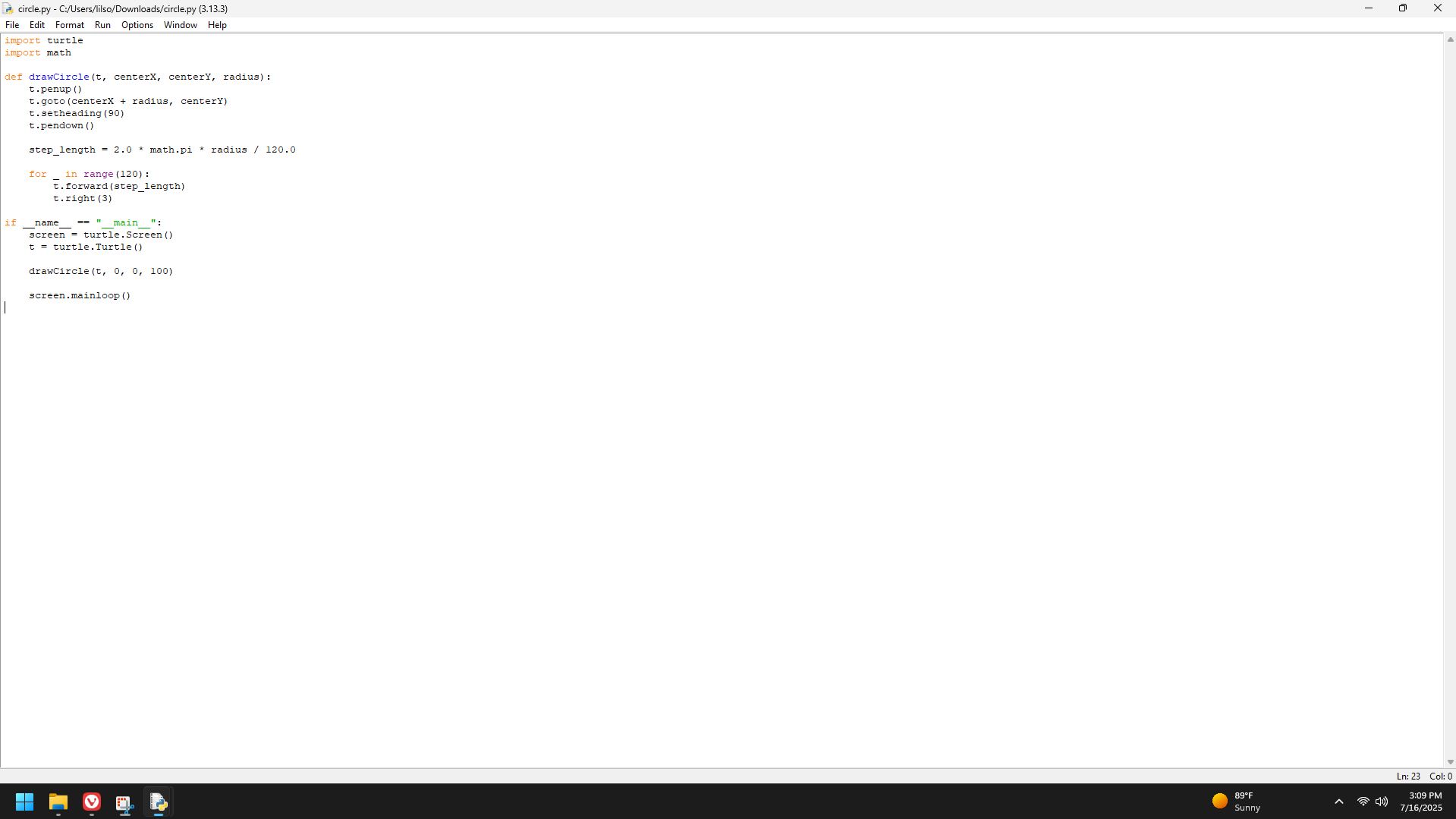
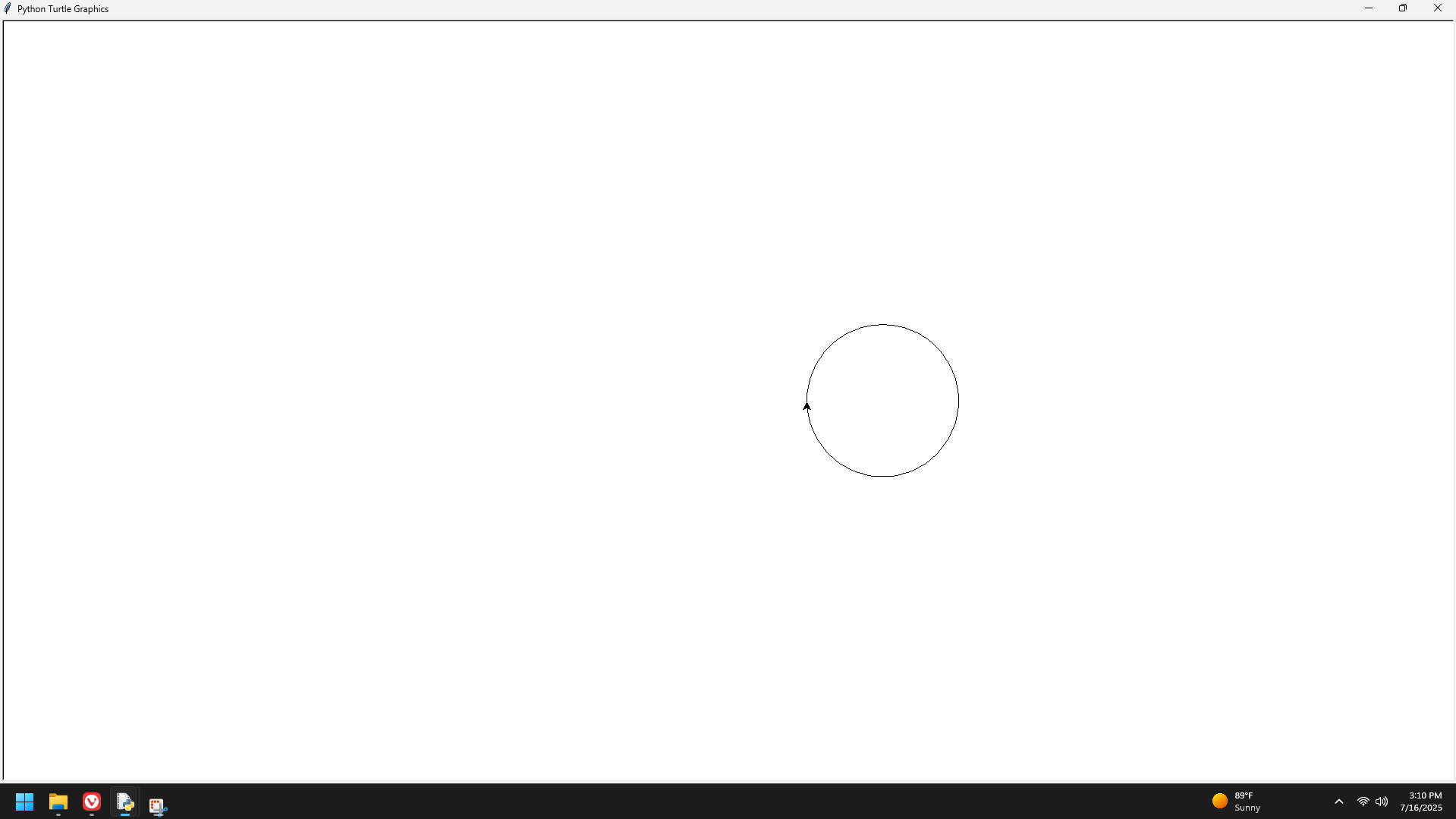
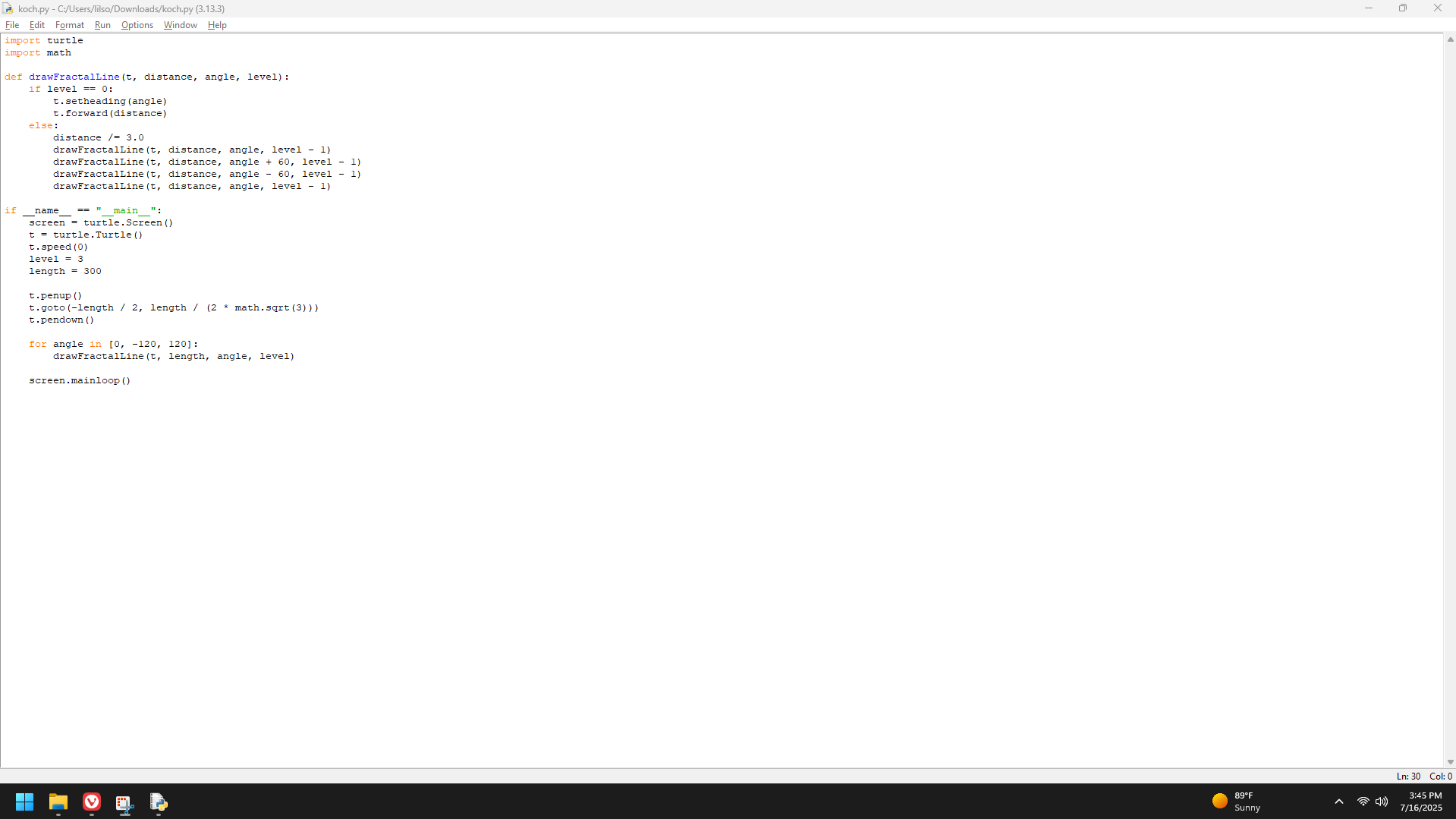
| Course Name | ITD 2313 – Script Programming |
| --- | --- |
| Instructor | Mark Pranger |
| Student Name | Weston Albright |
| Due date | 07/22/2025 |
| Grade | 95.63% |
