

```
[1]MyTenencoinEng. java
```

```
/*
```

```
-----  
10円玉を投げて円周率 $\pi$ を求める (英語版)
```

```
Android 4.1 (Jelly Bean)
```

```
Copyright (C) K. Niwa 2021. 8. 28  
-----
```

```
*/
```

```
package jp.kiyo.wuena.mytenencoineng;
```

```
import android.content.Context;
```

```
import android.graphics.Canvas;
```

```
import android.graphics.Color;
```

```
import android.graphics.Paint;
```

```
import android.graphics.Rect;
```

```
import android.graphics.RectF;
```

```
import android.util.AttributeSet;
```

```
import android.view.MotionEvent;
```

```
import android.view.View;
```

```
public class MyTenencoinEng extends View { //View クラスを継承した MyMonte クラス
```

```
    int i; //変数宣言と初期化
```

```
    int sum=0; //格子点と重なったコインの個数
```

```
    int px; //コインの位置の x 座標を整数型にしたもの
```

```
    int py; //コインの位置の y 座標を整数型にしたもの
```

```
    int N=9999; //実験回数の最大値
```

```
    int flag=0; //コインを投げたか(1)、否か(0)、初期化する(2)の識別子
```

```
    double pai;
```

```
    int k=0; //投げたコインの個数
```

```
    double[] x=new double[10001];
```

```
    double[] y=new double[10001];
```

```
    public MyTenencoinEng(Context context, AttributeSet attrs, int defStyle) { //コンストラク
```

```
        タ
```

```

        super(context, attrs, defStyle);
    }

    public MyTenencoinEng(Context context, AttributeSet attrs) {    //コンストラクタ
        super(context, attrs);
    }

    public MyTenencoinEng(Context context) {    //コンストラクタ
        super(context);
    }

    //onDraw メソッド-----
    -----

    @Override
    protected void onDraw(Canvas canvas) {

        super.onDraw(canvas);
        canvas.drawColor(Color.WHYTE);
        Paint paint = new Paint();
        paint.setColor(Color.BLUE);
        paint.setAlpha(50);
        canvas.drawRect((getWidth()/2-360)+20, (getHeight()/2-600)+10, (getWidth()/2-
360)+700, (getHeight()/2-600)+1190, paint);
        paint.setAlpha(10000);
        paint.setColor(Color.BLUE);

        for (int i=0;i<2;i++) {
            canvas.drawLine((getWidth()/2-360)+20+i, (getHeight()/2-600)+10+i, (getWidth()/2-
360)+20+i, (getHeight()/2-600)+1190-i, paint);
            canvas.drawLine((getWidth()/2-360)+20+i, (getHeight()/2-600)+1190-i, (getWidth()/2-
360)+700-i, (getHeight()/2-600)+1190-i, paint);
            canvas.drawLine((getWidth()/2-360)+700-i, (getHeight()/2-600)+1190-i, (getWidth()/2-
360)+700-i, (getHeight()/2-600)+10+i, paint);
            canvas.drawLine((getWidth()/2-360)+700-i, (getHeight()/2-600)+10+i, (getWidth()/2-
360)+20+i, (getHeight()/2-600)+10+i, paint);
        }
    }

