

[1]MyAmidakuji3.java

/*

あみだくじ

Android 4.1 (Jelly Bean)

Copyright (C) K. Niwa 2021. 2. 27

*/

package jp.kiyo.wuena.myamidakuji3;

import android.content.Context;

import android.graphics.Canvas;

import android.graphics.Color;

import android.graphics.Paint;

import android.graphics.Rect;

import android.util.AttributeSet;

import android.view.View;

import android.content.res.Resources; //画像用

import android.graphics.*;

import android.view.*;

public class MyAmidakuji3 **extends** View {

//変数宣言と初期化

private Bitmap **bitmap1** = **null**; //画像型として宣言し初期化する

int px=240, py=120; //当たり位置の座標

int xx, i, j; //あみだの描写に使用

int flag=0; //画面をタッチする前(0)か後(1)かに使用

int flag1; //あみだ横線の左端が縦線の奇数本目(1)か偶数本目(2)かに使用

int flag2=0; //既に、同じ位置にあみだ横線を引いている(1)か否(0)かに使用

int flag3=0; //内回りdoを繰り返す(0)、内回りdoから抜け出す(99)

int flag4=0; //あみだ抽選が一番下までとどいた(99)か否(0)かに使用

int flag5=0; //実験回数が80回を超えた(1)か否か(0)に使用

int ct=0; //あみだ横線本数に使用

```

int[] x=new int[51]; //あみだ横線の左端のx座標に使用 1次元配列
int[] y=new int[51]; //あみだ横線の左端のy座標に使用 1次元配列
int[][] yy=new int[21][11]; //あみだ横線の左端のソート後のy座標に使用 2次元配列
int tt; //ソートに使用
int r; //乱数
double r1, r2; //乱数
int count, caunt=0; //ループカウンター
int b1=0, b2=0, b3=0, b4=0, b5=0, b6=0, b7=0, b8=0, b9=0, b10=0; //あみだ抽選の最終位置のカウン
トに使用
int k1=0, k2=0, k3=0, k4=0, k5=0, k6=0, k7=0, k8=0, k9=0, k10=0; //あみだ横線の両端のy座標の設
定y y [k*] [j] に使用
int mini, kx, p, k; //あみだ抽選に使用

public MyAmidakuji3(Context context) {
    super(context);
    init(context);
}

public MyAmidakuji3(Context context, AttributeSet attrs) {
    super(context, attrs);
    init(context);
}

public MyAmidakuji3(Context context, AttributeSet attrs, int defStyle) {
    super(context, attrs, defStyle);
    init(context);
}

private void init(Context context) {
    Resources res = context.getResources(); //画像用
    bitmap1 = BitmapFactory.decodeResource(res, R.drawable.tama); //画像用
}

//onDraw メソッド
-----
@Override

```

```

protected void onDraw(Canvas canvas) {
    // TODO 自動生成されたメソッド・スタブ
    super.onDraw(canvas);
    //背景とタイトルの設定・表示-----
    -----

    canvas.drawColor(Color.WHITE);
    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); //外枠

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

    //外枠
    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.BLUE);
    paint.setTextSize(45.0f);
    canvas.drawText("【あみだくじは公平か?】", (getWidth()/2-360)+105, (getHeight()/2-
600)+70, paint);

    //初期化-----
    -----

    ct=0;
    int k1=0, k2=0, k3=0, k4=0, k5=0, k6=0, k7=0, k8=0, k9=0, k10=0; //あみだ横線の両端のy座標