| Grading Comments | N/A |

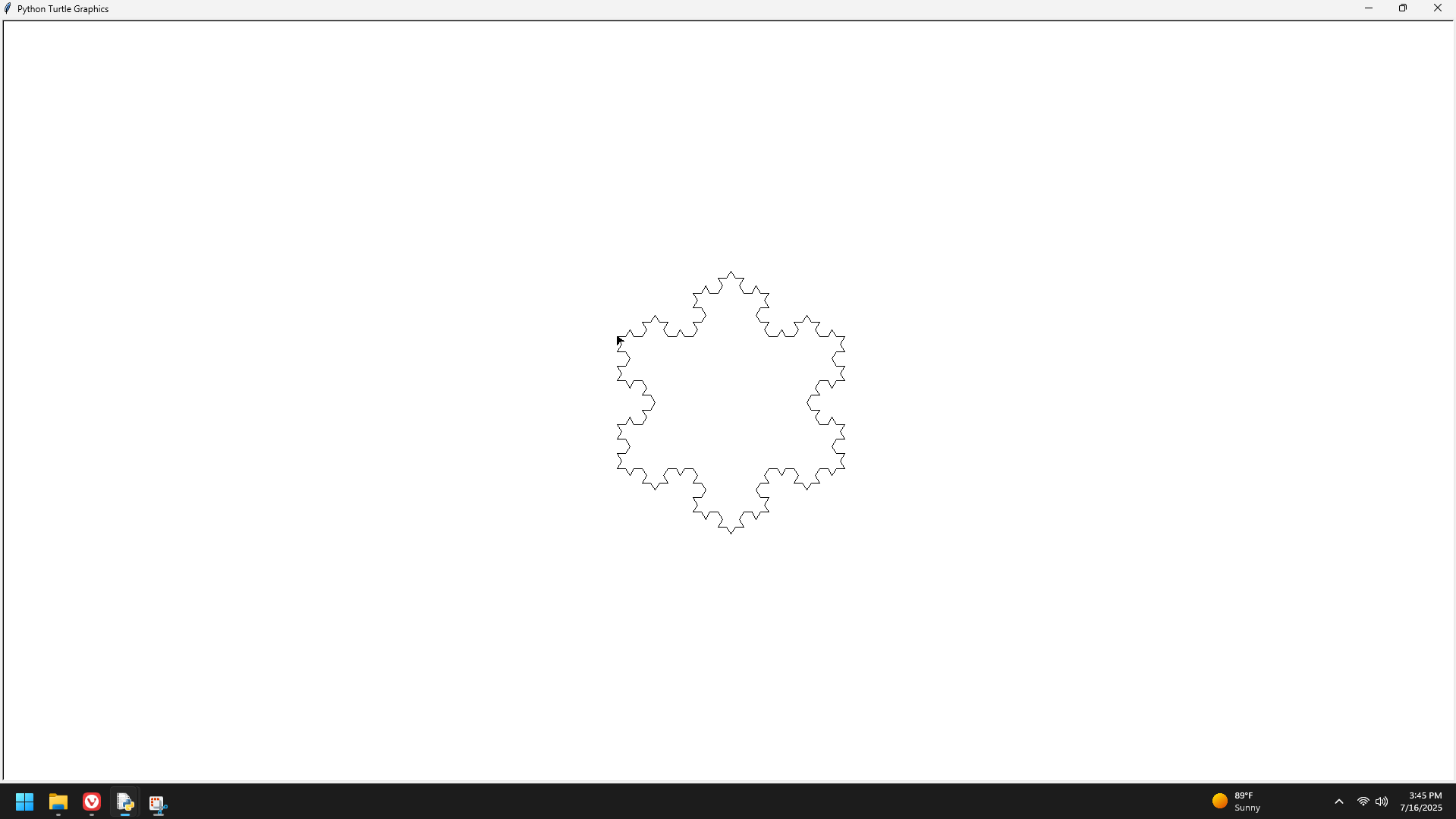
**Define a function drawCircle (in the file circle.py). This function should expect a Turtle object, the coordinates of the circle’s center point, and the circle’s radius as arguments. The function should draw the specified circle. The algorithm should