```

```

        paint.setColor (Color. BLACK) ;                                //格子の描画
        canvas.drawLine((getWidth()/2-360)+40+120, (getHeight()/2-600)+70, (getWidth()/2-
360)+40+120, (getHeight()/2-600)+530, paint) ;
        canvas.drawLine((getWidth()/2-360)+140+120, (getHeight()/2-600)+70, (getWidth()/2-
360)+140+120, (getHeight()/2-600)+530, paint) ;
        canvas.drawLine((getWidth()/2-360)+240+120, (getHeight()/2-600)+70, (getWidth()/2-
360)+240+120, (getHeight()/2-600)+530, paint) ;
        canvas.drawLine((getWidth()/2-360)+340+120, (getHeight()/2-600)+70, (getWidth()/2-
360)+340+120, (getHeight()/2-600)+530, paint) ;
        canvas.drawLine((getWidth()/2-360)+440+120, (getHeight()/2-600)+70, (getWidth()/2-
360)+440+120, (getHeight()/2-600)+530, paint) ;

        canvas.drawLine((getWidth()/2-360)+15+120, (getHeight()/2-600)+100, (getWidth()/2-
360)+465+120, (getHeight()/2-600)+100, paint) ;
        canvas.drawLine((getWidth()/2-360)+15+120, (getHeight()/2-600)+200, (getWidth()/2-
360)+465+120, (getHeight()/2-600)+200, paint) ;
        canvas.drawLine((getWidth()/2-360)+15+120, (getHeight()/2-600)+300, (getWidth()/2-
360)+465+120, (getHeight()/2-600)+300, paint) ;
        canvas.drawLine((getWidth()/2-360)+15+120, (getHeight()/2-600)+400, (getWidth()/2-
360)+465+120, (getHeight()/2-600)+400, paint) ;
        canvas.drawLine((getWidth()/2-360)+15+120, (getHeight()/2-600)+500, (getWidth()/2-
360)+465+120, (getHeight()/2-600)+500, paint) ;

        paint.setColor (Color. BLUE) ;                                //表題の表示
        paint.setTextSize(35.0f) ;
        canvas.drawText(" 【Throw Coins to Find the Pi】 ", (getWidth()/2-360)+50-20+30+63,
(getHeight()/2-600)+45+5, paint) ;

        paint.setColor (Color. BLACK) ;
        paint.setTextSize(30.0f) ;
        //if (k==0) {
        canvas.drawText(" Touch the screen to activate. ", (getWidth()/2-360)+50,
(getHeight()/2-600)+950, paint) ;                                //作者・作成年月の表示
        canvas.drawText(" Touch the screen again to stop the auto. ", (getWidth()/2-360)+50,
(getHeight()/2-600)+990, paint) ;                                //作者・作成年月の表示

```

```

        canvas.drawText("If you touch it further, it will be initialized.", (getWidth()/2-
360)+50, (getHeight()/2-600)+1030, paint);    //作者・作成年月の表示
        canvas.drawText("When the screen goes dark, touch the title bar !", (getWidth()/2-
360)+50, (getHeight()/2-600)+1070, paint);    //作者・作成年月の表示

        paint.setTextSize(30.0f);
        canvas.drawText("■ The coin that overlaps the grid point is", (getWidth()/2-360)+50,
(getHeight()/2-600)+425+300, paint);    //作者・作成年月の表示
        canvas.drawText(" represented by a red circle.", (getWidth()/2-360)+50,
(getHeight()/2-600)+445+300+10, paint);    //作者・作成年月の表示
        canvas.drawText("■ Let's observe that the pi is four times", (getWidth()/2-360)+50,
(getHeight()/2-600)+475+300+40-10, paint);    //作者・作成年月の表示
        canvas.drawText(" the value obtained by dividing (the number", (getWidth()/2-360)+50,
(getHeight()/2-600)+495+300+50-10, paint);    //作者・作成年月の表示
        canvas.drawText(" of coins that overlap the grid points)", (getWidth()/2-360)+50,
(getHeight()/2-600)+515+300+60-10, paint);    //作者・作成年月の表示
        canvas.drawText(" by (the total number of coins thrown)", (getWidth()/2-360)+50,
(getHeight()/2-600)+535+300+70-10, paint);    //作者・作成年月の表示

    //}

    paint.setColor(Color.BLUE);
    paint.setTextSize(30.0f);
    canvas.drawText("Copyright(C) Sohun 2021. 8. 28", (getWidth()/2-360)+150+10,
(getHeight()/2-600)+1130, paint);    //作者・作成年月の表示

    k=k+1;    //投げたコインの個数を1個増やす
    x[k]=40+120+400*Math.random();    //落ちるコインの位置 (x, y) を乱数
    で求める
    y[k]=100+400*Math.random();

    if ((x[k]-40-120)*(x[k]-40-120)+(y[k]-100)*(y[k]-100)<2500) {    //コインが格子点に重
なった場合
        sum++;    //格子点に重なったコインの個数を1つ増やす
    }
    else if ((x[k]-140-120)*(x[k]-140-120)+(y[k]-100)*(y[k]-100)<2500) {

