CS 142 Assignment 3
Stars and Stripes Forever

In this assignment, you will **write a class** `StarsAndStripes` that can draw a generalized United States flag. You will fill in the contents of two methods: `drawFlag` and `drawStar`.

Restrictions:

Only call these methods with `java.awt.Graphics`: `fillRect`, `drawLine`, `setColor`.

Only use these colors: `Color.red`, `Color.white`, `Color.blue`.

To reduce confusion and differences in calculation, only perform math with **`int`**, not with **`double`**.

Requirements:

Write the following method which draws a US flag representation with the given number of stars and stripes starting at the given (x, y) coordinate (upper left corner in Java's coordinate system) and of the given width and height. Note that the top-right corner and the width and height are variables. The flag does not have to start at (0,0), and does not have to fill the whole graphics window.

```
public static void drawFlag(int stars, int stripes, java.awt.Graphics g,
                            int x, int y, int width, int height)
```

Your `drawFlag` method should follow the following procedure so that it can be properly tested:

First, draw a filled white rectangle (called the *base*) of the given `x`, `y`, `width`, and `height`.

Second, draw filled red rectangles (*red stripes*) across the width of the *base*. The first rectangle should begin in the *base*'s upper left. Each *red stripe* height (except possibly the last one) should be (`height` divided by `stripes`) **rounded down**. Then, skip down according to the *red stripe* height which will expose the *base*. Draw a number of *red stripes* equal to half of `stripes` **rounded up**.

If `stripes` is odd, the final *red stripe* height should touch the bottom pixels of the underlying *base* rectangle, not beyond. This means when stripes is odd (so the last stripe is red), the last stripe may be taller than the others.

Third, draw a filled blue rectangle (*starfield*). The *starfield*'s height should be equal to the *red stripe* height times the number of *red stripes*. Given this height, the *starfield*'s width should be proportional to the flag (*starfield* height × *base* width / *base* height). You should **round down** for the width calculation.

Finally, draw the white stars over the *starfield*. Do this by writing the `drawStar` method described below, then calling it several times (ideally in a loop) to form a rectangular grid of stars.

For simplicity, we'll assume that the number of stars is factorable into pair of numbers that make a "nice" grid: one where the number of columns is greater than the number of rows, but is less than two times the number of rows (e.g. (rows x columns) = 4x6, 6x8, or 3x5).

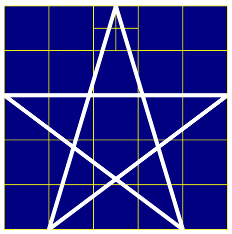
For the given number of `stars`, your code should determine the correct number of rows and columns.

When drawing stars, each star should be the same size, the stars should be as large as they can be within their grid while still fitting in the starfield, and <u>if there is leftover space, try to center the stars within the starfield.</u>

For a "regularly shaped" flag, there will likely be extra space to the sides, and the stars will fill the starfield from top to bottom. But since the graphics window is resizable, your code should also account for flags that are tall and skinny, so there is extra space in the top and bottom of the starfield, and the stars fill it out from side to side.

Note that due to line thickness, the edges of the lines may just barely go beyond the starfield, but the lines' coordinates should stay within it.

```
public static void drawStar(java.awt.Graphics g, int x, int y, int size)
```



Each star will have its own upper left (`x`, `y`) for its bounding box, as well as an equal width and height (called `size`). Draw it as line segments (with `g.drawLine`) resembling the diagram to the left: from the bottom 1/5 of the way from the left to 2/5 of the way from the top on the right, straight across to the left, down to the bottom 1/5 of the way from the right, up to the top center, and back down to the starting point. This will allow the test program to separately test your star drawing procedure and make sure that it works correctly.

Each calculation to draw the star line coordinates should round down. **Note that size denotes both star width and star height since they are equal.**