

	Ingeniería en Sistemas Computacionales	Guia p5.js Unidad 1
	Dr. Juan Gabriel Loaiza	02 / 2026

p5.js

Libro didáctico estructurado por capítulos

Incluye teoría detallada, ejemplos explicados y ejercicios progresivos

Contenido

p5.js	1
CAPÍTULO 1 — Fundamentos de p5.js	5
1. ¿Qué es p5.js?	5
1.1 Estructura de un proyecto	5
1.2 La función setup()	6
1.3 La función createCanvas(w, h)	6
1.4 Concepto matemático del píxel	8
1.5 ¿Qué pasa si no usamos createCanvas()?	8
1.6 Fondo del canvas — background()	8
1.7 Variables automáticas: width y height	9
1.8 Ejemplo completo explicado	9
1.9 Ejercicios Progresivos	10
CAPÍTULO 2 — La función draw() y el ciclo de animación	11
2.1 ¿Qué es draw()?	11
2.2 Concepto Matemático: Tiempo Discreto	11
2.3 Concepto de Frame	12
2.4 La animación como función matemática	12
2.5 Concepto físico: Movimiento Rectilíneo Uniforme (MRU)	13
2.6 ¿Por qué usamos background() dentro de draw()?	14
2.7 frameRate(n)	14
2.8 noLoop(), loop(), redraw()	15
2.9 Ejemplo Mejorado: Rebote en los Bordes	16
2.10 Explicación Matemática del Rebote	16
2.11 Modelo Matemático del Rebote	17

	Ingeniería en Sistemas Computacionales	Guia p5.js Unidad 1
	Dr. Juan Gabriel Loaiza	02 / 2026

2.12 Problema Común: Centro del círculo.....	17
2.13 Ejercicios Progresivos.....	18
2.14 Conclusión Conceptual	18
CAPÍTULO 3 — Sistema de Coordenadas en p5.js	19
3.1 ¿Qué es un sistema de coordenadas?	19
3.2 Sistema de coordenadas en p5.js	19
3.3 Explicación Matemática	20
3.4 Concepto de píxel como punto discreto.....	20
3.5 Variables automáticas: width y height	21
3.6 Explicación matemática del centro	21
3.7 Ejemplo completo: Ejes centrales.....	22
3.8 Coordenadas de las esquinas.....	23
3.9 Distancia entre puntos (base geométrica)	23
3.10 Ejercicio principal del capítulo.....	24
3.11 Ejercicios Progresivos.....	24
3.12 Reflexión Matemática Final	25
CAPÍTULO 4 — Colores y Estilos en p5.js.....	26
4.1 ¿Qué es el color en computación?	26
4.2 Funciones principales de color y estilo	26
4.3 Modelos de color.....	28
4.3.1 Escala de grises	28
4.3.2 Modelo RGB.....	29
4.3.3 Modelo Hexadecimal	29
4.3.4 Modelo HSB (Hue, Saturation, Brightness)	29
4.4 Ejemplo explicado paso a paso	30
4.5 Concepto Geométrico del Rectángulo	31
4.6 Interacción entre stroke y fill	31
4.7 Ejercicio Principal	31
4.8 Explicación Matemática del Ejercicio.....	32
4.9 Ejercicios Progresivos	32
4.10 Reflexión Conceptual.....	33

	Ingeniería en Sistemas Computacionales	Guia p5.js Unidad 1
	Dr. Juan Gabriel Loaiza	02 / 2026

CAPÍTULO 5 — Figuras Geométricas en p5.js.....	34
5.1 Introducción: Geometría en programación	34
5.2 line(x1, y1, x2, y2)	34
5.3 rect(x, y, w, h).....	35
5.4 square(x, y, s)	35
5.5 circle(x, y, d)	35
5.6 ellipse(x, y, w, h)	36
5.7 triangle(x1, y1, x2, y2, x3, y3)	36
5.8 quad(x1, y1, x2, y2, x3, y3, x4, y4)	37
5.9 arc(x, y, w, h, start, stop).....	37
5.10 Modos de interpretación	37
5.11 Ejemplo Integrador: Dibujar una Casa.....	38
5.12 Análisis Geométrico del Ejercicio.....	39
5.13 Ejercicios Progresivos.....	40
5.14 Reflexión Final	40
CAPÍTULO 7 — Arcos y Ángulos.....	41
7.1 ¿Qué es un arco?.....	41
7.2 ¿Por qué p5 usa radianes?	41
7.3 Constantes útiles en p5.js.....	42
7.4 Matemática sencilla detrás de un arco	43
7.5 Ejemplo básico de arc()	43
7.6 Detalle importante: orientación del ángulo en p5.js.....	44
7.7 Ejercicio del capítulo: Reloj analógico	44
7.8 Matemática de las manecillas (lo importante)	44
7.9 Código sencillo: reloj analógico (básico)	45
7.10 Explicación matemática del ejemplo	46
7.11 Ejercicios progresivos.....	46
7.12 Cierre conceptual.....	47
CAPÍTULO 8 — Suavizado y Calidad Visual.....	47
8.1 ¿Qué es el suavizado?.....	47
8.2 Concepto Matemático del Problema.....	48