```

```

        sum++;
    }
    else if ((x[k]-240-120)*(x[k]-240-120)+(y[k]-100)*(y[k]-100)<2500) {
        sum++;
    }
    else if ((x[k]-340-120)*(x[k]-340-120)+(y[k]-100)*(y[k]-100)<2500) {
        sum++;
    }
    else if ((x[k]-440-120)*(x[k]-440-120)+(y[k]-100)*(y[k]-100)<=2500) {
        sum++;
    }

    else if ((x[k]-40-120)*(x[k]-40-120)+(y[k]-200)*(y[k]-200)<2500) {
        sum++;
    }
    else if ((x[k]-140-120)*(x[k]-140-120)+(y[k]-200)*(y[k]-200)<2500) {
        sum++;
    }
    else if ((x[k]-240-120)*(x[k]-240-120)+(y[k]-200)*(y[k]-200)<2500) {
        sum++;
    }
    else if ((x[k]-340-120)*(x[k]-340-120)+(y[k]-200)*(y[k]-200)<2500) {
        sum++;
    }
    else if ((x[k]-440-120)*(x[k]-440-120)+(y[k]-200)*(y[k]-200)<2500) {
        sum++;
    }

    else if ((x[k]-40-120)*(x[k]-40-120)+(y[k]-300)*(y[k]-300)<2500) {
        sum++;
    }
    else if ((x[k]-140-120)*(x[k]-140-120)+(y[k]-300)*(y[k]-300)<2500) {
        sum++;
    }
    else if ((x[k]-240-120)*(x[k]-240-120)+(y[k]-300)*(y[k]-300)<2500) {
        sum++;
    }

```

```

}
else if ((x[k]-340-120)*(x[k]-340-120)+(y[k]-300)*(y[k]-300)<2500) {
    sum++;
}
else if ((x[k]-440-120)*(x[k]-440-120)+(y[k]-300)*(y[k]-300)<2500) {
    sum++;
}

else if ((x[k]-40-120)*(x[k]-40-120)+(y[k]-400)*(y[k]-400)<2500) {
    sum++;
}
else if ((x[k]-140-120)*(x[k]-140-120)+(y[k]-400)*(y[k]-400)<2500) {
    sum++;
}
else if ((x[k]-240-120)*(x[k]-240-120)+(y[k]-400)*(y[k]-400)<2500) {
    sum++;
}
else if ((x[k]-340-120)*(x[k]-340-120)+(y[k]-400)*(y[k]-400)<2500) {
    sum++;
}
else if ((x[k]-440-120)*(x[k]-440-120)+(y[k]-400)*(y[k]-400)<2500) {
    sum++;
}

else if ((x[k]-40-120)*(x[k]-40-120)+(y[k]-500)*(y[k]-500)<2500) {
    sum++;
}
else if ((x[k]-140-120)*(x[k]-140-120)+(y[k]-500)*(y[k]-500)<2500) {
    sum++;
}
else if ((x[k]-240-120)*(x[k]-240-120)+(y[k]-500)*(y[k]-500)<2500) {
    sum++;
}
else if ((x[k]-340-120)*(x[k]-340-120)+(y[k]-500)*(y[k]-500)<2500) {
    sum++;
}