```

の設定 y y [k は *] [j] に使用

```
flag4=0;
for (i=1;i<=50;i++) {
    x[i]=0;y[i]=0;
}
for (j=1;j<=10;j++) {
    for (i=1;i<=20;i++) {
        yy[i][j]=0;
    }
}

//-----
-----

if (bitmap1 != null) {
    canvas.drawBitmap(bitmap1, (getWidth()/2-360)+px-8, (getHeight()/2-600)+py+30-
8, paint); //当たり位置画像の表示
}

//内枠
paint.setColor(Color.BLACK);
canvas.drawLine((getWidth()/2-360)+100, (getHeight()/2-600)+120, (getWidth()/2-
360)+100, (getHeight()/2-600)+500+200-30, paint); //長方形の枠の描写
canvas.drawLine((getWidth()/2-360)+100, (getHeight()/2-600)+500+200-30, (getWidth()/2-
360)+630, (getHeight()/2-600)+500+200-30, paint);
canvas.drawLine((getWidth()/2-360)+630, (getHeight()/2-600)+500+200-30, (getWidth()/2-
360)+630, (getHeight()/2-600)+120, paint);
canvas.drawLine((getWidth()/2-360)+630, (getHeight()/2-600)+120, (getWidth()/2-
360)+100, (getHeight()/2-600)+120, paint);
//canvas.drawLine((getWidth()/2-360)+100, (getHeight()/2-600)+440+230, (getWidth()/2-
360)+630, (getHeight()/2-600)+440+230, paint);
canvas.drawLine((getWidth()/2-360)+100-1, (getHeight()/2-600)+120-1, (getWidth()/2-
360)+100-1, (getHeight()/2-600)+500+1+200-30, paint);
canvas.drawLine((getWidth()/2-360)+100-1, (getHeight()/2-600)+500+1+200-
30, (getWidth()/2-360)+630+1, (getHeight()/2-600)+500+1+200-30, paint);
canvas.drawLine((getWidth()/2-360)+630+1, (getHeight()/2-600)+500+1+200-
30, (getWidth()/2-360)+630+1, (getHeight()/2-600)+120-1, paint);
```

```

        canvas.drawLine((getWidth()/2-360)+630+1, (getHeight()/2-600)+120-1, (getWidth()/2-
360)+100-1, (getHeight()/2-600)+120-1, paint);
        //canvas.drawLine((getWidth()/2-360)+100, (getHeight()/2-600)+440+1+230, (getWidth()/2-
360)+630, (getHeight()/2-600)+440+1+230, paint);

        paint.setColor(Color.BLACK);
        paint.setTextSize(30.0f);
        canvas.drawText("■あみだくじは毎回ランダムに作成されます。", (getWidth()/2-360)+50,
(getHeight()/2-600)+610+100+20, paint);
        canvas.drawText("■上の赤い点の位置が当たりです。", (getWidth()/2-360)+50,
(getHeight()/2-600)+650+100+20, paint);
        canvas.drawText("■あみだ線の下端を選んだと考えます。", (getWidth()/2-360)+50,
(getHeight()/2-600)+690+100+20, paint);
        canvas.drawText("■当たりにたどり着く下端の位置をカウントし", (getWidth()/2-360)+50,
(getHeight()/2-600)+730+100+20, paint);
        canvas.drawText(" 当たりの回数を棒グラフで表示します。", (getWidth()/2-360)+50,
(getHeight()/2-600)+760+100+20, paint);

        paint.setColor(Color.BLACK);
        paint.setTextSize(30.0f);
        canvas.drawText("※ 画面を5回タッチすると自動になります。", (getWidth()/2-360)+50,
(getHeight()/2-600)+950, paint);
        canvas.drawText("※ あみだ抽選は自動で止まります。", (getWidth()/2-360)+50,
(getHeight()/2-600)+990, paint);
        canvas.drawText("※ 更に画面をタッチすると初期化されます。", (getWidth()/2-360)+50,
(getHeight()/2-600)+1030, paint);
        canvas.drawText("※ 画面が暗くなったらタイトルバーをタッチ！", (getWidth()/2-360)+50,
(getHeight()/2-600)+1070, paint);

        paint.setColor(Color.BLUE);
        paint.setTextSize(30.0f);
        canvas.drawText("Copyright(C).K.Niwa 2021.2.27", (getWidth()/2-360)+155,
(getHeight()/2-600)+1130, paint);

        //あみだ縦線の描写
        paint.setColor(Color.BLACK);

```

```
for (xx=140;xx<=590;xx=xx+50) {  
    canvas.drawLine((getWidth()/2-360)+xx, (getHeight()/2-600)+160, (getWidth()/2-  
360)+xx, (getHeight()/2-600)+275+115, paint);  
}
```