	Ingeniería en Sistemas Computacionales	Guia p5.js Unidad 1
	Dr. Juan Gabriel Loaiza	02 / 2026

8.3 ¿Qué hace smooth()?	48
8.4 ¿Qué hace noSmooth()?	49
8.5 Ejemplo Comparativo	49
8.6 Explicación Geométrica	50
8.7 Relación con Resolución	50
8.8 Cuándo usar cada uno	51
8.9 Ejercicio práctico	51
8.10 Reflexión	51
PROYECTO FINAL — Mini Proyecto Integrador	52

	Ingeniería en Sistemas Computacionales	Guia p5.js Unidad 1
	Dr. Juan Gabriel Loaiza	02 / 2026

CAPÍTULO 1 — Fundamentos de p5.js

1. ¿Qué es p5.js?

p5.js es una **librería de JavaScript orientada a la programación creativa**. Está basada en el lenguaje Processing y fue diseñada para facilitar el aprendizaje de programación gráfica, interacción y animación.

Desde el punto de vista técnico:

- p5.js trabaja sobre el elemento <canvas> de HTML5.
- Internamente utiliza el **contexto de dibujo 2D del navegador**.
- Puede trabajar también en modo **WEBGL**, lo que permite gráficos en 3D.

Concepto importante: Renderizado

Cuando ejecutamos un programa en p5.js, el navegador:

1. Crea un lienzo (canvas).
2. Ejecuta funciones de dibujo.
3. Convierte instrucciones matemáticas en píxeles visibles.

Es decir:

Programar en p5.js es transformar números (coordenadas, colores, tamaños) en imágenes.

1.1 Estructura de un proyecto

Un proyecto básico contiene:

- index.html
- sketch.js
- Librería p5.js

💡 ¿Por qué esta estructura?

Porque p5.js funciona dentro del navegador, y el navegador entiende:

- HTML → estructura
- JavaScript → lógica y comportamiento

	Ingeniería en Sistemas Computacionales	Guia p5.js Unidad 1
	Dr. Juan Gabriel Loaiza	02 / 2026

Flujo de ejecución

1. El navegador abre index.html
 2. Se carga la librería p5.js
 3. Se carga sketch.js
 4. p5 ejecuta automáticamente setup()
 5. Luego ejecuta draw() si existe
-

1.2 La función setup()

```
function setup() {
  createCanvas(600, 400);
}
```

¿Qué es setup()?

Es una función especial que:

- Se ejecuta **una sola vez**
- Inicializa el entorno gráfico
- Define tamaño del canvas
- Inicializa variables

En términos de programación:

setup() es la fase de inicialización del sistema.

1.3 La función createCanvas(w, h)

```
createCanvas(600, 400);
```

¿Qué hace?

Crea un plano bidimensional donde dibujaremos.

- w = ancho (width)

	Ingeniería en Sistemas Computacionales	Guia p5.js Unidad 1
	Dr. Juan Gabriel Loaiza	02 / 2026

- $h = \text{alto} (\text{height})$

Ambos se miden en **píxeles**.

Explicación matemática sencilla

El canvas puede entenderse como un **plano cartesiano modificado**.

En matemáticas tradicionales:

- El origen $(0,0)$ está en el centro.
- El eje Y crece hacia arriba.

En p5.js:

- El origen $(0,0)$ está en la esquina superior izquierda.
- El eje X crece hacia la derecha.
- El eje Y crece hacia abajo.

Representación:

$(0,0) \xrightarrow{\hspace{1cm}} X$



Si el canvas es 600×400 :

- X puede tomar valores entre 0 y 600
- Y puede tomar valores entre 0 y 400

Esto significa que el canvas es un conjunto discreto de puntos:

$$\text{Canvas} = \{(x, y) \mid 0 \leq x \leq 600, 0 \leq y \leq 400\}$$

Cada punto corresponde a un píxel.

	Ingeniería en Sistemas Computacionales	Guia p5.js Unidad 1
	Dr. Juan Gabriel Loaiza	02 / 2026

1.4 Concepto matemático del píxel

Un píxel es la unidad mínima de representación digital.

