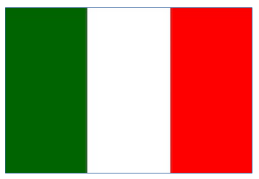
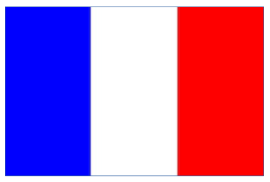
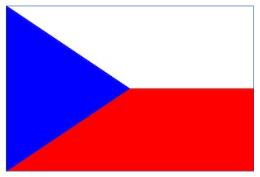
Lab 3 – Picture This!

1. In class we wrote a program that drew a picture of a "Japanese flag". Write a program called **flag()** that draws the flag of a country of your choice. Some possibilities are shown below, but feel free to choose a different country (some are more challenging than others!). Make your drawing window 600 pixels wide and 400 high.

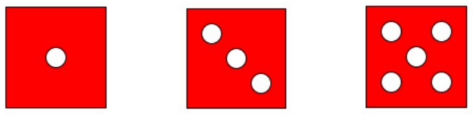

Italy


France

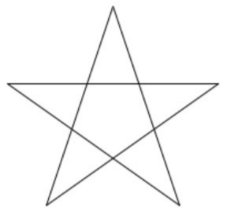
Germany


Czech Republic

2. Write a program that creates a graphics window and draws a red square of width and height 100 in the exact center of the window, with a white circle of radius 10 centered in the square, representing a die of value 1 (left image below). Next, add two more white circles to your die (middle image). Finally, add two more circles to make the number 5 (right image).



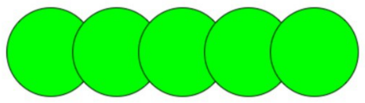
3. Write a program called **star()** that draws a 5-pointed star with vertices located at points (200,100), (105,170), (140,280), (260,280), and (295,170). Hint: this will be easier if you first create five Point objects called `p1`, `p2`, `p3`, `p4`, and `p5`, and then use them to draw either five individual Lines, or a single Polygon.



4. An archery target consists of a central circle of yellow surrounded by concentric rings of red, blue, black, and white. Each ring has the same "width", which is the same as the radius of the yellow circle. Write a program called **target()** that draws such a target. For simplicity, use a 500 $\times$ 500 window, and make each ring 50 pixels wide, so the yellow circle's diameter will be 100 pixels, the red circle's diameter 200 pixels, etc. Hint: objects drawn later will appear on top of objects drawn earlier.

5. Write a program called **design()** that draws an abstract artistic design, some of whose elements are determined randomly (by using Python's `random` module functions). This could include randomly choosing the positions of points, lines, or other shapes, or randomly choosing colors, or both. As a reminder, the function `random.choice(`*`list`*`)` can be used to pick an element from a list at random. To pick a random number in the range *a* to *b* (excluding *b*), you can use `random.randrange(`*`a, b`*`)`. Don't forget to include the line `import random` at the top of your file.

6. Write a program called **bug()** that draws the body of a green "caterpillar" as shown below. Use a for-loop to draw the five circles, starting from an initial center point (*x*, *y*) and increasing the *x* value by a fixed amount on each loop cycle. The color shown is `"green1"`. (Optional: see if you can add eyes to your caterpillar.)

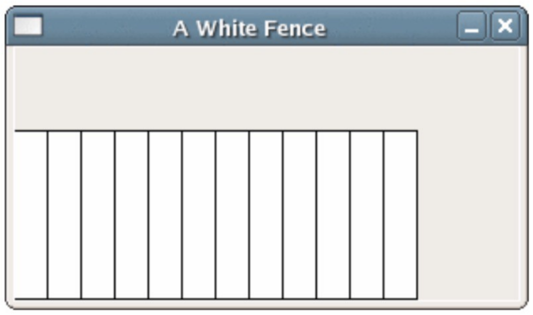


7. We can make a window respond to mouse clicks by calling `win.getMouse()`, which returns a Point object indicating where the user clicked. As an example, try out the `click.py` program available under Labs on our class web page. Modify the program so that each time the user clicks the mouse, the program draws a new red circle of radius 20 centered on the mouse click.

8. Write a program called **move()** that draws a colored circle at the upper left corner of the graphics window and then moves it a fixed amount diagonally toward the bottom right corner each time the user clicks the mouse. You can use the circle's move(*dx*, *dy*) operation with a small positive value for *dx* and *dy* to move the circle.

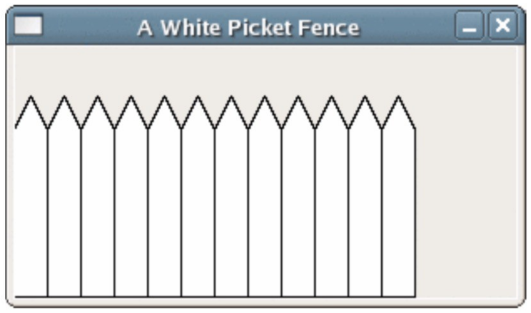
9. Write a program called **fence()** that draws a white fence. The program should ask the user for the width and height (in pixels) of a single fence plank, as well as the total number of planks in the fence. It should then draw the fence as a row of white rectangles along the bottom of the graphics window, starting from the left side. Hint: keep track of the coordinates (*x1*, *y1*) and (*x2*, *y2*) of the lower left and upper right corners of a rectangle, and use a for-loop to increase *x1* and *x2* by one plank width on each loop cycle. For example:

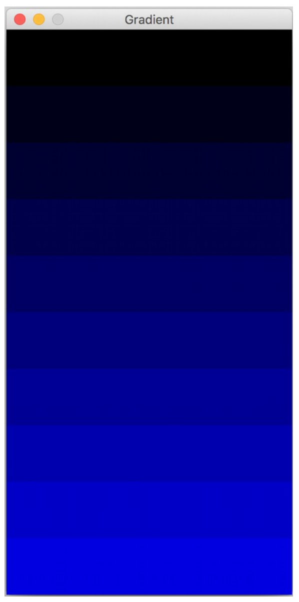
```
>>> fence()
Enter width of a single plank: 20
Enter height of a single plank: 100
Enter number of planks: 12
```



10. Modify your fence-drawing program so that it draws a white picket fence, as shown below. Hint: represent the plank tip as an additional coordinate (*x3*, *y3*), and increase *x3* by one plank width on each loop cycle. Instead of drawing a plank as a rectangle, draw it as a Polygon with five vertices.

```
>>> fence()
Enter width of a single plank: 20
Enter height of a single plank: 100
Enter number of planks: 12
```





11. Read about the `color_rgb` function on page 121 of your textbook and experiment with creating color gradients. Write a program called **gradient()** that uses a loop to fill the graphics window with multiple horizontal (or vertical) rectangular stripes, each one a slightly darker (or lighter) shade than the one before. An example is shown to the right: Your program should ask the user for the desired number of stripes, and then calculate the appropriate height (or width) of each stripe based on the size of the window.