//左から右へあみだ横線50本の描写開始

```
for (i=1;i<=50;i++) {  
    //あみだ横線の左端のx座標を乱数で選ぶ  
    r1=9*Math.random();  
    r=(int)r1;  
  
    switch (r) {  
        case 0:  
            x[i]=140;  
            //あみだ横線の左端が縦線の奇数本目 flag1=1  
            flag1=1;  
            break;  
        case 1:  
            x[i]=190;  
            //あみだ横線の左端が縦線の偶数本目 flag1=2  
            flag1=2;  
            break;  
        case 2:  
            x[i]=240;  
            //あみだ横線の左端が縦線の奇数本目 flag1=1  
            flag1=1;  
            break;  
        case 3:  
            x[i]=290;  
            //あみだ横線の左端が縦線の偶数本目 flag1=2  
            flag1=2;  
            break;  
        case 4:  
            x[i]=340;  
            //あみだ横線の左端が縦線の奇数本目 flag1=1  
            flag1=1;
```

```

        break;
    case 5:
        x[i]=390;
        //あみだ横線の左端が縦線の偶数本目 flag1=2
        flag1=2;
        break;
    case 6:
        x[i]=440;
        //あみだ横線の左端が縦線の奇数本目 flag1=1
        flag1=1;
        break;
    case 7:
        x[i]=490;
        //あみだ横線の左端が縦線の偶数本目 flag1=2
        flag1=2;
        break;
    case 8:
        x[i]=540;
        //あみだ横線の左端が縦線の奇数本目 flag1=1
        flag1=1;
        break;
} //switch (r)

//あみだ横線の左端の y 座標を乱数で選ぶ
r2=Math.random();

switch (flag1) {
    //あみだ横線の左端の縦軸が奇数番目のとき
    case 1:
        if (r2<0.1) {
            y[i]=130+10;
            for (j=1;j<i;j++) {
                if (x[j]==x[i] && y[j]==y[i]) {
                    flag2=1; //既に、同じ位置にあみだ横線をひいているとき
                }
            }
        }
}

```

```

}
else if (r2<0.2) {
    y[i]=130+30;
    for (j=1;j<i;j++) {
        if (x[j]==x[i] && y[j]==y[i]) {
            flag2=1; //既に、同じ位置にあみだ横線をひいているとき
        }
    }
}
else if (r2<0.3) {
    y[i]=130+50;
    for (j=1;j<i;j++) {
        if (x[j]==x[i] && y[j]==y[i]) {
            flag2=1; //既に、同じ位置にあみだ横線をひいているとき
        }
    }
}
else if (r2<0.4) {
    y[i]=130+70;
    for (j=1;j<i;j++) {
        if (x[j]==x[i] && y[j]==y[i]) {
            flag2=1; //既に、同じ位置にあみだ横線をひいているとき
        }
    }
}
else if (r2<0.5) {
    y[i]=130+90;
    for (j=1;j<i;j++) {
        if (x[j]==x[i] && y[j]==y[i]) {
            flag2=1; //既に、同じ位置にあみだ横線をひいているとき
        }
    }
}
else if (r2<0.6) {
    y[i]=130+110;
    for (j=1;j<i;j++) {

```



```

        if (x[j]==x[i] && y[j]==y[i]) {
            flag2=1; //既に、同じ位置にあみだ横線をひいているとき
        }
    }
}
else if (r2<0.7) {
    y[i]=130+130;
    for (j=1;j<i;j++) {
        if (x[j]==x[i] && y[j]==y[i]) {
            flag2=1; //既に、同じ位置にあみだ横線をひいているとき
        }
    }
}
else if (r2<0.8) {
    y[i]=130+150;
    for (j=1;j<i;j++) {
        if (x[j]==x[i] && y[j]==y[i]) {
            flag2=1; //既に、同じ位置にあみだ横線をひいているとき
        }
    }
}
else if (r2<0.9) {
    y[i]=130+170;
    for (j=1;j<i;j++) {
        if (x[j]==x[i] && y[j]==y[i]) {
            flag2=1; //既に、同じ位置にあみだ横線をひいているとき
        }
    }
}
else if (r2<1) {
    y[i]=130+190;
    for (j=1;j<i;j++) {
        if (x[j]==x[i] && y[j]==y[i]) {
            flag2=1; //既に、同じ位置にあみだ横線をひいているとき
        }
    }
}
}
}