Desde el punto de vista matemático:

El canvas es una matriz:

$$\text{Canvas} = \mathbb{R}^{w \times h}$$

Más específicamente:

Es una matriz de valores RGB:

$$\text{Pixel}(x, y) = (R, G, B)$$

Donde cada componente:

$$0 \leq R, G, B \leq 255$$

Por lo tanto:

Programar gráficos es manipular matrices de color.

1.5 ¿Qué pasa si no usamos createCanvas()?

Si no definimos el canvas, p5 crea uno por defecto de:

100 x 100 píxeles.

Esto limita el espacio de trabajo y las coordenadas disponibles.

1.6 Fondo del canvas — background()

Si agregamos:

`background(200);`

Estamos diciendo:

- 200 en escala de grises
- 0 sería negro

	Ingeniería en Sistemas Computacionales	Guia p5.js Unidad 1
	Dr. Juan Gabriel Loaiza	02 / 2026

- 255 sería blanco

Matemáticamente:

$$Color = (200,200,200)$$

1.7 Variables automáticas: width y height

Después de ejecutar:

```
createCanvas(600, 400);
```

p5 crea automáticamente:

- width = 600
- height = 400

Esto permite cálculos dinámicos:

```
line(0, 0, width, height);
```

Matemáticamente:

Estamos dibujando una recta entre:

$$(0,0) \rightarrow (600,400)$$

1.8 Ejemplo completo explicado

```
function setup() {
  createCanvas(800, 600);
  background(150);
}
```

Explicación paso a paso:

1. Se crea una superficie de 800x600.
2. Se asigna a cada píxel el valor RGB (150,150,150).

	Ingeniería en Sistemas Computacionales	Guia p5.js Unidad 1
	Dr. Juan Gabriel Loaiza	02 / 2026

3. El programa termina (porque no hay draw()).
-

1.9 Ejercicios Progresivos

Nivel 1 (Muy básico)

1. Crear un canvas de 800x600.
 2. Cambiar el fondo a gris.
 3. Cambiar el fondo a azul usando RGB.
-

Nivel 2 (Pensamiento matemático)

4. Dibujar un punto exactamente en el centro usando:

`width/2`

`height/2`

Pregunta:

¿Por qué funciona matemáticamente dividir entre 2?

Respuesta esperada:

Porque estamos encontrando el punto medio del intervalo:

$$\frac{0 + width}{2}$$

Nivel 3 (Razonamiento geométrico)

5. Dibujar cuatro puntos en las esquinas del canvas.

Reflexión matemática:

- Esquina superior izquierda → (0,0)
- Esquina superior derecha → (width,0)
- Esquina inferior izquierda → (0,height)
- Esquina inferior derecha → (width,height)

	Ingeniería en Sistemas Computacionales	Guia p5.js Unidad 1
	Dr. Juan Gabriel Loaiza	02 / 2026

Resumen Conceptual del Capítulo

En este capítulo aprendimos que:

- p5.js transforma números en gráficos.
 - El canvas es un plano cartesiano discreto.
 - Cada figura se define mediante coordenadas.
 - Programar gráficos es aplicar matemáticas básicas sobre un plano.
-

CAPÍTULO 2 — La función draw() y el ciclo de animación

2.1 ¿Qué es draw()?

La función:

```
function draw() {  
}
```

es una función especial que:

- Se ejecuta automáticamente después de setup()
- Se repite continuamente
- Se ejecuta aproximadamente **60 veces por segundo** por defecto

Es decir:

draw() → 60 ejecuciones por segundo

2.2 Concepto Matemático: Tiempo Discreto

En matemáticas existen dos formas de modelar el tiempo:

Tiempo continuo

	Ingeniería en Sistemas Computacionales	Guia p5.js Unidad 1
	Dr. Juan Gabriel Loaiza	02 / 2026

$$t \in \mathbb{R}$$

Tiempo discreto

$$t = 0, 1, 2, 3, 4, 5, \dots$$

p5.js trabaja con **tiempo discreto**.

Cada vez que draw() se ejecuta es como si el tiempo avanzara una unidad:

$$t_{n+1} = t_n + 1$$

Cada ejecución es un "frame".

2.3 Concepto de Frame

Un frame es una imagen individual.