```

```

else if ((x[k]-440-120)*(x[k]-440-120)+(y[k]-500)*(y[k]-500)<2500) {
    sum++;
}

for (i=1;i<=k;i++) {
    if ((x[i]-40-120)*(x[i]-40-120)+(y[i]-100)*(y[i]-100)<2500) { //コイン
        //格子点に重なった場合
        px=(int)x[i]+(getWidth()/2-360); //倍精度型変数を整数型変数に
        //キャストする
        py=(int)y[i]+(getHeight()/2-600);
        paint.setColor(Color.RED); //コインの色を赤にする
        paint.setStyle(Paint.Style.STROKE);
        canvas.drawCircle(px, py, 50, paint); //格子点に重なったコインを描く
    }
    else if ((x[i]-140-120)*(x[i]-140-120)+(y[i]-100)*(y[i]-100)<2500) {
        px=(int)x[i]+(getWidth()/2-360); //
        py=(int)y[i]+(getHeight()/2-600);
        paint.setColor(Color.RED); //
        paint.setStyle(Paint.Style.STROKE);
        canvas.drawCircle(px, py, 50, paint); //
    }
    else if ((x[i]-240-120)*(x[i]-240-120)+(y[i]-100)*(y[i]-100)<2500)
{
    //
    px=(int)x[i]+(getWidth()/2-360); //
    py=(int)y[i]+(getHeight()/2-600);
    paint.setColor(Color.RED); //
    paint.setStyle(Paint.Style.STROKE);
    canvas.drawCircle(px, py, 50, paint); //
}
    else if ((x[i]-340-120)*(x[i]-340-120)+(y[i]-100)*(y[i]-100)<2500)
{
    //
    px=(int)x[i]+(getWidth()/2-360); //
    py=(int)y[i]+(getHeight()/2-600);
    paint.setColor(Color.RED); //
    paint.setStyle(Paint.Style.STROKE);
    canvas.drawCircle(px, py, 50, paint); //
}
}

```

```

    }

    else if ((x[i]-440-120)*(x[i]-440-120)+(y[i]-100)*(y[i]-100)<2500)
    {
        //
        px=(int)x[i]+(getWidth()/2-360); //
        py=(int)y[i]+(getHeight()/2-600);
        paint.setColor(Color.RED); //
        paint.setStyle(Paint.Style.STROKE);
        canvas.drawCircle(px, py, 50, paint); //
    }

    else if ((x[i]-40-120)*(x[i]-40-120)+(y[i]-200)*(y[i]-200)<2500) { //
        px=(int)x[i]+(getWidth()/2-360); //
        py=(int)y[i]+(getHeight()/2-600);
        paint.setColor(Color.RED); //
        paint.setStyle(Paint.Style.STROKE);
        canvas.drawCircle(px, py, 50, paint); //
    }

    else if ((x[i]-140-120)*(x[i]-140-120)+(y[i]-200)*(y[i]-200)<2500)
    {
        //
        px=(int)x[i]+(getWidth()/2-360); //
        py=(int)y[i]+(getHeight()/2-600);
        paint.setColor(Color.RED); //
        paint.setStyle(Paint.Style.STROKE);
        canvas.drawCircle(px, py, 50, paint); //
    }

    else if ((x[i]-240-120)*(x[i]-240-120)+(y[i]-200)*(y[i]-200)<2500)
    {
        //
        px=(int)x[i]+(getWidth()/2-360); //
        py=(int)y[i]+(getHeight()/2-600);
        paint.setColor(Color.RED); //
        paint.setStyle(Paint.Style.STROKE);
        canvas.drawCircle(px, py, 50, paint); //
    }

    else if ((x[i]-340-120)*(x[i]-340-120)+(y[i]-200)*(y[i]-200)<2500)
    {
        //
        px=(int)x[i]+(getWidth()/2-360); //

