

# A 3

---

**Due** Sep 21 by 11:59pm    **Points** 100    **Submitting** a file upload    **File Types** py, jpg, jpeg, and ipynb  
**Available** Sep 11 at 12am - Sep 21 at 11:59pm 11 days


---

This assignment was locked Sep 21 at 11:59pm.

## Turtle Graphics

*Please don't view this assignment in Safari: Canvas is having trouble providing Safari with graphics it can display.*

This assignment will have you create a sequence of Python programs to draw Turtle Graphics. You will be writing python programs, rather than working in a Notebook.

Please hand in your programs and a screen shot of each creation saved as a .jpg file. Please include your program in the screenshot, as I have done [here](https://canvas.harvard.edu/courses/95534/files/12801667/download?download_frd=1)  ([https://canvas.harvard.edu/courses/95534/files/12801667/download?download\\_frd=1](https://canvas.harvard.edu/courses/95534/files/12801667/download?download_frd=1)) . You may not get it all in: get the important parts. You should submit 8 files:

pentagram.py pentagram.jpg

grid.py grid.jpg

honeycomb.py honeycomb.jpg

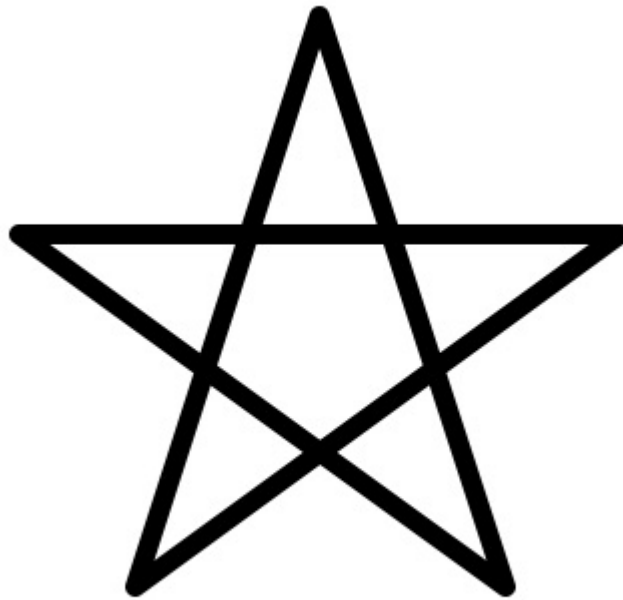
freestyle.py freestyle.jpg

The webpage below describes some options for the Turtle. For example, I have hidden the turtle in the figures below. I suggest you only hide it when you are done, as it is useful to watch the turtle as you move through your program. You may also need to pick up the pen and put it down again. You can learn about these commands here:

<https://docs.python.org/3/library/turtle.html>  (<https://docs.python.org/3/library/turtle.html>)

These figures can become complex: it is best if you can locate the common parts, and put them in a function. I used multiple functions in problems 2 and 3.

# Pentagram



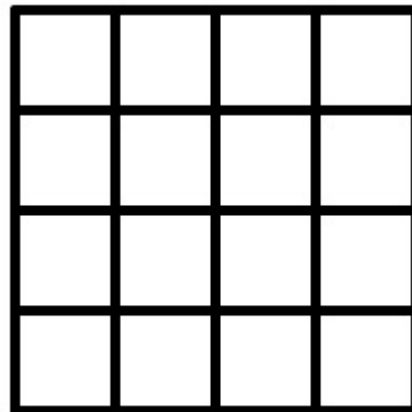
1) The Pentangle or Pentagram was used as a symbol by early Christians as well as many other religions.

Write a Python program to draw a pentangle. Your program should define a function that takes three arguments: a turtle, the edge length, and the pen width.

```
draw_pentagram(t, edgeLen, penWidth)
```

which draws a pentangle whose edges have length `edgeLen`, and whose lines have width `penWidth`. The pentagram is always drawn with one point headed North. Draw with a broad stroke.