Si tenemos:

60 frames por segundo

Entonces:

$$1 \text{ segundo} = 60 \text{ frames}$$

Si una animación dura 5 segundos:

$$5 \times 60 = 300 \text{ frames}$$

2.4 La animación como función matemática

Observemos el ejemplo:

```
let x = 0;  
  
function setup() {  
  createCanvas(400, 200);  
}
```

	Ingeniería en Sistemas Computacionales	Guia p5.js Unidad 1
	Dr. Juan Gabriel Loaiza	02 / 2026

```

    }

function draw() {
  background(220);
  circle(x, 100, 40);
  x += 3;
}

```

¿Qué está pasando matemáticamente?

Tenemos una variable:

$$x_0 = 0$$

Y en cada frame:

$$x_{n+1} = x_n + 3$$

Esto es una **sucesión aritmética**.

Fórmula explícita:

$$x_n = 3n$$

Por lo tanto:

La pelota se mueve con **velocidad constante**.

2.5 Concepto físico: Movimiento Rectilíneo Uniforme (MRU)

En física:

$$x(t) = x_0 + vt$$

Aquí:

- $x_0 = 0$

	Ingeniería en Sistemas Computacionales	Guia p5.js Unidad 1
	Dr. Juan Gabriel Loaiza	02 / 2026

- $v = 3\text{píxeles por frame}$
- $t = \text{número de frame}$

Entonces:

$$x = 3t$$

p5 está simulando un modelo físico básico.

2.6 ¿Por qué usamos `background()` dentro de `draw()`?

Si quitamos:

`background(220);`

La pelota deja un rastro.

¿Por qué?

Porque el canvas no se limpia automáticamente.

Matemáticamente:

Estamos sumando dibujos acumulativamente:

$$\text{Imagen}_{final} = \sum_{n=0}^N \text{Frame}_n$$

Cuando usamos `background()`, reiniciamos la matriz de píxeles.

2.7 `frameRate(n)`

`frameRate(30);`

Modifica la cantidad de frames por segundo.

Si:

`frameRate(30)`

	Ingeniería en Sistemas Computacionales	Guia p5.js Unidad 1
	Dr. Juan Gabriel Loaiza	02 / 2026

Entonces:

$$1\text{segundo} = 30\text{frames}$$

Si la velocidad es:

$$x += 3$$

Con 60 fps \rightarrow 180 píxeles por segundo

Con 30 fps \rightarrow 90 píxeles por segundo

La velocidad depende de:

$$\text{Velocidad} = \text{incremento} \times \text{frameRate}$$

2.8 noLoop(), loop(), redraw()

noLoop()

Detiene la ejecución continua de draw().

Matemáticamente:

Congela el tiempo discreto.

$$t = \text{constante}$$

loop()

Reanuda la animación.

redraw()

Ejecuta draw() solo una vez.

Útil cuando trabajamos con eventos.

	Ingeniería en Sistemas Computacionales	Guia p5.js Unidad 1
	Dr. Juan Gabriel Loaiza	02 / 2026

2.9 Ejemplo Mejorado: Rebote en los Bordes

Ahora vamos a hacer que la pelota rebote.

Código:

```
let x = 0;
let velocidad = 3;
function setup() {
  createCanvas(400, 200);
}
function draw() {
  background(220);
  circle(x, 100, 40);
  x += velocidad;
  if (x > width || x < 0) {
    velocidad = -velocidad;
  }
}
```

2.10 Explicación Matemática del Rebote

Tenemos ahora:

$$x_{n+1} = x_n + v$$

Pero si:

$$x > width \text{ or } x < 0$$

Entonces:

$$v = -v$$

	Ingeniería en Sistemas Computacionales	Guia p5.js Unidad 1
	Dr. Juan Gabriel Loaiza	02 / 2026

Esto genera una función por partes:

$$v = \begin{cases} +3 & \text{si se mueve derecha} \\ -3 & \text{si se mueve izquierda} \end{cases}$$

Es un modelo discreto de colisión elástica simple.

2.11 Modelo Matemático del Rebote

Es una función periódica triangular.

El movimiento total es:

$$x(t) = |(vt \bmod 2L) - L|$$

Donde:

- $L = width$
- $v = velocidad$

Pero p5 lo simula de forma iterativa.

2.12 Problema Común: Centro del círculo

Observa que:

`circle(x, 100, 40);`

El valor x representa el centro.

El radio es:

$$r = \frac{40}{2} = 20$$

Para que el rebote sea correcto:

`if (x > width - 20 || x < 20)`

	Ingeniería en Sistemas Computacionales	Guia p5.js Unidad 1
	Dr. Juan Gabriel Loaiza	02 / 2026

Eso es más preciso matemáticamente.