```

```

        py=(int)y[i]+(getHeight()/2-600);
        paint.setColor(Color.RED);          //
        paint.setStyle(Paint.Style.STROKE);
        canvas.drawCircle(px, py, 50, paint);    //
    }
else if ((x[i]-440-120)*(x[i]-440-120)+(y[i]-200)*(y[i]-200)<2500)
{
    //
    px=(int)x[i]+(getWidth()/2-360);          //
    py=(int)y[i]+(getHeight()/2-600);
    paint.setColor(Color.RED);          //
    paint.setStyle(Paint.Style.STROKE);
    canvas.drawCircle(px, py, 50, paint);    //
}

else if ((x[i]-40-120)*(x[i]-40-120)+(y[i]-300)*(y[i]-300)<2500) {          //
    px=(int)x[i]+(getWidth()/2-360);          //
    py=(int)y[i]+(getHeight()/2-600);
    paint.setColor(Color.RED);          //
    paint.setStyle(Paint.Style.STROKE);
    canvas.drawCircle(px, py, 50, paint);    //
}

else if ((x[i]-140-120)*(x[i]-140-120)+(y[i]-300)*(y[i]-300)<2500)
{
    //
    px=(int)x[i]+(getWidth()/2-360);          //
    py=(int)y[i]+(getHeight()/2-600);
    paint.setColor(Color.RED);          //
    paint.setStyle(Paint.Style.STROKE);
    canvas.drawCircle(px, py, 50, paint);    //
}

else if ((x[i]-240-120)*(x[i]-240-120)+(y[i]-300)*(y[i]-300)<2500)
{
    //
    px=(int)x[i]+(getWidth()/2-360);          //
    py=(int)y[i]+(getHeight()/2-600);
    paint.setColor(Color.RED);          //
    paint.setStyle(Paint.Style.STROKE);
    canvas.drawCircle(px, py, 50, paint);    //
}

```

```

    }
    else if ((x[i]-340-120)*(x[i]-340-120)+(y[i]-300)*(y[i]-300)<2500)
    {
        //
        px=(int)x[i]+(getWidth()/2-360); //
        py=(int)y[i]+(getHeight()/2-600);
        paint.setColor(Color.RED); //
        paint.setStyle(Paint.Style.STROKE);
        canvas.drawCircle(px, py, 50, paint); //
    }
    else if ((x[i]-440-120)*(x[i]-440-120)+(y[i]-300)*(y[i]-300)<2500)
    {
        //
        px=(int)x[i]+(getWidth()/2-360); //
        py=(int)y[i]+(getHeight()/2-600);
        paint.setColor(Color.RED); //
        paint.setStyle(Paint.Style.STROKE);
        canvas.drawCircle(px, py, 50, paint); //
    }
}

else if ((x[i]-40-120)*(x[i]-40-120)+(y[i]-400)*(y[i]-400)<2500) { //
    px=(int)x[i]+(getWidth()/2-360); //
    py=(int)y[i]+(getHeight()/2-600);
    paint.setColor(Color.RED); //
    paint.setStyle(Paint.Style.STROKE);
    canvas.drawCircle(px, py, 50, paint); //
}

else if ((x[i]-140-120)*(x[i]-140-120)+(y[i]-400)*(y[i]-400)<2500)
{
    //
    px=(int)x[i]+(getWidth()/2-360); //
    py=(int)y[i]+(getHeight()/2-600);
    paint.setColor(Color.RED); //
    paint.setStyle(Paint.Style.STROKE);
    canvas.drawCircle(px, py, 50, paint); //
}

else if ((x[i]-240-120)*(x[i]-240-120)+(y[i]-400)*(y[i]-400)<2500)
{
    //
    px=(int)x[i]+(getWidth()/2-360); //

