［１］MyTwokansu.java

```
/*
-----------------------------------------------------------------
          ２次関数のグラフの平行移動
            Android 4.4（Kit Kat）
       Copyright(C) K.Niwa 2019.12.13
-----------------------------------------------------------------
*/

package jp.kiyo.wuena.mytwokansu;

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;

import android.graphics.*;
import android.util.AttributeSet;
import android.view.*;

//View クラスを継承した Twokan1 クラス
public class MyTwokansu extends View {

    int flag=0;     //グラフの移動(1)、グラフの停止(2)、グラフの初期化(0)初期化　識別子
    double x,y;     //グラフ描写に利用
    int px,py,oldpx,oldpy;   //グラフ描写に利用
    int fg;          //グラフ描写に利用
    double a=0,b=0;          //グラフ描写に利用
    static int n1,n2,n3;

    public MyTwokansu(Context context, AttributeSet attrs, int defStyle) {  //コンストラクタ（クラス名と同じメソッドで最初に読まれる）
        super(context, attrs, defStyle);
    }
```

```
    public MyTwokansu(Context context, AttributeSet attrs) {      //コンストラクタ（クラス名と同じメソッドで最初に読まれる）
        super(context, attrs);
    }

    public MyTwokansu(Context context) {    //コンストラクタ（クラス名と同じメソッドで最初に読まれる）
        super(context);
    }

    //onDraw メソッド----------------------------------------------------------------------------
    @Override
    protected void onDraw(Canvas canvas) {

        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,(getHeight()/2-343)+675,paint);
        paint.setAlpha(10000);
        paint.setColor(Color.BLUE);

        //額縁を付ける
        for（int i=0;i<2;i++）{
            canvas.drawLine((getWidth()/2-240)+10+i,(getHeight()/2-343)+10+i,(getWidth()/2-240)+10+i,(getHeight()/2-343)+675-i,paint);
            canvas.drawLine((getWidth()/2-240)+10+i,(getHeight()/2-343)+675-i,(getWidth()/2-240)+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()/2-240)+10+i,(getHeight()/2-343)+10+i,paint);
        }

        //実験枠の描画
        paint.setColor(Color.BLACK);
          canvas.drawRect((getWidth()/2-240)+90,  (getHeight()/2-343)+100,(getWidth()
```

```
/2-240)+390,(getHeight()/2-343)+400,paint);
        paint.setColor(Color.WHITE);
          canvas.drawRect((getWidth()/2-240)+91,  (getHeight()/2-343)+101,(getWidth()
/2-240)+389,(getHeight()/2-343)+399,paint);

        //座標軸の描画
        paint.setColor(Color.BLACK);
          canvas.drawLine((getWidth()/2-240)+100,  (getHeight()/2-343)+250,(getWidth()
/2-240)+380,(getHeight()/2-343)+250,paint);
          canvas.drawLine((getWidth()/2-240)+380,  (getHeight()/2-343)+250,(getWidth()
/2-240)+375,(getHeight()/2-343)+245,paint);
          canvas.drawLine((getWidth()/2-240)+380,  (getHeight()/2-343)+250,(getWidth()
/2-240)+375,(getHeight()/2-343)+255,paint);
          canvas.drawLine((getWidth()/2-240)+240,  (getHeight()/2-343)+110,(getWidth()
/2-240)+240,(getHeight()/2-343)+390,paint);
          canvas.drawLine((getWidth()/2-240)+240,  (getHeight()/2-343)+110,(getWidth()
/2-240)+245,(getHeight()/2-343)+115,paint);
          canvas.drawLine((getWidth()/2-240)+240,  (getHeight()/2-343)+110,(getWidth()
/2-240)+235,(getHeight()/2-343)+115,paint);

        //x軸メモリの描写
        for(int xx=110;xx<380;xx=xx+10){
            canvas.drawLine((getWidth()/2-240)+xx,  (getHeight()/2-343)+247,(getWidth()
/2-240)+xx,(getHeight()/2-343)+253,paint);
