;                                  //初期化識別子

        int width;
        int height;
```

```
	public Cointosu(Context context, AttributeSet attrs, int defStyle) {
		super(context, attrs, defStyle);
		// TODO 自動生成されたコンストラクター・スタブ
		init(context);
	}

	private void init(Context context) {
		// TODO 自動生成されたメソッド・スタブ
		Resources res = context.getResources();
		bitmap1 = BitmapFactory.decodeResource(res, R.drawable.coin1);
		bitmap2 = BitmapFactory.decodeResource(res, R.drawable.coin2);

		WindowManager wm = (WindowManager)context.getSystemService(Context.WINDOW_SERVICE);
		Display disp = wm.getDefaultDisplay();
		width = disp.getWidth();
		height = disp.getHeight();
	}


	public Cointosu(Context context, AttributeSet attrs) {
		super(context, attrs);
		// TODO 自動生成されたコンストラクター・スタブ
		init(context);
	}

	public Cointosu(Context context) {
		super(context);
		// TODO 自動生成されたコンストラクター・スタブ
		init(context);
	}

	@Override
	protected void onDraw(Canvas canvas) {
		// TODO 自動生成されたメソッド・スタブ

		float a=0;
		float b=0;

		super.onDraw(canvas);
		canvas.drawColor(Color.WHITE);
		Paint paint = new Paint();
		paint.setColor(Color.BLUE);
		paint.setAlpha(50);
		canvas.drawRect((getWidth()/2-240)+10,(getHeight()/2-343)+10,(getWidth()/2-240)+470,(getH
eight()/2-343)+675, paint);

		paint.setAlpha(10000);
		paint.setColor(Color.BLUE);

		for (int i=0;i<2;i++) {
			canvas.drawLine((getWidth()/2-240)+10+i,(getHeight()/2-343)+10+i,(getWidth()/2-24
0)+10+i,(getHeight()/2-343)+675-i, paint);
			canvas.drawLine((getWidth()/2-240)+10+i,(getHeight()/2-343)+675-i,(getWidth()/2-2
40)+470-i,(getHeight()/2-343)+675-i, paint);
			canvas.drawLine((getWidth()/2-240)+470-i,(getHeight()/2-343)+675-i,(getWidth()/2-
240)+470-i,(getHeight()/2-343)+10+i, paint);
			canvas.drawLine((getWidth()/2-240)+470-i,(getHeight()/2-343)+10+i,(getWidth()/2-2
40)+10+i,(getHeight()/2-343)+10+i, paint);
		}

		paint.setColor(Color.BLUE);
		paint.setTextSize(23.0f);
		canvas.drawText("【２個のコイントス】", (getWidth()/2-240)+145-20, (getHeight()/2-343)+80,
 paint);

		paint.setColor(Color.BLACK);
		paint.setTextSize(17.0f);
```

```
                canvas.drawText("コイン１", (getWidth()/2-240)+145, (getHeight()/2-343)+185, paint);
                canvas.drawText("コイン２", (getWidth()/2-240)+265, (getHeight()/2-343)+185, paint);

                if (CointosuActivity.ritsu != 0) {
                        a=(float)0.9*320/CointosuActivity.ritsu;  //---------------------------- ＜画像の
拡大・縮小の横の倍率を指定する＞
                        b=(float)0.9*320/CointosuActivity.ritsu;  //---------------------------- ＜画像
の拡大・縮小の縦の倍率を指定する＞
                }
                else {
                        a=(float) 1.0;
                        b=(float) 1.0;
                }

                Matrix Mat = new Matrix();        //----------------------***
                Mat.postScale(a, b);      //-------------------------***
                Bitmap bitmap11 = Bitmap.createBitmap(     //---------***
                                bitmap1,0,0,      //--------------------------***
                                bitmap1.getWidth(),       //----------------------***
                                bitmap1.getHeight(),      //------------------***
                                Mat,true //-----------------------------***
                                );       //-------------------------------------***


                //Matrix Mat = new Matrix();      //----------------------***
                //Mat.postScale(a, b);    //-------------------------***
                Bitmap bitmap22 = Bitmap.createBitmap(     //---------***
                                bitmap2,0,0,      //--------------------------***
                                bitmap2.getWidth(),       //----------------------***
                                bitmap2.getHeight(),      //------------------***
                                Mat,true //-----------------------------***
                                );       //-------------------------------------***


                if (bitmap11 != null && bitmap22 != null) {
                        ct++;

