**Kaleido-Spiral**

**Concepts Learned:**

* **Recursion**: The draw\_circle() function calls itself, teaching the concept of recursion and how it can be used for repetitive tasks.
* **Turtle Graphics**: Using the turtle module for drawing circles and controlling movement introduces visual programming.
* **Colour Cycling:** The itertools.cycle() function cycles through colours, teaching the use of external libraries for repeated operations.
* **Function Parameters:** The function takes several parameters (size, angle, shift, and max\_size), which control the behaviour of the drawing, teaching flexible function design.
* **Base Case**: The recursive function includes a base case (if size > max\_size), teaching students to define stopping conditions in recursion.
* **Coordinate Movement:** The use of turtle.right() and turtle.forward() to change direction and position demonstrates how to control movement in graphics.

**Key Learning Outcomes:**

* **Recursive thinking:** Understand how recursive functions work, including setting base conditions to prevent infinite loops.
* **Graphics programming:** Gain experience with turtle graphics, learning to create visually appealing designs through recursion.
* **Colour manipulation**: Use the itertools.cycle() function to repeatedly apply different colours in a sequence, introducing efficient cycling through lists.
* **Parameter control:** Learn to control graphical outputs through function parameters, enhancing flexibility in drawing operations.
* **Turtle module functions:** Practice using turtle commands like circle(), right(), and forward() to create dynamic and interactive drawings.
* **Recursion with stopping condition:** Apply recursive functions with stopping conditions to ensure proper control over iterative drawing processes.