## Grid



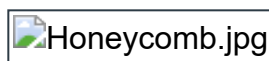
2) Use Turtle Graphics to solve the problem of drawing an  $n \times n$  set of boxes. Your program should have a function that takes three parameters: a turtle, the edge length, and the number of boxes per side.

*Hint: You could draw this as  $n \times n$  boxes, or as two sets of  $n+1$  parallel lines.*

```
draw_grid(t, edgeLen, count)
```

## Honeycomb

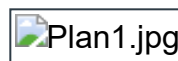
3) Use the Turtle to draw 7 hexagons in the honeycomb pattern below



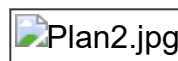
Your solution should have a function that draws a hexagon, and logic to call it 7 times.

Hint: My first solution drew a hexagon centered around a given point. This made it easy to see what I needed to do. Here are the movements that my program made between calls to draw a hexagon. I haven't included the movements to the west to start the hexagon and the movement to the east to return to the center.

Rather than move directly, I moved in a 'dog leg' which lengthened the journey but simplified computing direction and length to travel.



After I got that working, I could see simpler way to draw it, moving from the western corner of each hexagon rather than the center. This reduced the number of steps and the time the turtle spent traveling.



To prepare to draw the next hexagon, you can make all movements relative to the current position and heading, or use commands to move to a given position, and to turn to a given direction.

That is, you could `turt.forward(10)` to move relative to your current position and heading, or `turt.setpos(0,0)` to return to screen center. Likewise you can `turt.rt(60)` to turn right 60 degrees from the current heading, or `turt.setheading(90)` to point North, whatever your current heading.

## Freestyle

4) Draw something interesting using the Turtle. Use at least two features from the Turtle Library that we have not discussed, such as color, fill, or motion.

<https://docs.python.org/3/library/turtle.html> ↗ [\\_ \(https://docs.python.org/3/library/turtle.html\)](https://docs.python.org/3/library/turtle.html)

## Resources

A chapter from "How to Think Like a Computer Scientist" about the Turtle

[https://python.camden.rutgers.edu/python\\_resources/python3\\_book/hello\\_little\\_turtles.html](https://python.camden.rutgers.edu/python_resources/python3_book/hello_little_turtles.html) ↗  
[\\_ \(https://python.camden.rutgers.edu/python\\_resources/python3\\_book/hello\\_little\\_turtles.html\)](https://python.camden.rutgers.edu/python_resources/python3_book/hello_little_turtles.html)

The Anaconda distribution include a directory with interesting examples

`../anaconda3/lib/python3.8/turtledemo/`

Website with many interesting examples that might provide inspiration

<http://www.fractalcurves.com> ↗ [\\_ \(http://www.fractalcurves.com\)](http://www.fractalcurves.com)

Rubric for Assignment 3			
Criteria	Ratings		Pts
Overall Grading 1. Solves problems as described  2. Brief Comments appear in appropriate places and meaningful variable names are used.  3. Logic concise, uses Python syntax and Python abilities appropriately  4. Discretionary deductions  5. Delivery: are all the appropriate files in the right places?	<b>0 pts</b> <b>Full Marks</b>	<b>0 pts</b> <b>No Marks</b>	0 pts
Pentagonam 5 pointed star, with one point headed North. Broad lines, no cruft.	<b>25 pts</b> <b>Full Marks</b>	<b>0 pts</b> <b>No Marks</b>	25 pts
Grid Proper image. Good use of functions	<b>25 pts</b> <b>Full Marks</b>	<b>0 pts</b> <b>No Marks</b>	25 pts
Honeycomb Proper image, good use of functions.	<b>25 pts</b> <b>Full Marks</b>	<b>0 pts</b> <b>No Marks</b>	25 pts
Freestyle Interesting Image. Sensible use of functions.	<b>25 pts</b> <b>Full Marks</b>	<b>0 pts</b> <b>No Marks</b>	25 pts
Total Points: 100			