        }

        //y軸メモリの描写
        for(int yy=120;yy<390;yy=yy+10){
            canvas.drawLine((getWidth()/2-240)+237,  (getHeight()/2-343)+yy,(getWidth()
/2-240)+243,(getHeight()/2-343)+yy,paint);
        }

        //x軸メモリの描写
        canvas.drawText("5",(getWidth()/2-240)+287,(getHeight()/2-343)+265, paint);
        canvas.drawText("10",(getWidth()/2-240)+334,(getHeight()/2-343)+265, paint);
        canvas.drawText("x",(getWidth()/2-240)+375,(getHeight()/2-343)+265, paint);
        canvas.drawText("O",(getWidth()/2-240)+227,(getHeight()/2-343)+265, paint);
        canvas.drawText("-5",(getWidth()/2-240)+187,(getHeight()/2-343)+265, paint);
        canvas.drawText("-10",(getWidth()/2-240)+134,(getHeight()/2-343)+265, paint);
```

```
        //ｙ軸メモリの描写
        canvas.drawText("5",（getWidth()/2-240)+225,（getHeight()/2-343)+205, paint);
        canvas.drawText("10",（getWidth()/2-240)+220,（getHeight()/2-343)+155, paint);
        canvas.drawText("ｙ",（getWidth()/2-240)+220,（getHeight()/2-343)+115, paint);
        canvas.drawText("-5",（getWidth()/2-240)+220,（getHeight()/2-343)+305, paint);
        canvas.drawText("-10",（getWidth()/2-240)+215,（getHeight()/2-343)+355, paint);

        //y=ax^2 のグラフの描画//
        paint.setColor(Color.RED);

        if(MainActivity.n1 < 0){
            canvas.drawText("ｙ＝ａｘ　", （getWidth()/2-240)+280, （getHeight()/2-343)
+350, paint);
            canvas.drawText("2",（getWidth()/2-240)+322-5+1+1,（getHeight()/2-343)+345,
paint);
        }
        else if(MainActivity.n1 > 0){
            canvas.drawText("ｙ＝ａｘ　", （getWidth()/2-240)+280, （getHeight()/2-343)
+140, paint);
            canvas.drawText("2",（getWidth()/2-240)+322-5+1+1,（getHeight()/2-343)+135,
paint);
        }

        fg=0;

        for（x=-4+a;x<=4+a;x=x+0.01）{
            y=MainActivity.n1*(x-a)*(x-a)+b;

            px=(int)(240+10*x);
            py=(int)(250-10*y);
            if（fg==0）{
                if((getHeight()/2-343)+py>(getHeight()/2-343)+110　&&　(getHeight()
/2-343)+py<(getHeight()/2-343)+390){
                    canvas.drawLine((int)(getWidth()/2-240)+px,　(int)(getHeight()
/2-343)+py,(int)(getWidth()/2-240)+px,(int)(getHeight()/2-343)+py,paint);
                }
            }
            else {
                if((getHeight()/2-343)+py>(getHeight()/2-343)+110　&&　(getHeight()
/2-343)+py<(getHeight()/2-343)+390){
```

```
                        canvas.drawLine((int)(getWidth()/2-240)+oldpx, (int)(getHeight()
/2-343)+oldpy,(int)(getWidth()/2-240)+px,(int)(getHeight()/2-343)+py,paint);
                    }
                }
                oldpx=px;oldpy=py;
                fg++;
            }

            //y=a(x-b)^2+c のグラフの描画
            paint.setColor(Color.BLUE);

            if(MainActivity.n1 > 0 && MainActivity.n2 > 0){
                canvas.drawText("y = a(x -b)      +c", (getWidth()/2-240)+270, (getHeight()
/2-343)+375, paint);
                canvas.drawText("2", (getWidth()/2-240)+335-5+1, (getHeight()/2-343)+370,
paint);
            }
            else if(MainActivity.n1 > 0 && MainActivity.n2 < 0){
                canvas.drawText("y = a(x -b)      +c", (getWidth()/2-240)+120, (getHeight()
/2-343)+375, paint);