```

```

        py=(int)y[i]+(getHeight()/2-600);
        paint.setColor(Color.RED);          //
        paint.setStyle(Paint.Style.STROKE);
        canvas.drawCircle(px, py, 50, paint);    //
    }
else if ((x[i]-340-120)*(x[i]-340-120)+(y[i]-400)*(y[i]-400)<2500)
{
    //
    px=(int)x[i]+(getWidth()/2-360);          //
    py=(int)y[i]+(getHeight()/2-600);
    paint.setColor(Color.RED);          //
    paint.setStyle(Paint.Style.STROKE);
    canvas.drawCircle(px, py, 50, paint);    //
}
else if ((x[i]-440-120)*(x[i]-440-120)+(y[i]-400)*(y[i]-400)<2500)
{
    //
    px=(int)x[i]+(getWidth()/2-360);          //
    py=(int)y[i]+(getHeight()/2-600);
    paint.setColor(Color.RED);          //
    paint.setStyle(Paint.Style.STROKE);
    canvas.drawCircle(px, py, 50, paint);    //
}

else if ((x[i]-40-120)*(x[i]-40-120)+(y[i]-400)*(y[i]-400)<2500) {          //
    px=(int)x[i]+(getWidth()/2-360);          //
    py=(int)y[i]+(getHeight()/2-600);
    paint.setColor(Color.RED);          //
    paint.setStyle(Paint.Style.STROKE);
    canvas.drawCircle(px, py, 50, paint);    //
}
else if ((x[i]-140-120)*(x[i]-140-120)+(y[i]-400)*(y[i]-400)<2500)
{
    //
    px=(int)x[i]+(getWidth()/2-360);          //
    py=(int)y[i]+(getHeight()/2-600);
    paint.setColor(Color.RED);          //
    paint.setStyle(Paint.Style.STROKE);
    canvas.drawCircle(px, py, 50, paint);    //
}

```

```

    }
    else if ((x[i]-240-120)*(x[i]-240-120)+(y[i]-400)*(y[i]-400)<2500)
    {
        //
        px=(int)x[i]+(getWidth()/2-3360); //
        py=(int)y[i]+(getHeight()/2-600);
        paint.setColor(Color.RED); //
        paint.setStyle(Paint.Style.STROKE);
        canvas.drawCircle(px, py, 50, paint); //
    }
    else if ((x[i]-340-120)*(x[i]-340-120)+(y[i]-400)*(y[i]-400)<2500)
    {
        //
        px=(int)x[i]+(getWidth()/2-360); //
        py=(int)y[i]+(getHeight()/2-600);
        paint.setColor(Color.RED); //
        paint.setStyle(Paint.Style.STROKE);
        canvas.drawCircle(px, py, 50, paint); //
    }
    else if ((x[i]-440-120)*(x[i]-440-120)+(y[i]-400)*(y[i]-400)<2500)
    {
        //
        px=(int)x[i]+(getWidth()/2-360); //
        py=(int)y[i]+(getHeight()/2-600);
        paint.setColor(Color.RED); //
        paint.setStyle(Paint.Style.STROKE);
        canvas.drawCircle(px, py, 50, paint); //
    }
}

else if ((x[i]-40-120)*(x[i]-40-120)+(y[i]-500)*(y[i]-500)<2500) { //
    px=(int)x[i]+(getWidth()/2-360); //
    py=(int)y[i]+(getHeight()/2-600);
    paint.setColor(Color.RED); //
    paint.setStyle(Paint.Style.STROKE);
    canvas.drawCircle(px, py, 50, paint); //
}
else if ((x[i]-140-120)*(x[i]-140-120)+(y[i]-500)*(y[i]-500)<2500)
{
    //
    px=(int)x[i]+(getWidth()/2-360); //