```

```
}
```

```
break;
```

```
//あみだ横線の左端の縦軸が偶数番目のとき
```

```
case 2:
```

```
if (r2<0.1) {  
    y[i]=130+15;  
    for (j=1;j<i;j++) {  
        if (x[j]==x[i] && y[j]==y[i]) {  
            flag2=1;  
        }  
    }  
}  
  
else if (r2<0.2) {  
    y[i]=130+35;  
    for (j=1;j<i;j++) {  
        if (x[j]==x[i] && y[j]==y[i]) {  
            flag2=1;  
        }  
    }  
}  
  
else if (r2<0.3) {  
    y[i]=130+55;  
    for (j=1;j<i;j++) {  
        if (x[j]==x[i] && y[j]==y[i]) {  
            flag2=1;  
        }  
    }  
}  
  
else if (r2<0.4) {  
    y[i]=130+75;  
    for (j=1;j<i;j++) {  
        if (x[j]==x[i] && y[j]==y[i]) {  
            flag2=1;  
        }  
    }  
}
```

```
}  
else if (r2<0.5) {  
    y[i]=130+95;  
    for (j=1;j<i;j++) {  
        if (x[j]==x[i] && y[j]==y[i]) {  
            flag2=1;  
        }  
    }  
}  
else if (r2<0.6) {  
    y[i]=130+115;  
    for (j=1;j<i;j++) {  
        if (x[j]==x[i] && y[j]==y[i]) {  
            flag2=1;  
        }  
    }  
}  
else if (r2<0.7) {  
    y[i]=130+135;  
    for (j=1;j<i;j++) {  
        if (x[j]==x[i] && y[j]==y[i]) {  
            flag2=1;  
        }  
    }  
}  
else if (r2<0.8) {  
    y[i]=130+155;  
    for (j=1;j<i;j++) {  
        if (x[j]==x[i] && y[j]==y[i]) {  
            flag2=1;  
        }  
    }  
}  
else if (r2<0.9) {  
    y[i]=130+175;  
    for (j=1;j<i;j++) {
```

```

        if (x[j]==x[i] && y[j]==y[i]) {
            flag2=1;
        }
    }
}
else if (r2<1) {
    y[i]=130+195;
    for (j=1;j<i;j++) {
        if (x[j]==x[i] && y[j]==y[i]) {
            flag2=1;
        }
    }
}
break;
} //switch (flag1)

//既に、同じあみだ横線があったとき
if (flag2==1) {
    i=i-1; //あみだ横線の左端座標 x [ i ]、 y [ i ] をもう一度探す
    flag2=0;
}

//同じあみだ横線がなかったとき
else if (flag2==0) {
    ct=ct+1; //あみだ横線本数
    paint.setColor (Color. BLACK);
    canvas.drawLine((getWidth()/2-360)+x[i], (getHeight()/2-
600)+y[i]+30, (getWidth()/2-360)+x[i]+50, (getHeight()/2-600)+y[i]+30, paint);
}
} //for (i=1; i<=50; i++)

//あみだ横線の両端の y 座標の設定 y y [ i ] [ j ] の開始
for (i=1; i<=50; i++) {
    if (x[i]==140) {
        k1=k1+1;
        yy[k1][1]=y[i];
        k2=k2+1;
    }
}

```

```
        yy[k2][2]=y[i];
    }
    else if (x[i]==190) {
        k2=k2+1;
        yy[k2][2]=y[i];
        k3=k3+1;
        yy[k3][3]=y[i];
    }
    else if (x[i]==240) {
        k3=k3+1;
        yy[k3][3]=y[i];
        k4=k4+1;
        yy[k4][4]=y[i];
    }
    else if (x[i]==290) {
        k4=k4+1;
        yy[k4][4]=y[i];
        k5=k5+1;
        yy[k5][5]=y[i];
    }
    else if (x[i]==340) {
        k5=k5+1;
        yy[k5][5]=y[i];
        k6=k6+1;
        yy[k6][6]=y[i];
    }
    else if (x[i]==390) {
        k6=k6+1;
        yy[k6][6]=y[i];
        k7=k7+1;
        yy[k7][7]=y[i];
    }
    else if (x[i]==440) {
        k7=k7+1;
        yy[k7][7]=y[i];
        k8=k8+1;
```

```

        yy[k8][8]=y[i];
    }
    else if (x[i]==490) {
        k8=k8+1;
        yy[k8][8]=y[i];
        k9=k9+1;
        yy[k9][9]=y[i];
    }
    else if (x[i]==540) {
        k9=k9+1;
        yy[k9][9]=y[i];
        k10=k10+1;
        yy[k10][10]=y[i];
    }
} //for (i=1; i<=50; i++)

```

//あみだ横線の両端のy座標のソート（昇順）開始

```

for (j=1; j<=19; j++) {
    for (i=j+1; i<=20; i++) {
        if (yy[i][1] < yy[j][1]) {
            tt=yy[i][1];
            yy[i][1]=yy[j][1];
            yy[j][1]=tt;
        }
    }
}
for (j=1; j<=19; j++) {
    for (i=j+1; i<=20; i++) {
        if (yy[i][2] < yy[j][2]) {
            tt=yy[i][2];
            yy[i][2]=yy[j][2];
            yy[j][2]=tt;
        }
    }
}
for (j=1; j<=19; j++) {