2.13 Ejercicios Progresivos

Nivel 1 — Comprensión básica

1. Cambiar la velocidad a 5.
 2. Cambiar la pelota para que se mueva verticalmente.
 3. Cambiar frameRate a 30.
-

Nivel 2 — Razonamiento matemático

4. Calcular cuántos frames tarda en cruzar toda la pantalla si:
 - o width = 400
 - o velocidad = 4

Respuesta esperada:

$$400/4 = 100 \text{ frames}$$

Nivel 3 — Aplicación física

5. Hacer que la pelota rebote en ambas direcciones (x e y).
6. Simular gravedad:

$$v = v + g$$

2.14 Conclusión Conceptual

En este capítulo aprendimos que:

- draw() representa tiempo discreto.

	Ingeniería en Sistemas Computacionales	Guia p5.js Unidad 1
	Dr. Juan Gabriel Loaiza	02 / 2026

- Las animaciones son sucesiones matemáticas.
- El movimiento uniforme es una progresión aritmética.
- El rebote es un cambio de signo en la velocidad.
- frameRate controla la relación tiempo–velocidad.

CAPÍTULO 3 — Sistema de Coordenadas en p5.js

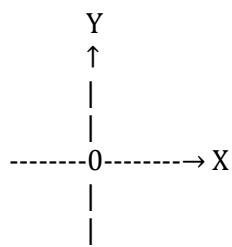
3.1 ¿Qué es un sistema de coordenadas?

Un sistema de coordenadas es un mecanismo matemático que permite ubicar puntos en el espacio mediante números.

En el plano cartesiano tradicional (matemáticas):

- El origen está en el centro.
- El eje X crece hacia la derecha.
- El eje Y crece hacia arriba.

Representación clásica:



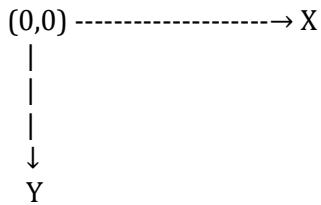
3.2 Sistema de coordenadas en p5.js

En p5.js el sistema cambia ligeramente:

- El origen (0,0) está en la esquina superior izquierda.
- El eje X crece hacia la derecha.
- El eje Y crece hacia abajo.

Representación en pantalla:

	Ingeniería en Sistemas Computacionales	Guia p5.js Unidad 1
	Dr. Juan Gabriel Loaiza	02 / 2026



Esto significa que el eje Y está invertido respecto al plano matemático tradicional.

3.3 Explicación Matemática

Si el canvas es:

```
createCanvas(600, 400);
```

Entonces el dominio de puntos posibles es:

$$\begin{aligned} 0 \leq x \leq 600 \\ 0 \leq y \leq 400 \end{aligned}$$

El canvas puede interpretarse como el conjunto:

$$C = \{(x, y) \in \mathbb{R}^2 \mid 0 \leq x \leq width, 0 \leq y \leq height\}$$

Pero en realidad es discreto (trabaja en píxeles):

$$x, y \in \mathbb{Z}$$

Es decir, coordenadas enteras.

3.4 Concepto de píxel como punto discreto

Cada punto del canvas es un píxel:

$$Pixel(x, y)$$

Y cada píxel almacena un color:

$$(R, G, B)$$

	Ingeniería en Sistemas Computacionales	Guia p5.js Unidad 1
	Dr. Juan Gabriel Loaiza	02 / 2026

Por lo tanto:

El sistema de coordenadas es una matriz de tamaño:

$$width \times height$$

Si:

```
createCanvas(800, 600);
```

Tenemos:

$$800 \times 600 = 480,000 \text{ píxeles}$$

3.5 Variables automáticas: width y height

Después de ejecutar:

```
createCanvas(600, 400);
```

p5 crea automáticamente:

- width = 600
- height = 400

Estas variables permiten hacer cálculos dinámicos.

Por ejemplo:

```
circle(width/2, height/2, 50);
```

3.6 Explicación matemática del centro

Si el intervalo en X es:

$$[0, width]$$

El punto medio es:

$$\frac{0 + width}{2} = \frac{width}{2}$$

	Ingeniería en Sistemas Computacionales Dr. Juan Gabriel Loaiza	Guia p5.js Unidad 1 02 / 2026
---	---	----------------------------------

Lo mismo para Y:

$$\frac{height}{2}$$

Por eso el centro del canvas es:

$$\left(\frac{width}{2}, \frac{height}{2}\right)$$

3.7 Ejemplo completo: Ejes centrales

```
function setup() {
  createCanvas(600, 400);
}

function draw() {
  background(220);
  // Línea vertical central
  line(width/2, 0, width/2, height);
  // Línea horizontal central
  line(0, height/2, width, height/2);
}
```