                canvas.drawText("2", (getWidth()/2-240)+185-5+1, (getHeight()/2-343)+370,
paint);
            }
            else if(MainActivity.n1 < 0 && MainActivity.n2 < 0){
                canvas.drawText("y = a(x -b)      +c", (getWidth()/2-240)+120, (getHeight()
/2-343)+135, paint);
                canvas.drawText("2", (getWidth()/2-240)+185-5+1, (getHeight()/2-343)+130,
paint);
            }
            else if(MainActivity.n1 < 0 && MainActivity.n2 > 0){
                canvas.drawText("y = a(x -b)      +c", (getWidth()/2-240)+270, (getHeight()
/2-343)+135, paint);
                canvas.drawText("2", (getWidth()/2-240)+335-5+1, (getHeight()/2-343)+130,
paint);
            }

            fg=0;

            for（x=MainActivity.n2-4;x<=MainActivity.n2+4;x=x+0.01）{
                y=MainActivity.n1*(x-MainActivity.n2)*(x-MainActivity.n2)+MainActivity.n3;
```

```
                px=(int)(240+10*x);
                py=(int)(250-10*y);
                if (fg==0) {
                    if((getHeight()/2-343)+py>(getHeight()/2-343)+110 && (getHeight()
/2-343)+py<(getHeight()/2-343)+390){
                        canvas.drawLine((int)(getWidth()/2-240)+px, (int)(getHeight()
/2-343)+py,(int)(getWidth()/2-240)+px,(int)(getHeight()/2-343)+py,paint);
                    }
                }
                else {
                    if((getHeight()/2-343)+py>(getHeight()/2-343)+110 && (getHeight()
/2-343)+py<(getHeight()/2-343)+390){
                        canvas.drawLine((int)(getWidth()/2-240)+oldpx, (int)(getHeight()
/2-343)+oldpy,(int)(getWidth()/2-240)+px,(int)(getHeight()/2-343)+py,paint);
                    }
                }
                oldpx=px;oldpy=py;
                fg++;
            }

            //実験枠の描画
            paint.setColor(Color.BLACK);
              canvas.drawLine((getWidth()/2-240)+90, (getHeight()/2-343)+100,(getWidth()
/2-240)+90,(getHeight()/2-343)+400,paint);
              canvas.drawLine((getWidth()/2-240)+90, (getHeight()/2-343)+100,(getWidth()
/2-240)+390,(getHeight()/2-343)+100,paint);

            //表題の表示
            paint.setColor(Color.BLUE);
            paint.setTextSize(22.0f);
            canvas.drawText("",(getWidth()/2-240)+60,(getHeight()/2-343)+65, paint);
             canvas.drawText("【ｙ＝ａ（ｘ－ｂ） ＋ｃ のｸﾞﾗﾌの平行移動】", (getWidth()
/2-240)+30-2-1,(getHeight()/2-343)+65, paint);
             canvas.drawText("2", (getWidth()/2-240)+195+2-5+1+1, (getHeight()/2-343)+55,
paint);

            //目標の提示
            paint.setColor(Color.BLUE);
            paint.setTextSize(19.0f);
            canvas.drawText("y ＝ a(x － b)^2 ＋ c のｸﾞﾗﾌは、y ＝ ax^2 のｸﾞﾗﾌを",(getWidth()
```

```
/2-240)+48,(getHeight()/2-343)+440, paint);
        //canvas.drawText("2", (getWidth()/2-240)+141-5, (getHeight()/2-343)+430, paint);
        //canvas.drawText("2",(getWidth()/2-240)+331-13,(getHeight()/2-343)+430, paint);
        canvas.drawText("ｘ軸方向に＋ｂだけ、ｙ軸方向に＋ｃだけ、平行",(getWidth()
/2-240)+30,(getHeight()/2-343)+465, paint);
         canvas.drawText("移動したものであることを観察してみよう。", (getWidth()
/2-240)+30,(getHeight()/2-343)+490, paint);

        //ａ、ｂ、ｃ の値の表示
        paint.setColor(Color.BLUE);
        paint.setTextSize(21.0f);
         canvas.drawText("ａ ＝"+MainActivity.n1, (getWidth()/2-240)+120, (getHeight()