draw the circle’s circumference by turning 3 degrees and moving a given distance 120 times. Calculate the distance moved with the formula 2.0\* π\* radius / 120.0.**

****

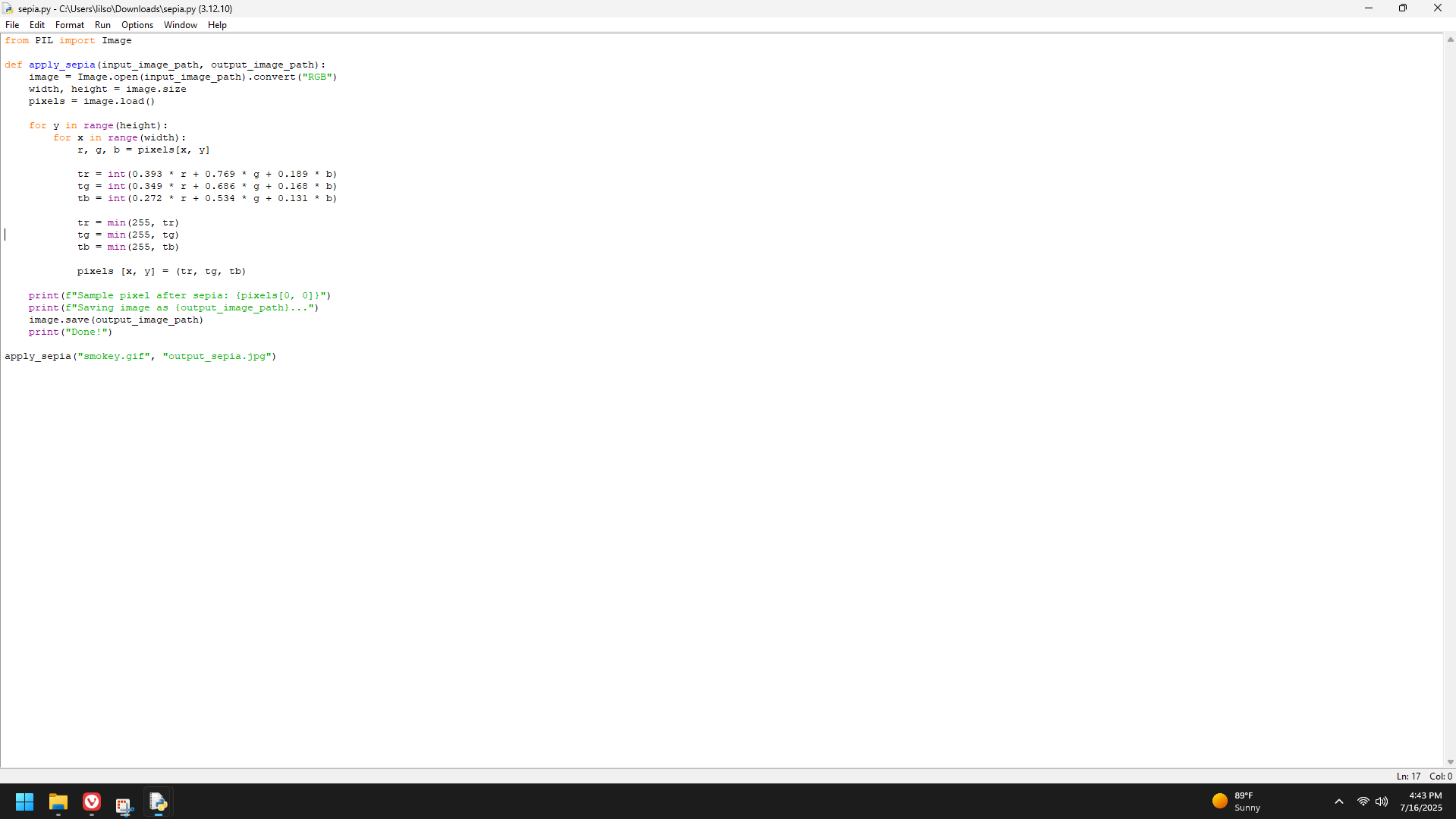
****

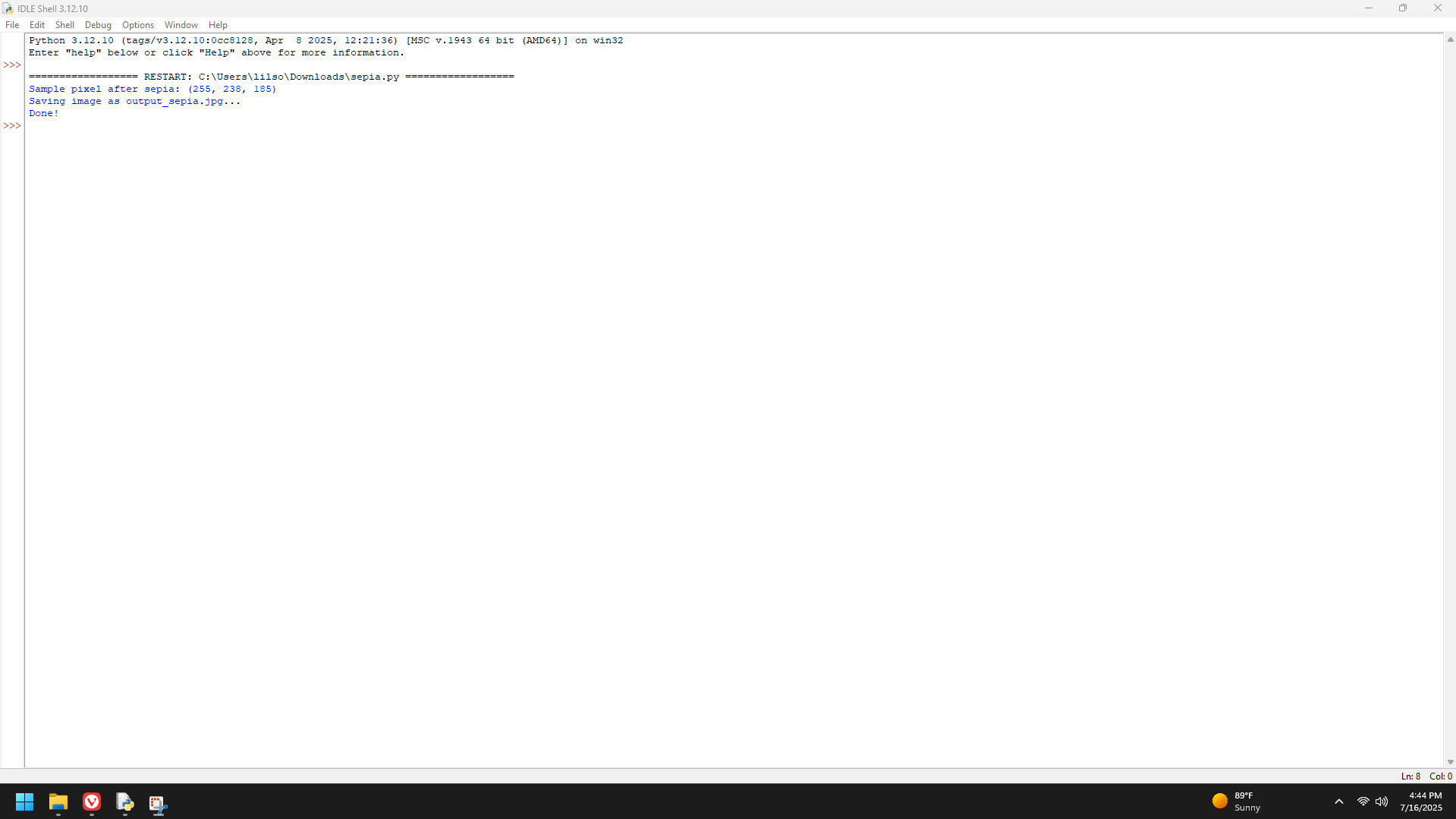
**The Koch snowflake is a fractal shape. At level 