Explicación geométrica

La línea vertical está definida por:

$$x = \frac{width}{2}$$

La línea horizontal está definida por:

	Ingeniería en Sistemas Computacionales	Guia p5.js Unidad 1
	Dr. Juan Gabriel Loaiza	02 / 2026

$$y = \frac{height}{2}$$

Estamos dibujando dos rectas:

$$x = \text{constante}$$

$$y = \text{constante}$$

3.8 Coordenadas de las esquinas

Las cuatro esquinas del canvas son:

- Superior izquierda → (0, 0)
- Superior derecha → (width, 0)
- Inferior izquierda → (0, height)
- Inferior derecha → (width, height)

Ejemplo:

```
point(0, 0);
point(width, 0);
point(0, height);
point(width, height);
```

3.9 Distancia entre puntos (base geométrica)

La distancia entre dos puntos se calcula con:

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

Esto es importante porque:

Muchas animaciones usan esta fórmula para:

- Detectar colisiones

	Ingeniería en Sistemas Computacionales	Guia p5.js Unidad 1
	Dr. Juan Gabriel Loaiza	02 / 2026

- Detectar proximidad al mouse
- Crear interacciones

Ejemplo sencillo:

Si queremos saber la distancia del mouse al centro:

$$d = \sqrt{(mouseX - width/2)^2 + (mouseY - height/2)^2}$$

3.10 Ejercicio principal del capítulo

Dibujar líneas que crucen el centro del canvas

Código esperado:

```
function setup() {
  createCanvas(600, 400);
}

function draw() {
  background(240);
  line(width/2, 0, width/2, height);
  line(0, height/2, width, height/2);
}
```

3.11 Ejercicios Progresivos

Nivel 1 — Ubicación básica

1. Dibujar un punto en el centro.
 2. Dibujar un círculo en cada esquina.
 3. Dibujar un rectángulo centrado.
-

	Ingeniería en Sistemas Computacionales	Guia p5.js Unidad 1
	Dr. Juan Gabriel Loaiza	02 / 2026

Nivel 2 — Pensamiento geométrico

4. Dividir la pantalla en 4 cuadrantes usando líneas.
5. Colocar un círculo en el centro de cada cuadrante.

Centro de cada cuadrante:

$$\begin{aligned} & \left(\frac{\text{width}}{4}, \frac{\text{height}}{4} \right) \\ & \left(\frac{3\text{width}}{4}, \frac{\text{height}}{4} \right) \\ & \left(\frac{\text{width}}{4}, \frac{3\text{height}}{4} \right) \\ & \left(\frac{3\text{width}}{4}, \frac{3\text{height}}{4} \right) \end{aligned}$$

Nivel 3 — Aplicación matemática

6. Dibujar una diagonal:

`line(0, 0, width, height);`

Esta línea representa la recta:

$$y = \frac{\text{height}}{\text{width}} x$$

3.12 Reflexión Matemática Final

El sistema de coordenadas en p5.js es:

- Un plano cartesiano invertido en el eje Y.
- Una matriz discreta de píxeles.
- Un espacio bidimensional donde toda figura es una función matemática.

Toda forma gráfica puede expresarse como:

$$F(x, y)$$

	Ingeniería en Sistemas Computacionales	Guia p5.js Unidad 1
	Dr. Juan Gabriel Loaiza	02 / 2026

Programar gráficos es aplicar geometría analítica en tiempo real.

CAPÍTULO 4 — Colores y Estilos en p5.js

4.1 ¿Qué es el color en computación?

En programación gráfica, el color no es una propiedad abstracta: es un **vector numérico**.

Cada píxel del canvas almacena información de color en forma de números.

Matemáticamente:

$$Color = (R, G, B)$$

Donde:

$$0 \leq R, G, B \leq 255$$

Esto significa que cada componente puede tomar 256 valores posibles.

Por lo tanto:

$$256^3 = 16,777,216 \text{ colores posibles}$$

4.2 Funciones principales de color y estilo

1. background()

```
background(200);
```

```
background(0, 100, 200);
```

Define el color del fondo del canvas.

Matemáticamente:

Asigna el mismo valor RGB a toda la matriz de píxeles.

	Ingeniería en Sistemas Computacionales	Guia p5.js Unidad 1
	Dr. Juan Gabriel Loaiza	02 / 2026

$$Canvas(x, y) = Color$$

Si no se usa dentro de draw(), el fondo no se reinicia.

2. stroke()

stroke(255);

stroke(255, 0, 0);

Define el color del contorno de las figuras.

Es el color del borde.

