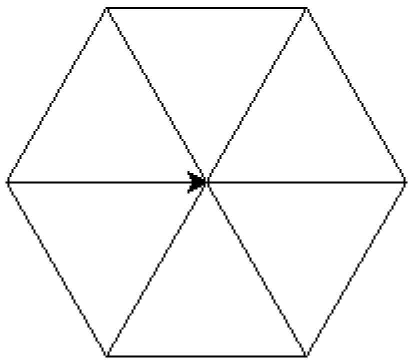
Name ______________________________

CPADS Assignment #2

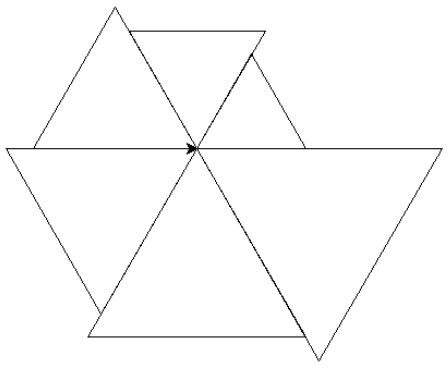
1. Develop a **strategy** with pseudocode that creates the following shapes, each of which must use a loop to perform their function:
    a. A regular hexagon using the **drawEquiTriangle** function, which draws equilateral triangles.
    b. A pinwheel hexagon using the **drawEquiTriangle** function.
    c. A 4-pointed star, using the **drawSquareFromCenter** function and the **drawIsoTriangle** function, which draws isosceles triangles for the specified **base** and **height**.
    d. A "constellation" of 4-pointed stars, equidistant from a central star, and equally spaced around the central star, by calling **drawStar**.

    Hint: ***Your strategy should have four parts – one for each of the three functions and one for the portion of the main program that is drawing the constellation of stars. Consider what information the functions will need in order to accomplish their specific tasks. Be sure to leave the cursor in the same location and orientation in which it began.***

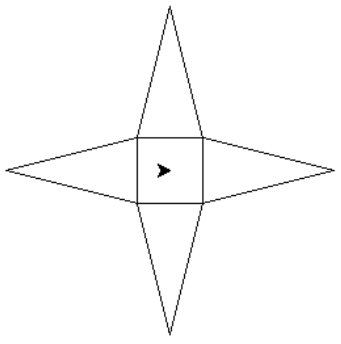
a.



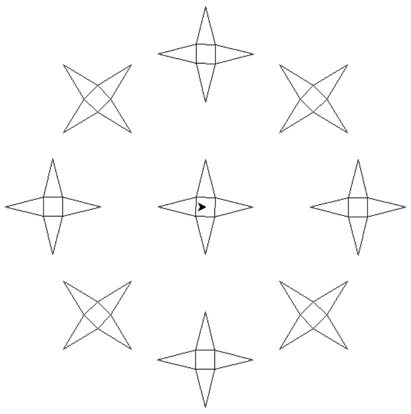
b.



c.



d.



CS100 Fall 2017

Name ____

Write your strategies for the four parts here:

Name ______________________________

2. **Download the file triangles.py from the course webpage**

   https://ycpcs.github.io/cs100-fall2017/assign/src/triangles.py

   Open the file with PyCharm. USING YOUR STRATEGY FROM PART 1, complete the program by adding code for each of the TODOs shown in the program.

   - **TODO 1:** Convert the given **drawSquareFromCenter** function so that it uses a loop to draw the four sides of the square.
   - **TODO 2:** Convert the given **drawEquiTriangle** function so that it uses a loop to draw the 3 sides of the equilateral triangle.
   - **TODO 3:** Complete the **drawRegHexagon** function so that it uses a loop to draw a regular hexagon by calling the **drawEquiTriangle** function from within the loop. Hint: Consider how this problem is similar to drawing a pinwheel of squares that all have the same size.
   - **TODO 4:** Complete the **drawPinwheelHexagon** function so that it uses a loop to draw a pinwheel hexagon. Use the loop counter to calculate the size of each of the triangles that make up the hexagon. The size of each triangle should increase by ***size/5*** over the previous triangle such that the last triangle will be twice the size of the first triangle. Hint: Consider the similarity between this function and the **drawRegHexagon** function above – and what changes you would need to make to **drawRegHexagon** in order to create **drawPinwheelHexagon**.
   - **TODO 5:** Implement the **drawStar** function so that it draws a central square (using **drawSquareFromCenter**), and then draws 4 isosceles triangles (using **drawIsoTriangle**), with their bases aligned to each side of the central square, as shown. The base of each isosceles triangle will be the same as the size specified for the square.
   - **TODO 6:** In **main**, *call* the new **drawStar** function, passing a ***turtle***, and the user input for ***size*** and ***height*** to **drawStar**.
   - **TODO 7:** Complete **main** by implementing a loop that draws a "constellation" of stars around the central star that was drawn in **TODO 6**. The center of each star should be positioned ***radius*** pixels from the origin, and the stars themselves should be positioned at regular intervals around the central start, based on the user input for ***stars***.

More detail is given in the provided code with each of the TODO's.

Make sure to provide appropriate and sufficient comments for all of the code that you modify or add.