```

```

        py=(int)y[i]+(getHeight()/2-600);
        paint.setColor(Color.RED);          //
        paint.setStyle(Paint.Style.STROKE);
        canvas.drawCircle(px, py, 50, paint);    //
    }
    else if ((x[i]-240-120)*(x[i]-240-120)+(y[i]-500)*(y[i]-500)<2500)
    {
        //
        px=(int)x[i]+(getWidth()/2-360);      //
        py=(int)y[i]+(getHeight()/2-600);
        paint.setColor(Color.RED);          //
        paint.setStyle(Paint.Style.STROKE);
        canvas.drawCircle(px, py, 50, paint);    //
    }
    else if ((x[i]-340-120)*(x[i]-340-120)+(y[i]-500)*(y[i]-500)<2500)
    {
        //
        px=(int)x[i]+(getWidth()/2-360);      //
        py=(int)y[i]+(getHeight()/2-600);
        paint.setColor(Color.RED);          //
        paint.setStyle(Paint.Style.STROKE);
        canvas.drawCircle(px, py, 50, paint);    //
    }
    else if ((x[i]-440-120)*(x[i]-440-120)+(y[i]-500)*(y[i]-500)<2500)
    {
        //
        px=(int)x[i]+(getWidth()/2-360);      //
        py=(int)y[i]+(getHeight()/2-600);
        paint.setColor(Color.RED);          //
        paint.setStyle(Paint.Style.STROKE);
        canvas.drawCircle(px, py, 50, paint);    //
    }
    else {                                     //コインが格子点に重ならなかった場合
        px=(int)x[i]+(getWidth()/2-360);      //落ちたコインの位置 x 座標
        py=(int)y[i]+(getHeight()/2-600);      //落ちたコインの位置 y 座標
        paint.setColor(Color.BLUE);          //コインの色を青にする
        paint.setStyle(Paint.Style.STROKE);
        canvas.drawCircle(px, py, 50, paint);    //格子点に重ならなかったコイン描
    }

```

```

    }
}

    pai=(double)4*sum/k;          //πの近似値の計算
    paint.setColor(Color.BLUE);
    paint.setTextSize(35.0f);
    canvas.drawText("Pi π ≐ "+pai, (getWidth()/2-360)+60-10, (getHeight()/2-600)+570+20,
paint);          //テキストと数値を絵として描く
    paint.setTextSize(28.0f);
    paint.setColor(Color.BLACK);
    canvas.drawText("Number of coins that overlap the grid point="+sum+"", (getWidth()/2-
360)+60-10, (getHeight()/2-600)+600+10+20, paint);          //テキストと数値を絵として描く
    paint.setTextSize(28.0f);
    canvas.drawText("Total number of coins thrown="+k+"", (getWidth()/2-360)+60-10,
(getHeight()/2-600)+630+20+20, paint);          //テキストと数値を絵として描く

    if (k<=N && flag==1) { //投げる最大のコインの個数以下で、かつコインを投げる識別子が1
の場合
        invalidate(); //再描画する (clear & goto onDraw)
    }

    if (k==N) {
        flag=2;    //コインを投げる識別子が2（初期化）の場合
    }

} //protected void onDraw(Canvas canvas)

//画面にタッチしたときのイベント処理-----
@Override
public boolean onTouchEvent(MotionEvent event) {

    flag=flag+1;    //flagに1を加える
    flag=flag % 3; //flagに1、2、0を代入する

    if (flag==0) { //コインを投げる識別子が0（停止）の場合

```

```

        sum=0;    //格子点と重なったコインの個数を0にする
        k=0;      //投げたコインの個数を0にする
    }

    invalidate();    //再描画する (clear & goto onDraw)
    return false;

}
}

```

[2]activity_main.xml

```

<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello World!"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <jp.kiyo.wuena.mytenencoineng.MyTenencoinEng
        android:id="@+id/myfview1"
        android:layout_height="match_parent"
        android:layout_width="match_parent"/>

</androidx.constraintlayout.widget.ConstraintLayout>

```

[3]MainActivity.java

/*

10円玉を投げて円周率 π を求める(英語版)

Android 4.1 (Jelly Bean)

Copyright (C) K. Niwa 2021. 8. 28

*/

package jp.kiyo.wuena.mytenencoineng;

import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;

public class MainActivity **extends** AppCompatActivity {

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

 setContentView(R.layout. *activity_main*);

 }

}