3. fill()

fill(255, 0, 0);

Define el color interno de una figura.

Si usamos:

noFill();

La figura solo tendrá contorno.

4. strokeWeight()

strokeWeight(5);

Define el grosor del contorno.

Desde el punto de vista geométrico:

Un borde no es una línea ideal matemática (sin grosor),
sino una franja con espesor.

Si:

$$strokeWeight = 5$$

	Ingeniería en Sistemas Computacionales	Guia p5.js Unidad 1
	Dr. Juan Gabriel Loaiza	02 / 2026

Entonces el contorno tiene 5 píxeles de grosor.

5. strokeCap()

Define la forma del extremo de una línea.

Opciones:

- ROUND (redondeado)
- SQUARE (cuadrado)
- PROJECT (extendido)

Desde el punto de vista geométrico:

Modifica la forma del segmento en sus extremos.

6. strokeJoin()

Define cómo se unen dos segmentos de línea.

Opciones:

- MITER
- BEVEL
- ROUND

Esto modifica la geometría del vértice.

4.3 Modelos de color

4.3.1 Escala de grises

`background(150);`

Un solo número:

$$Color = (150,150,150)$$

	Ingeniería en Sistemas Computacionales	Guia p5.js Unidad 1
	Dr. Juan Gabriel Loaiza	02 / 2026

Representa intensidad luminosa.

0 → negro
255 → blanco

4.3.2 Modelo RGB

`fill(255, 0, 0);`

RGB significa:

- Red
- Green
- Blue

Matemáticamente es un vector tridimensional:

$$Color \in \mathbb{R}^3$$

El espacio RGB puede representarse como un cubo tridimensional.

4.3.3 Modelo Hexadecimal

`fill("#FF0000");`

Es otra representación del mismo modelo RGB.

#FF0000 significa:

FF → 255
00 → 0
00 → 0

4.3.4 Modelo HSB (Hue, Saturation, Brightness)

HSB trabaja diferente.

- Hue → tono (0–360 grados)
- Saturation → saturación

	Ingeniería en Sistemas Computacionales	Guia p5.js Unidad 1
	Dr. Juan Gabriel Loaiza	02 / 2026

- Brightness → brillo

Para usarlo:

```
colorMode(HSB);
```

Matemáticamente:

Hue representa un ángulo en una circunferencia:

$$0^\circ - 360^\circ$$

Es un modelo polar del color.

4.4 Ejemplo explicado paso a paso

```
function setup() {
  createCanvas(400, 400);
}

function draw() {
  background(0, 100, 200);
  stroke(255);
  strokeWeight(5);
  fill(255, 0, 0);
  rect(100, 100, 200, 200);
}
```

Explicación matemática:

1. Fondo = (0,100,200)
2. Contorno = blanco (255,255,255)
3. Grosor = 5 píxeles
4. Relleno = rojo (255,0,0)
5. Rectángulo definido por:

	Ingeniería en Sistemas Computacionales Dr. Juan Gabriel Loaiza	Guia p5.js Unidad 1 02 / 2026
--	---	----------------------------------

$$(x, y) = (100, 100)$$

ancho = 200

alto = 200

4.5 Concepto Geométrico del Rectángulo

Un rectángulo puede definirse como:

$$R = \{(x, y) \mid a \leq x \leq a + w, b \leq y \leq b + h\}$$

p5 dibuja todos los píxeles dentro de ese conjunto.

4.6 Interacción entre stroke y fill

Si hacemos:

`noStroke();`

El borde desaparece.

Si hacemos:

`noFill();`

Solo queda el contorno.

Esto permite construir figuras técnicas.

4.7 Ejercicio Principal

Crear una bandera usando rectángulos.

Ejemplo: bandera simple de tres franjas horizontales.

```
function setup() {
  createCanvas(600, 300);
}
```

	Ingeniería en Sistemas Computacionales	Guia p5.js Unidad 1
	Dr. Juan Gabriel Loaiza	02 / 2026

```
function draw() {
    background(255);

    fill(0, 128, 0);
    rect(0, 0, width, height/3);

    fill(255);
    rect(0, height/3, width, height/3);

    fill(255, 0, 0);
    rect(0, 2*height/3, width, height/3);
}
```

4.8 Explicación Matemática del Ejercicio

Estamos dividiendo el intervalo vertical:

$$[0, height]$$

En tres partes iguales:

$$\frac{height}{3}$$

Esto es partición de un intervalo.