0, the shape is an equilateral triangle. At level 1, each line segment is split into four equal parts, producing an equilateral bump in the middle of each segment. Figure 8-15 shows these shapes at levels 0, 1, and 2. At the top level, the script uses a function drawFractalLine to draw three fractal lines. Each line is specified by a given distance, direction (angle), and level. The initial angles are 0, –120, and 120 degrees. The initial distance can be any size, such as 200 pixels. The function drawFractalLine is recursive. If the level is 0, then the turtle moves the given distance in the given direction. Otherwise, the function draws four fractal lines with one-third of the given distance, angles that produce the given effect, and the given level minus 1. Write a script (in the file koch.py) that draws the Koch snowflake.**

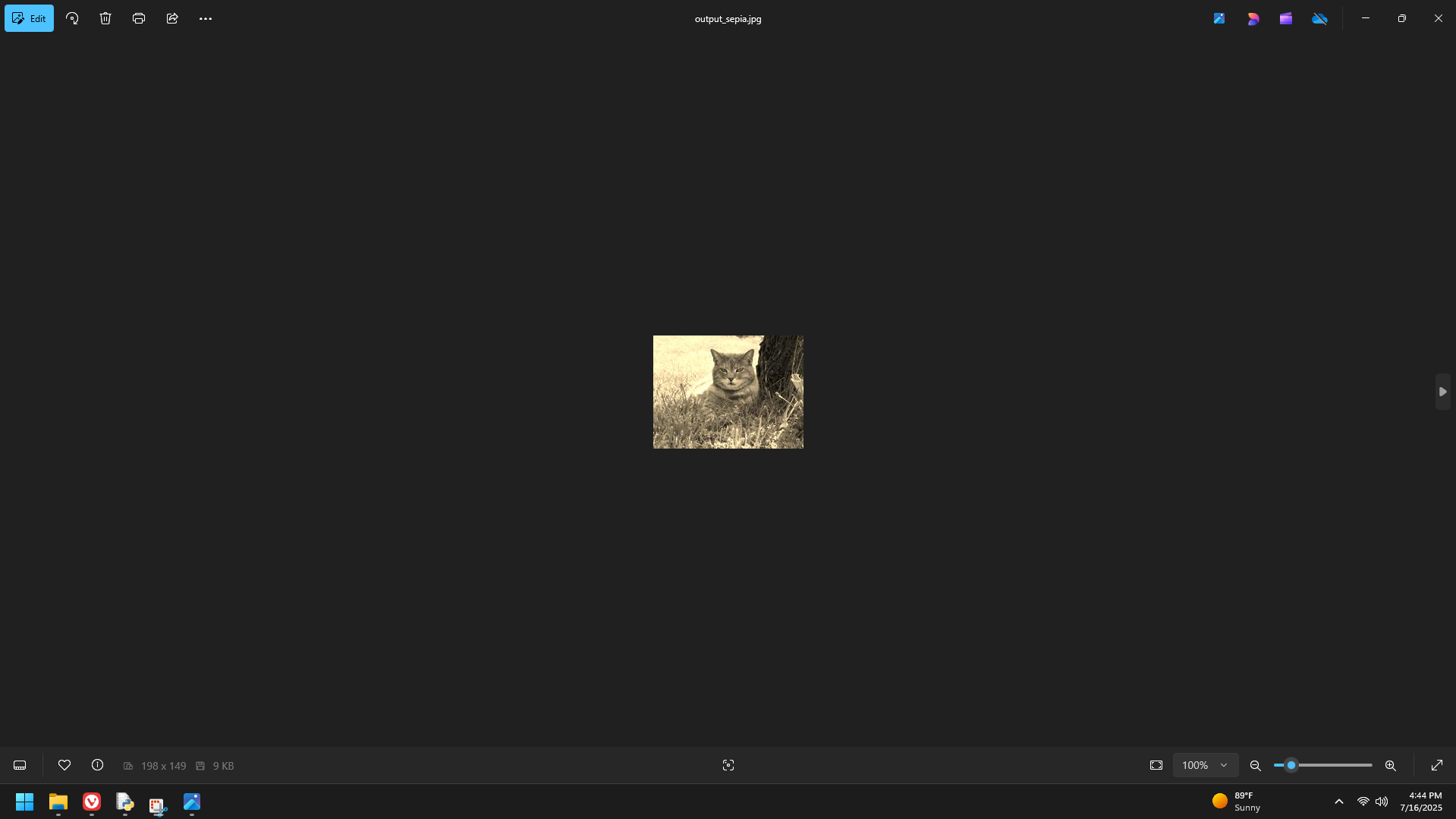
****

****

**Write and test a function named sepia (in the file sepia.py) that converts a color image to sepia. This function should first call grayscale to convert the color image to grayscale. A code segment for transforming the grayscale values to achieve a sepia effect follows. Note that the value for green does not change.**

****

****

****