/2-343)+90, paint);
         canvas.drawText("ｂ ＝"+MainActivity.n2, (getWidth()/2-240)+210, (getHeight()
/2-343)+90, paint);
         canvas.drawText("ｃ ＝"+MainActivity.n3, (getWidth()/2-240)+300, (getHeight()
/2-343)+90, paint);

        //説明の表示
        paint.setColor(Color.BLACK);
        paint.setTextSize(19.0f);
        canvas.drawText("※ 画面ﾀｯﾁで、ｙ＝ａｘ^2 のｸﾞﾗﾌが動きます。", (getWidth()
/2-240)+30,(getHeight()/2-343)+545, paint);
        //canvas.drawText("2",(getWidth()/2-240)+231-13,(getHeight()/2-343)+535, paint);
        canvas.drawText("※ もう一度タッチすると止まります。",(getWidth()/2-240)+30,
(getHeight()/2-343)+570, paint);
        canvas.drawText("※ 更にタッチすると初期化されます。",(getWidth()/2-240)+30,
(getHeight()/2-343)+595, paint);
        canvas.drawText("※ 画面が暗くなったらタイトルバーをタッチ！", (getWidth()
/2-240)+30,(getHeight()/2-343)+620, paint);
        paint.setColor(Color.BLUE);
        paint.setTextSize(19.0f);
          canvas.drawText("Copyright(C) K.Niwa 2019.11.1", (getWidth()/2-240)+100,
(getHeight()/2-343)+650, paint);                    //作者・作成年月の表示

        //識別子が１のとき、グラフが平行移動する
        if (flag==1) {
            if (MainActivity.n2<=0 && MainActivity.n3<=0){
                if (a>=MainActivity.n2) {
                    a=a-0.1;
```

```
            }
            else if（b>=MainActivity.n3）{
                b=b-0.1;
            }
        }
        else if（MainActivity.n2>=0 && MainActivity.n3>=0）{
            if（a<=MainActivity.n2）{
                a=a+0.1;
            }
            else if（b<=MainActivity.n3）{
                b=b+0.1;
            }
        }
        else if（MainActivity.n2>=0 && MainActivity.n3<=0）{
            if（a<=MainActivity.n2）{
                a=a+0.1;
            }
            else if（b>=MainActivity.n3）{
                b=b-0.1;
            }
        }
        else if（MainActivity.n2<=0 && MainActivity.n3>=0）{
            if（a>=MainActivity.n2）{
                a=a-0.1;
            }
            else if（b<=MainActivity.n3）{
                b=b+0.1;
            }
        }

        invalidate（）; //再描画する（clear & goto onDraw） そして、この行へ戻ってくる。
        //invalidate（）は、onDraw メソッドの中にも記述できる。
        //invalidate（）は、繰り返し処理に利用できる。
        //もちろん、onTouchEvent ソッドの中にも記述できる。
    }
    //識別子が２のとき、グラフが停止する
    else if（flag==2）{
        //何も変化を加えない
    }
```

```
        //識別子が０のとき、グラフを初期化する（元の位置に戻す）
        else if（flag==0）{
            a=0; //初期化する
            b=0; //初期化する
            invalidate()； //再描画する（clear & goto onDraw）　そして、この行へ戻ってくる。
        }

    }//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

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

    }//public boolean onTouchEvent(MotionEvent event)

}//public class MyTwokansu 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.mytwokansu.MyTwokansu
            android:id="@+id/myfview1"
            android:layout_height="match_parent"
            android:layout_width="match_parent"/>

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

［３］MainActivity.java

```
/*
-----------------------------------------------------------------------
          ２次関数のグラフの平行移動
            Android 4.4（Kit Kat）
       Copyright(C) K.Niwa 2019.12.13
-----------------------------------------------------------------------
