```java
/*
    I will mark ** if penalty imposed

    1. No images displayed from my sample testing images
         (-0.5, non refundable!!)
    2. Class Country (-0.5)
         instance variables
         constructors
         meaningful methods
         toString  method
    3. Main Frame class (Total -2 mark)
         : GUI (-0.5)
         : ranking (-0.5, if reverse -0.2)
         : event driven (-1)  ==> ranking issue on button -0.3 **
    4. Driven of tasks (Total -2 marks)
         : Frame: not able to close -0.2
         : Rank: should be clicked one by one -0.2
         :  button label should be extracted from toString -0.3 **

    Other penalty
    : compile time errors (-3.5)
    : Not able to execute due to image file names, for example
          not absolute path, stored in your directory,
          getResource.getClass issue, etc (non refundable) -0.5
    : no declaration (-0.3)
    : run time crashed (-1)

    6.4 / 7
    Marks allocated subject to demo
    - No demo -2.5
    - Demo no images -1
*/

// Full Name: Eldwin Koh Jun Hao
// Full Time
// Tutorial Group: T04
// JDK version: 11.0.20
// Declaration: The assignment is my own work and I have not passed my
// program to my friends.I am willing to accept whatever penalty given to me.
// File name: EldwinKoh_131_A3.java

import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.Icon;
import javax.swing.ImageIcon;
import javax.swing.JOptionPane;

import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import java.awt. *;
import java.util.*;


class Olympic
{
    private static final int NO = 5;
    private String country;
    private double[] score = new double[NO];
    private int rank;

//  Default Constructor
    public Olympic(String country)
    {
        this.country = country;
        processScores();
    }

    // Copy Constructor
    public Olympic(Olympic oly)
    {
        this.country = oly.country;
    }

//  generate some scores (upon 100) for each judge
    public void processScores()
    {
        for (int i = 0; i < NO; i++)
        {
            score[i] = (double)(Math.random() * 100.00);
        }
    }

    //  calculate the sum of the elements in the score array
    public double totalScores()
    {
        double sum = 0.0;

        for (double s : score) {
            sum += s;
        }

        return sum;
    }

    // Mutator methods
    public void set(int rank)
    {
        this.rank = rank;
    }

    // Accessor methods
    public int getRank()
    {
        return rank;
    }

    public String getName()
    {
        return country;
    }

    public double[] getScoreArray()
    {
        return score;
    }

    // toString method
    public String toString()
    {
        return String.format("Rank %d: %s(%.2f)", getRank(), getName(),
totalScores());
    }
}

class OlympicFrame extends JFrame implements ActionListener
{
    private JButton[] jbArray;

    //  String of Array consisting countries
    private final String[] countryArray = {"USA", "SPAIN", "CHINA", "JAPAN",
"ITALY",
                                            "GERMANY", "FRANCE", "BRAZIL",
"NETHERLAND",
                                            "POLAND", "RUSSIA", "UKRAINE"};
    private ArrayList<Olympic> alist = new ArrayList<Olympic>();

    public OlympicFrame()
    {
        super ("Olympic 2023");
        setLayout(new GridLayout(4, 3));    //Assign GridLayout to 4 columns 3 per row

        constructAList();   //Call upon method to construct Country in contryArray

        for (int x = 0; x < alist.size(); x++)
        {
            double score = getScores(alist,alist.get(x).getName());
            int rank = getRank(alist.get(x).getScoreArray(),score);
            String country = getCountry(alist,rank);
            System.out.println(country);
        }

        // Initialize the array
        jbArray = new JButton[countryArray.length];

        // Initialize each of the buttons with imageIcon and name
        for (int x = 0; x < jbArray.length; x++)
        {
            jbArray[x] = new JButton(alist.get(x).getName());
            Icon image = new ImageIcon(String.valueOf(x + 1) + ".jpg");
            jbArray[x].setIcon(image);
        }

        // Add buttons to frame
        for (JButton jb : jbArray)
            add(jb);

        // Register the event to each buttons
        for (JButton jb : jbArray)
            jb.addActionListener(this);
    }

    private void constructAList()
    {
        for (int i = 0; i < countryArray.length; i++)
        {
            alist.add(new Olympic(countryArray[i]));    //Create new Olympic Object with String country name
        }
    }

    // Returns the country rank
    private int getRank(double[] scoreArray, double d)
    {
        int rank = 1;  // Start the rank from 1

        for (int i = 0; i < scoreArray.length; i++) {
            if (d > scoreArray[i]) {
                rank += 1;
            }
        }
        return rank;
    }

    // Constructs and return a string that you can display this string in the panel
    private String getFinalRanking()
    {
        Collections.sort(alist, Comparator.comparingDouble(Olympic::totalScores).reversed());   //Return country rank from highest score to lowest

        int ranking = 1;

        for (Olympic oly : alist) {
            oly.set(ranking++);
        }

        StringBuilder str = new StringBuilder("FINAL RANKING\n");

        for (Olympic oly : alist) {
            str.append(oly.toString()).append("\n");
        }
        return str.toString();
    }

    // Returns the country name that has a rank n
    private String getCountry(ArrayList<Olympic> alist, int n)
    {
        String country = "";

        for (int i = 0; i < alist.size(); i++)
        {
            if(alist.get(i).getRank() == n)
            {
                country = alist.get(i).getName();
            }
        }

        return country;
    }

    // Returns the total scores of a country
    private double getScores(ArrayList<Olympic> alist, String name)
    {
        double totalScore = 0.0;

        for(int i = 0; i < alist.size(); i++)
        {
            if(alist.get(i).getName().equals(name))
            {
                alist.get(i).processScores();
                totalScore = alist.get(i).totalScores();
            }
        }
        return totalScore;
    }

    @Override
    public void actionPerformed (ActionEvent e)
    {
        for (int i = 0; i < jbArray.length; i++) {
            if (e.getSource() == jbArray[i]) {
                int rank = alist.get(i).getRank();
                String country = alist.get(i).getName();

                // Update the text of the clicked button
                jbArray[i].setText(country + " ==> Rank: " + rank);

                ImageIcon trophy = new ImageIcon("trophy.jpg");
                Image image = trophy.getImage();
                Image newTrophy = image.getScaledInstance(50, 70, java.awt.Image.SCALE_SMOOTH);
                trophy = new ImageIcon(newTrophy);

                // Note we display an image in JOptionPane
                JOptionPane.showMessageDialog(null, getFinalRanking(), "Olym
2023",
                        JOptionPane.PLAIN_MESSAGE, trophy);
            }
        }

    }
}

public class EldwinKoh_131_A3 {
    public static void main(String[] args) {
        OlympicFrame aFrame = new OlympicFrame();
        aFrame.setSize (900, 700);
        aFrame.setVisible (true);
        aFrame.setDefaultCloseOperation (JFrame.EXIT_ON_CLOSE);
    }
}
```