```

```
    for (i=j+1; i<=20; i++) {
        if (yy[i][3] < yy[j][3]) {
            tt=yy[i][3];
            yy[i][3]=yy[j][3];
            yy[j][3]=tt;
        }
    }
}

for (j=1; j<=19; j++) {
    for (i=j+1; i<=20; i++) {
        if (yy[i][4] < yy[j][4]) {
            tt=yy[i][4];
            yy[i][4]=yy[j][4];
            yy[j][4]=tt;
        }
    }
}

for (j=1; j<=19; j++) {
    for (i=j+1; i<=20; i++) {
        if (yy[i][5] < yy[j][5]) {
            tt=yy[i][5];
            yy[i][5]=yy[j][5];
            yy[j][5]=tt;
        }
    }
}

for (j=1; j<=19; j++) {
    for (i=j+1; i<=20; i++) {
        if (yy[i][6] < yy[j][6]) {
            tt=yy[i][6];
            yy[i][6]=yy[j][6];
            yy[j][6]=tt;
        }
    }
}

for (j=1; j<=19; j++) {
```

```
    for (i=j+1; i<=20; i++) {
        if (yy[i][7] < yy[j][7]) {
            tt=yy[i][7];
            yy[i][7]=yy[j][7];
            yy[j][7]=tt;
        }
    }
}
for (j=1; j<=19; j++) {
    for (i=j+1; i<=20; i++) {
        if (yy[i][8] < yy[j][8]) {
            tt=yy[i][8];
            yy[i][8]=yy[j][8];
            yy[j][8]=tt;
        }
    }
}
for (j=1; j<=19; j++) {
    for (i=j+1; i<=20; i++) {
        if (yy[i][9] < yy[j][9]) {
            tt=yy[i][9];
            yy[i][9]=yy[j][9];
            yy[j][9]=tt;
        }
    }
}
for (j=1; j<=19; j++) {
    for (i=j+1; i<=20; i++) {
        if (yy[i][10] < yy[j][10]) {
            tt=yy[i][10];
            yy[i][10]=yy[j][10];
            yy[j][10]=tt;
        }
    }
}
}
```



```
////////////////////////////////////  
////////////////////////////////////
```

```
    if (flag==1) {  
        //あみだ抽選開始  
        mini=130; //現時点での最小のy座標  
        kx=240;    //現時点でのx座標  
        p=3;      //現時点での縦線の位置  
        paint.setColor(Color.RED);  
        flag4=0;  
  
        do{  
  
            k=1;  
            flag3=0;  
            do {  
                if (p % 2 == 1) { //縦線が奇数番目のとき  
  
                    if (yy[k][p]>mini) {  
                        canvas.drawLine((getWidth()/2-360)+kx, (getHeight()/2-  
600)+mini+30, (getWidth()/2-360)+kx, (getHeight()/2-600)+yy[k][p]+30, paint);  
                        canvas.drawLine((getWidth()/2-360)+kx-1, (getHeight()/2-  
600)+mini+30, (getWidth()/2-360)+kx-1, (getHeight()/2-600)+yy[k][p]+30, paint);  
                        canvas.drawLine((getWidth()/2-360)+kx+1, (getHeight()/2-  
600)+mini+30, (getWidth()/2-360)+kx+1, (getHeight()/2-600)+yy[k][p]+30, paint);  
                        if (yy[k][p] % 10 == 0) {  
                            canvas.drawLine((getWidth()/2-360)+kx, (getHeight()/2-  
600)+yy[k][p]+30, (getWidth()/2-360)+kx+50, (getHeight()/2-600)+yy[k][p]+30, paint);  
                            canvas.drawLine((getWidth()/2-360)+kx, (getHeight()/2-  
600)+yy[k][p]-1+30, (getWidth()/2-360)+kx+50, (getHeight()/2-600)+yy[k][p]-1+30, paint);  
                            canvas.drawLine((getWidth()/2-360)+kx, (getHeight()/2-  
600)+yy[k][p]+1+30, (getWidth()/2-360)+kx+50, (getHeight()/2-600)+yy[k][p]+1+30, paint);  
                            mini=yy[k][p];  
                            kx=kx+50;  
                            p=p+1;  
                        }  
                    }  
                }  
            }  
        }  
    }
```

```

        else if (yy[k][p] % 10 != 0) {
            canvas.drawLine((getWidth()/2-360)+kx, (getHeight()/2-
600)+yy[k][p]+30, (getWidth()/2-360)+kx-50, (getHeight()/2-600)+yy[k][p]+30, paint);
            canvas.drawLine((getWidth()/2-360)+kx, (getHeight()/2-
600)+yy[k][p]+1+30, (getWidth()/2-360)+kx-50, (getHeight()/2-600)+yy[k][p]+1+30, paint);
            canvas.drawLine((getWidth()/2-360)+kx, (getHeight()/2-
600)+yy[k][p]-1+30, (getWidth()/2-360)+kx-50, (getHeight()/2-600)+yy[k][p]-1+30, paint);
            mini=yy[k][p];
            kx=kx-50;
            p=p-1;
        }
        flag3=99;
    }
    else { //yy[k][p]<=mini
        k=k+1;
        if (k>20) {
            canvas.drawLine((getWidth()/2-360)+kx, (getHeight()/2-
600)+mini+30, (getWidth()/2-360)+kx, (getHeight()/2-600)+245+30+115, paint);
            canvas.drawLine((getWidth()/2-360)+kx-1, (getHeight()/2-
600)+mini+30, (getWidth()/2-360)+kx-1, (getHeight()/2-600)+245+30+115, paint);
            canvas.drawLine((getWidth()/2-360)+kx+1, (getHeight()/2-
600)+mini+30, (getWidth()/2-360)+kx+1, (getHeight()/2-600)+245+30+115, paint);

            //抽選位置の判断の開始
            switch (kx) {
                case 140:
                    b1=b1+1;
                    break;
                case 190:
                    b2=b2+1;
                    break;
                case 240:
                    b3=b3+1;
                    break;
                case 290:
                    b4=b4+1;

