PROBLEM 5
HYPNO SPIRAL / SUNFLOWER
(LINK) / (LINK)

The HYPNO SPIRAL code generates an animated spiral of circles, inspired by the Fibonacci pattern, using polar coordinates and smooth rotation for a hypnotic effect.

At the top, two global variables are defined: `t` (a global time variable for animation) and `rot` (which controls the rotation of the entire spiral). In `setup()`, the canvas is set to the full window with a black background, and shapes are drawn without fill, only outlines.

The main animation happens in `draw()`. Each frame, a semi-transparent black rectangle is drawn (`fill(10, 60); rect(...)`) to create a fading trail effect. The origin is moved to the center of the canvas with `translate()`. Then, the code uses `push()` and `pop()` to rotate the spiral by `rot` radians (`rotate(rot); drawSpiral();`). The variables `t` and `rot` are increased each frame, controlling both the pulsation and the spinning.

The function `drawSpiral()` contains the main loop, where for each point (`n`) up to `maxN`, the code computes the angle (using the golden angle in radians and a slight sinusoidal oscillation) and a radius that grows like `sqrt(n)`. These polar coordinates are converted to Cartesian (`x, y`), and each circle is drawn with a white outline whose thickness and diameter also oscillate in time, becoming smaller towards the outside.

The result is a clean, dynamic spiral made up of gently pulsing white circles, slowly rotating against a black background, for an elegant “hypno-spiral” visual effect.

The SUNFLOWER sketch also uses polar coordinates and the golden angle to create a spiral, as seen in the previous example.

Here, the global variable `n` counts the number of steps, and each frame in `draw()` adds one circle at a position computed from

`angle = n * radians(137.5)` and `radius = 6 * sqrt(n)`.

The `(x, y)` coordinates center each circle, and `fill(40 + n % 215, 160, 200, 180)` assigns a unique color to each.

Unlike the previous sketch, this one **does not animate or rotate** the spiral: every circle stays fixed, and `noLoop()` halts the drawing when the spiral reaches the canvas edge.

In summary, both codes use the same math, but this version builds a **static, step-by-step sunflower spiral** with changing colors and no motion.

```javascript
let n = 0; // Counter: number of steps (circles) drawn so far

function setup() {
  createCanvas(windowWidth, windowHeight); // Fullscreen canvas
  background(255);      // White background
  noStroke();           // Circles will have no outline
}

function draw() {
  let cx = width / 2;
  let cy = height / 2;

  // Calculate angle using golden angle for phyllotaxis spiral
  let angle = n * radians(137.5);
  // Radius increases like sqrt(n) to spread circles evenly
  let radius = 6 * sqrt(n);

  // Convert polar to cartesian coordinates
  let x = cx + radius * cos(angle);
  let y = cy + radius * sin(angle);

  // Assign a color based on n for a smooth gradient
  fill(40 + n % 215, 160, 200, 180);
  ellipse(x, y, 15, 15); // Draw one circle at (x, y)

  n++; // Next circle in next frame

  // Stop drawing when the spiral reaches the edge of the canvas
  if (radius > min(width, height) / 2) noLoop();
}

/*
  This sketch draws a static sunflower spiral, placing one colored circle at a time using the golden angle.
  The pattern grows outward from the center, with each circle fixed in place.
  Color and spacing are controlled by n; the process stops automatically at the canvas edge.
*/
```