*/

package jp.kiyo.wuena.mytwokansu;

//ライブラリを読み込む
import android.app.Activity;
import android.os.Bundle;
import android.text.Editable;
import android.widget.LinearLayout;
import android.view.*;
import android.view.View.OnClickListener;
import android.widget.*;

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

```
public class MainActivity extends AppCompatActivity {

    TextView text0;          //TextView 型で宣言する

    TextView text1;          //TextView 型で宣言する
    EditText edit1; //EditText 型で宣言する

    TextView text2;          //TextView 型で宣言する
    EditText edit2; //EditText 型で宣言する

    TextView text3;          //TextView 型で宣言する
    EditText edit3; //EditText 型で宣言する

    Button button1; //Button 型で宣言する

    static int n1;   //y=a(x-b)^2+c の  a の値
    static int n2;   //y=a(x-b)^2+c の  b の値
    static int n3;   //y=a(x-b)^2+c の  c の値



    /*
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
     */



    @Override
    public void onCreate(Bundle savedInstanceState) {

        super.onCreate(savedInstanceState);
```

```
        LinearLayout layout = new LinearLayout(this);
        layout.setOrientation(LinearLayout.VERTICAL);

        text0 = new TextView(this);
        text0.setText("　２次関数　ｙ＝ａ（ｘ-b)^2 ＋ｃ　について");
        layout.addView(text0);

        text1 = new TextView(this);
        text1.setText("　整数 a を入力！（ ただし、－３≦ａ≦３　ａ≠０)");
        layout.addView(text1);

        edit1 = new EditText(this);
        layout.addView(edit1);

        text2 = new TextView(this);
        text2.setText("　整数　ｂ　を入力！（ただし、－７≦ｂ≦７)");
        layout.addView(text2);

        edit2 = new EditText(this);
        layout.addView(edit2);

        text3 = new TextView(this);
        text3.setText("　整数　ｃ　を入力！（ただし、－１０≦ｃ≦１０)");
        layout.addView(text3);

        edit3 = new EditText(this);
        layout.addView(edit3);

        button1 = new Button(this);
        button1.setText("click");
        layout.addView(button1);
        button1.setOnClickListener(new MyBtnClickAdapter());

        setContentView(layout);
    }//public void onCreate(Bundle savedInstanceState)
```

```
class MyBtnClickAdapter implements OnClickListener {

    public void onClick(View view) {
        Editable ed1 = edit1.getText();
        Editable ed2 = edit2.getText();
        Editable ed3 = edit3.getText();

        try {
            n1 = Integer.parseInt(ed1.toString());
            if (n1<-3 || n1>3) {
                n1=1;
            }
        } catch (NumberFormatException e) {
            n1=0;
        }

        try {
            n2 = Integer.parseInt(ed2.toString());
            if (n2<-7 || n2>7) {
                n2=0;
            }
        } catch (NumberFormatException e) {
            n2=0;
        }

        try {
            n3 = Integer.parseInt(ed3.toString());
            if (n3<-10 || n3>10) {
                n3=0;
            }
        } catch (NumberFormatException e) {
            n3=0;
        }

        setContentView(R.layout.activity_main);    //res フォルダの layout フォルダの
main.xlm に移動する

    }//public void onClick(View view)

}//class MyBtnClickAdapter implements OnClickListener
```

```
} //public class MainActivity extends AppCompatActivity
```