```

```

        break;
    case 340:
        b5=b5+1;
        break;
    case 390:
        b6=b6+1;
        break;
    case 440:
        b7=b7+1;
        break;
    case 490:
        b8=b8+1;
        break;
    case 540:
        b9=b9+1;
        break;
    case 590:
        b10=b10+1;
        break;
} //switch (kx)

//棒グラフの描写
paint.setColor (Color. BLUE);
canvas.drawRect ((getWidth () /2-360)+130, (getHeight () /2-
600)+410+30+220, (getWidth () /2-360)+150, (getHeight () /2-600)+410+30+220-15*b1, paint);
canvas.drawRect ((getWidth () /2-360)+180, (getHeight () /2-
600)+410+30+220, (getWidth () /2-360)+200, (getHeight () /2-600)+410+30+220-15*b2, paint);
canvas.drawRect ((getWidth () /2-360)+230, (getHeight () /2-
600)+410+30+220, (getWidth () /2-360)+250, (getHeight () /2-600)+410+30+220-15*b3, paint);
canvas.drawRect ((getWidth () /2-360)+280, (getHeight () /2-
600)+410+30+220, (getWidth () /2-360)+300, (getHeight () /2-600)+410+30+220-15*b4, paint);
canvas.drawRect ((getWidth () /2-360)+330, (getHeight () /2-
600)+410+30+220, (getWidth () /2-360)+350, (getHeight () /2-600)+410+30+220-15*b5, paint);
canvas.drawRect ((getWidth () /2-360)+380, (getHeight () /2-
600)+410+30+220, (getWidth () /2-360)+400, (getHeight () /2-600)+410+30+220-15*b6, paint);
canvas.drawRect ((getWidth () /2-360)+430, (getHeight () /2-