                        r1=(int) (1+2*Math.random());
                        if (r1==1) {
                                canvas.drawBitmap(bitmap11, (getWidth()/2-240)+160,(getHeight()/2-343)+1
30, paint);
                        }
                        else if (r1==2) {
                                canvas.drawBitmap(bitmap22, (getWidth()/2-240)+160,(getHeight()/2-343)+1
30, paint);
                        }

                        r2=(int) (1+2*Math.random());
                        if (r2==1) {
                                canvas.drawBitmap(bitmap11, (getWidth()/2-240)+280,(getHeight()/2-343)+1
30, paint);
                        }
                        else if (r2==2) {
                                canvas.drawBitmap(bitmap22, (getWidth()/2-240)+280,(getHeight()/2-343)+1
30, paint);
                        }
                }

                if (r1==1 && r2==1) {
                        d1++;
                }
                else if (r1==1 && r2==2) {
                        d2++;
                }
                else if (r1==2 && r2==1) {
                        d3++;
                }
                else if (r1==2 && r2==2) {
```

```
                    d4++;
                }

                paint.setColor(Color.BLACK);
                canvas.drawLine((getWidth()/2-240)+110, (getHeight()/2-343)+210, (getWidth()/2-240)+370,
(getHeight()/2-343)+210, paint);
                canvas.drawLine((getWidth()/2-240)+110, (getHeight()/2-343)+240, (getWidth()/2-240)+370,
(getHeight()/2-343)+240, paint);
                canvas.drawLine((getWidth()/2-240)+110, (getHeight()/2-343)+270, (getWidth()/2-240)+370,
(getHeight()/2-343)+270, paint);
                canvas.drawLine((getWidth()/2-240)+110, (getHeight()/2-343)+300, (getWidth()/2-240)+370,
(getHeight()/2-343)+300, paint);
                canvas.drawLine((getWidth()/2-240)+110, (getHeight()/2-343)+210, (getWidth()/2-240)+110,
(getHeight()/2-343)+300, paint);
                canvas.drawLine((getWidth()/2-240)+190, (getHeight()/2-343)+210, (getWidth()/2-240)+190,
(getHeight()/2-343)+300, paint);
                canvas.drawLine((getWidth()/2-240)+235, (getHeight()/2-343)+210, (getWidth()/2-240)+235,
(getHeight()/2-343)+300, paint);
                canvas.drawLine((getWidth()/2-240)+280, (getHeight()/2-343)+210, (getWidth()/2-240)+280,
(getHeight()/2-343)+300, paint);
                canvas.drawLine((getWidth()/2-240)+325, (getHeight()/2-343)+210, (getWidth()/2-240)+325,
(getHeight()/2-343)+300, paint);
                canvas.drawLine((getWidth()/2-240)+370, (getHeight()/2-343)+210, (getWidth()/2-240)+370,
(getHeight()/2-343)+300, paint);

                paint.setColor(Color.BLACK);
                paint.setTextSize(19.0f);
                canvas.drawText("コイン１", (getWidth()/2-240)+115, (getHeight()/2-343)+232, paint);
                canvas.drawText("コイン２", (getWidth()/2-240)+115, (getHeight()/2-343)+262, paint);
                canvas.drawText("度 数", (getWidth()/2-240)+123, (getHeight()/2-343)+292, paint);

                canvas.drawText("表", (getWidth()/2-240)+205, (getHeight()/2-343)+232, paint);
                canvas.drawText("表", (getWidth()/2-240)+205, (getHeight()/2-343)+262, paint);
                canvas.drawText("表", (getWidth()/2-240)+250, (getHeight()/2-343)+232, paint);
                canvas.drawText("裏", (getWidth()/2-240)+250, (getHeight()/2-343)+262, paint);
                canvas.drawText("裏", (getWidth()/2-240)+295, (getHeight()/2-343)+232, paint);
                canvas.drawText("表", (getWidth()/2-240)+295, (getHeight()/2-343)+262, paint);
                canvas.drawText("裏", (getWidth()/2-240)+340, (getHeight()/2-343)+232, paint);
                canvas.drawText("裏", (getWidth()/2-240)+340, (getHeight()/2-343)+262, paint);