```

```

600)+410+30+220, (getWidth()/2-360)+450, (getHeight()/2-600)+410+30+220-15*b7, paint);
        canvas.drawRect((getWidth()/2-360)+480, (getHeight()/2-
600)+410+30+220, (getWidth()/2-360)+500, (getHeight()/2-600)+410+30+220-15*b8, paint);
        canvas.drawRect((getWidth()/2-360)+530, (getHeight()/2-
600)+410+30+220, (getWidth()/2-360)+550, (getHeight()/2-600)+410+30+220-15*b9, paint);
        canvas.drawRect((getWidth()/2-360)+580, (getHeight()/2-
600)+410+30+220, (getWidth()/2-360)+600, (getHeight()/2-600)+410+30+220-15*b10, paint);
        paint.setColor(Color.BLACK);

        flag3=99;
        flag4=99;
        flag=0;
    }//if (k>20) {
    }//else {
}//if (p % 2 == 1)

else if (p % 2 == 0) { //縦線が偶数番目のとき

    if (yy[k][p]>mini) {
        canvas.drawLine((getWidth()/2-360)+kx, (getHeight()/2-
600)+mini+30, (getWidth()/2-360)+kx, (getHeight()/2-600)+yy[k][p]+30, paint);
        canvas.drawLine((getWidth()/2-360)+kx-1, (getHeight()/2-
600)+mini+30, (getWidth()/2-360)+kx-1, (getHeight()/2-600)+yy[k][p]+30, paint);
        canvas.drawLine((getWidth()/2-360)+kx+1, (getHeight()/2-
600)+mini+30, (getWidth()/2-360)+kx+1, (getHeight()/2-600)+yy[k][p]+30, paint);
        if (yy[k][p] % 10 != 0) {
            canvas.drawLine((getWidth()/2-360)+kx, (getHeight()/2-
600)+yy[k][p]+30, (getWidth()/2-360)+kx+50, (getHeight()/2-600)+yy[k][p]+30, paint);
            canvas.drawLine((getWidth()/2-360)+kx, (getHeight()/2-
600)+yy[k][p]-1+30, (getWidth()/2-360)+kx+50, (getHeight()/2-600)+yy[k][p]-1+30, paint);
            canvas.drawLine((getWidth()/2-360)+kx, (getHeight()/2-
600)+yy[k][p]+1+30, (getWidth()/2-360)+kx+50, (getHeight()/2-600)+yy[k][p]+1+30, paint);
            mini=yy[k][p];
            kx=kx+50;
            p=p+1;
        }
    }
}

```

```

        else if (yy[k][p] % 10 == 0) {
            canvas.drawLine((getWidth()/2-360)+kx, (getHeight()/2-
600)+yy[k][p]+30, (getWidth()/2-360)+kx-50, (getHeight()/2-600)+yy[k][p]+30, paint);
            canvas.drawLine((getWidth()/2-360)+kx, (getHeight()/2-
600)+yy[k][p]-1+30, (getWidth()/2-360)+kx-50, (getHeight()/2-600)+yy[k][p]-1+30, paint);
            canvas.drawLine((getWidth()/2-360)+kx, (getHeight()/2-
600)+yy[k][p]+1+30, (getWidth()/2-360)+kx-50, (getHeight()/2-600)+yy[k][p]+1+30, paint);
            mini=yy[k][p];
            kx=kx-50;
            p=p-1;
        }
        flag3=99;
    }
    else { //yy[k][p]<=mini
        k=k+1;
        if (k>20) {
            //k=k-1;
            canvas.drawLine((getWidth()/2-360)+kx, (getHeight()/2-
600)+mini+30, (getWidth()/2-360)+kx, (getHeight()/2-600)+245+30+115, paint);
            canvas.drawLine((getWidth()/2-360)+kx-1, (getHeight()/2-
600)+mini+30, (getWidth()/2-360)+kx-1, (getHeight()/2-600)+245+30+115, paint);
            canvas.drawLine((getWidth()/2-360)+kx+1, (getHeight()/2-
600)+mini+30, (getWidth()/2-360)+kx+1, (getHeight()/2-600)+245+30+115, paint);

            //抽選位置の判断の開始
            switch (kx) {
                case 140:
                    b1=b1+1;
                    break;
                case 190:
                    b2=b2+1;
                    break;
                case 240:
                    b3=b3+1;
                    break;
                case 290:

```

```

        b4=b4+1;
        break;
    case 340:
        b5=b5+1;
        break;
    case 390:
        b6=b6+1;
        break;
    case 440:
        b7=b7+1;
        break;
    case 490:
        b8=b8+1;
        break;
    case 540:
        b9=b9+1;
        break;
    case 590:
        b10=b10+1;
        break;
} //switch (kx)

//棒グラフの描写
paint.setColor (Color. BLUE);
canvas.drawRect ((getWidth () /2-360)+130, (getHeight () /2-
600)+410+30+220, (getWidth () /2-360)+150, (getHeight () /2-600)+410+30+220-15*b1, paint);
canvas.drawRect ((getWidth () /2-360)+180, (getHeight () /2-
600)+410+30+220, (getWidth () /2-360)+200, (getHeight () /2-600)+410+30+220-15*b2, paint);
canvas.drawRect ((getWidth () /2-360)+230, (getHeight () /2-
600)+410+30+220, (getWidth () /2-360)+250, (getHeight () /2-600)+410+30+220-15*b3, paint);
canvas.drawRect ((getWidth () /2-360)+280, (getHeight () /2-
600)+410+30+220, (getWidth () /2-360)+300, (getHeight () /2-600)+410+30+220-15*b4, paint);
canvas.drawRect ((getWidth () /2-360)+330, (getHeight () /2-
600)+410+30+220, (getWidth () /2-360)+350, (getHeight () /2-600)+410+30+220-15*b5, paint);
canvas.drawRect ((getWidth () /2-360)+380, (getHeight () /2-
600)+410+30+220, (getWidth () /2-360)+400, (getHeight () /2-600)+410+30+220-15*b6, paint);

```

```

        canvas.drawRect((getWidth()/2-360)+430, (getHeight()/2-
600)+410+30+220, (getWidth()/2-360)+450, (getHeight()/2-600)+410+30+220-15*b7, paint);
        canvas.drawRect((getWidth()/2-360)+480, (getHeight()/2-
600)+410+30+220, (getWidth()/2-360)+500, (getHeight()/2-600)+410+30+220-15*b8, paint);
        canvas.drawRect((getWidth()/2-360)+530, (getHeight()/2-
600)+410+30+220, (getWidth()/2-360)+550, (getHeight()/2-600)+410+30+220-15*b9, paint);
        canvas.drawRect((getWidth()/2-360)+580, (getHeight()/2-
600)+410+30+220, (getWidth()/2-360)+600, (getHeight()/2-600)+410+30+220-15*b10, paint);
        paint.setColor(Color.BLACK);

        flag3=99;
        flag4=99;
        flag=0;
        }//if (k>20) {
        }//else {
        }//else if (p % 2 == 0) {
        } while (flag3!=99);
        }while(flag4!=99);
    }//if (flag==1) {

    caunt=caunt+1;
    if (caunt>5 && caunt<76) {
        flag=1;
        ct=0;
        k1=0;k2=0;k3=0;k4=0;k5=0;k6=0;k7=0;k8=0;k9=0;k10=0; //あみだ横線の両端の y 座標
        の設定 y y [k*] [j] に使用
        flag3=0;
        flag4=0;
        for (i=1;i<=50;i++) {
            x[i]=0;y[i]=0;
        }
        for (j=1;j<=10;j++) {
            for (i=1;i<=20;i++) {
                yy[i][j]=0;
            }
        }
    }
}

```

```

        invalidate(); //repaint()メソッドと同じ、再描画、onDrawメソッドに戻る、画面をク
リアする
    }
    else if (caunt>75) {
        caunt=0;
        flag5=1;
    }

} //protected void onDraw(Canvas canvas) {

////////////////////////////////////
////////////////////////////////////

@Override
public boolean onTouchEvent(MotionEvent event) {

    if (flag5==1) {
        b1=0;b2=0;b3=0;b4=0;b5=0;b6=0;b7=0;b8=0;b9=0;b10=0;
        flag5=0;
        flag=0;
    }
    else {
        flag=1;
    }

    ct=0;
    int k1=0, k2=0, k3=0, k4=0, k5=0, k6=0, k7=0, k8=0, k9=0, k10=0; //あみだ横線の両端の y 座標
の設定 y y [ k * ] [ j ] に使用
    flag3=0;
    flag4=0;
    for (i=1; i<=50; i++) {
        x[i]=0; y[i]=0;
    }
    for (j=1; j<=10; j++) {
        for (i=1; i<=20; i++) {

```



```

        yy[i][j]=0;
    }
}

invalidate(); //repaint()メソッドと同じ、再描画、onDrawメソッドに戻る、画面をクリア
する

return false;

}

} //public class MyAmida extends View {

```

[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.myamidakuji3.MyAmidakuji3
    android:id="@+id/myfview1"
    android:layout_height="match_parent"
    android:layout_width="match_parent"/>

```

```
</androidx.constraintlayout.widget.ConstraintLayout>
```

```
[3]MainActivity.java
```

```
/*
```

```
-----  
                あみだくじ  
                Android 4.1 (Jelly Bean)  
                Copyright (C) K. Niwa 2021. 2. 27  
-----
```

```
*/
```

```
package jp.kiyo.wuena.myamidakuji3;
```

```
import androidx.appcompat.app.AppCompatActivity;
```

```
import android.os.Bundle;
```

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

```
import android.app.Activity;
```

```
import android.view.Menu;
```

```
public class MainActivity extends AppCompatActivity {
```

```
    static int ritsu;
```

```
    @Override
```

```
    protected void onCreate(Bundle savedInstanceState) {
```

```
        super.onCreate(savedInstanceState);
```

```
        setContentView(R.layout.activity_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;
```