                paint.setColor(Color.BLUE);
                paint.setTextSize(16.0f);
                canvas.drawText(""+d1, (getWidth()/2-240)+195, (getHeight()/2-343)+292, paint);
                canvas.drawText(""+d2, (getWidth()/2-240)+240, (getHeight()/2-343)+292, paint);
                canvas.drawText(""+d3, (getWidth()/2-240)+285, (getHeight()/2-343)+292, paint);
                canvas.drawText(""+d4, (getWidth()/2-240)+330, (getHeight()/2-343)+292, paint);

                paint.setColor(Color.BLUE);
                paint.setTextSize(19.0f);
                canvas.drawText("実験回数＝ "+ct, (getWidth()/2-240)+180, (getHeight()/2-343)+340, paint)
;

                paint.setColor(Color.BLACK);
                canvas.drawText("［表・表］の割合＝ "+((float)(d1)/(float)(ct)), (getWidth()/2-240)+100,
(getHeight()/2-343)+380, paint);
                canvas.drawText("［表・裏］の割合＝ "+((float)(d2)/(float)(ct)), (getWidth()/2-240)+100,
(getHeight()/2-343)+410, paint);
                canvas.drawText("［裏・表］の割合＝ "+((float)(d3)/(float)(ct)), (getWidth()/2-240)+100,
(getHeight()/2-343)+440, paint);
                canvas.drawText("［裏・裏］の割合＝ "+((float)(d4)/(float)(ct)), (getWidth()/2-240)+100,
(getHeight()/2-343)+470, paint);

                paint.setColor(Color.BLACK);
                paint.setTextSize(18.0f);
                canvas.drawText("※ 画面を５回タッチすると自動になります。", (getWidth()/2-240)+50-20, (g
etHeight()/2-343)+520, paint);
                canvas.drawText("※ 画面をタッチすると自動が止まります。", (getWidth()/2-240)+50-20, (get
Height()/2-343)+545, paint);
```

```
                canvas.drawText("※ 更に画面をタッチすると初期化されます。", (getWidth()/2-240)+50-20, (getHeight()/2-343)+570, paint);
                canvas.drawText("※ 画面が暗くなったらステータスバーをタッチ！", (getWidth()/2-240)+50-20, (getHeight()/2-343)+595, paint);

                paint.setColor(Color.BLUE);
                paint.setTextSize(19.0f);
                canvas.drawText("Copyright(C) K.Niwa 2014.06.22", (getWidth()/2-240)+100, (getHeight()/2-343)+640, paint);

                if (flag >= 5) {
                        if (d1<1000 && d2<1000 && d3<1000 && d4<1000) {
                                invalidate();
                        }
                }

        }       //protected void onDraw(Canvas canvas)

        @Override
        public boolean onTouchEvent(MotionEvent event) {
                flag++;
                flag = flag % 6;

                syoki++;
                if (syoki > 6){
                        ct=0;
                        d1=0;d2=0;d3=0;d4=0;
                        flag=0;
                        syoki=0;
                }

                invalidate();
                return false;
        }

}
```

[２] main.xml

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

<LinearLayout
        xmlns:android="http://schemas.android.com/apk/res/android"
        android:orientation="vertical"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:weightSum="1" >

        <jp.seitoku.cointosu.Cointosu
                android:layout_height="match_parent"
                android:layout_width="match_parent"
                android:id="@+id/myview1">
        </jp.seitoku.cointosu.Cointosu>
</LinearLayout>
```

[３] CointosuActivity.java

```
package jp.seitoku.cointosu;

import android.os.Bundle;
import android.app.Activity;
import android.view.Menu;
```

```
import android.util.DisplayMetrics;        //-----------------------------〈画像の拡大・縮小に必要なライブラリ〉


public class CointosuActivity extends Activity {

    static int ritsu;

        @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);

        DisplayMetrics metrics = new DisplayMetrics();      //---------------------------〈端末の情報を取得する〉
                getWindowManager().getDefaultDisplay().getMetrics(metrics);//-----------------------------------
                StringBuilder buffer = new StringBuilder();         //-------------------------------------------
        buffer.append("densityDpi（ドット数／インチ）:" + String.valueOf(metrics.densityDpi) + "¥n");
//------
        ritsu=metrics.densityDpi;// ------------------------------------------------これで値が取り出せた!
        }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.main, menu);
        return true;
    }
}
```