4.9 Ejercicios Progresivos

Nivel 1 — Comprensión básica

1. Dibujar tres círculos de diferentes colores.

	Ingeniería en Sistemas Computacionales	Guia p5.js Unidad 1
	Dr. Juan Gabriel Loaiza	02 / 2026

2. Cambiar el grosor del borde.
 3. Dibujar una figura sin relleno.
-

Nivel 2 — Razonamiento matemático

4. Crear un degradado usando un ciclo:

$$Color(x) = \frac{x}{width} \times 255$$

Nivel 3 — Espacio de color

5. Cambiar a HSB y crear un arco iris.

```
colorMode(HSB);
for (let i = 0; i < width; i++) {
  stroke(i % 360, 100, 100);
  line(i, 0, i, height);
}
```

4.10 Reflexión Conceptual

En este capítulo aprendimos que:

- El color es información numérica.
- RGB es un espacio tridimensional.
- HSB es un modelo polar.
- strokeWeight tiene interpretación geométrica.
- fill y stroke modifican subconjuntos del plano.

Programar gráficos es manipular:

$$Color(x, y)$$

	Ingeniería en Sistemas Computacionales	Guia p5.js Unidad 1
	Dr. Juan Gabriel Loaiza	02 / 2026

Es decir, aplicar funciones matemáticas sobre una matriz.

CAPÍTULO 5 — Figuras Geométricas en p5.js

5.1 Introducción: Geometría en programación

En p5.js, cada figura geométrica es una representación digital de un conjunto de puntos en el plano.

Desde el punto de vista matemático:

Toda figura es un subconjunto del plano cartesiano:

$$F \subset \mathbb{R}^2$$

Pero en la computadora trabajamos con puntos discretos:

$$F \subset \mathbb{Z}^2$$

Cada función de dibujo define qué píxeles deben activarse.

5.2 line(x1, y1, x2, y2)

line(50, 100, 350, 100);

Definición geométrica

Una línea es el conjunto de puntos entre dos puntos dados.

Matemáticamente:

Dados:

$$\begin{aligned} P_1 &= (x_1, y_1) \\ P_2 &= (x_2, y_2) \end{aligned}$$

La recta que los une puede describirse como:

$$y = mx + b$$

	Ingeniería en Sistemas Computacionales	Guia p5.js Unidad 1
	Dr. Juan Gabriel Loaiza	02 / 2026

donde:

$$m = \frac{y_2 - y_1}{x_2 - x_1}$$

p5 utiliza un algoritmo interno (similar a Bresenham) para aproximar esa recta en píxeles.

5.3 rect(x, y, w, h)

rect(100, 100, 200, 150);

Interpretación matemática

Define el conjunto:

$$R = \{(x, y) \mid a \leq x \leq a + w, b \leq y \leq b + h\}$$

Donde:

- (a,b) es la esquina
 - w = ancho
 - h = alto
-

5.4 square(x, y, s)

Es un caso particular de rectángulo donde:

$$w = h = s$$

Es decir, todos los lados son iguales.

5.5 circle(x, y, d)

circle(200, 200, 100);

Interpretación matemática

	Ingeniería en Sistemas Computacionales	Guia p5.js Unidad 1
	Dr. Juan Gabriel Loaiza	02 / 2026

Un círculo es el conjunto de puntos que cumplen:

$$(x - a)^2 + (y - b)^2 = r^2$$

Donde:

- (a,b) es el centro
- r = radio
- d = diámetro = 2r

En p5, se da el diámetro, no el radio.

5.6 ellipse(x, y, w, h)

ellipse(200, 200, 150, 100);

Definición matemática

Una elipse cumple:

$$\frac{(x-a)^2}{r_x^2} + \frac{(y-b)^2}{r_y^2} = 1$$

Donde:

- $r_x = \frac{w}{2}$
- $r_y = \frac{h}{2}$

Es una deformación del círculo.

5.7 triangle(x1, y1, x2, y2, x3, y3)

Define un polígono de tres lados.

Matemáticamente:

Es el conjunto de puntos dentro del triángulo definido por los vértices.

p5 rellena el área interna usando interpolación.

	Ingeniería en Sistemas Computacionales	Guia p5.js Unidad 1
	Dr. Juan Gabriel Loaiza	02 / 2026

5.8 quad(x1, y1, x2, y2, x3, y3, x4, y4)

Define un polígono de cuatro lados.

Importante:

Los puntos deben darse en orden (horario o antihorario).

Desde el punto de vista geométrico:

Es la unión de dos triángulos.

5.9 arc(x, y, w, h, start, stop)

arc(200, 200, 200, 200, 0, PI);

Interpretación matemática

Un arco es una porción de una elipse.

Los ángulos se expresan en radianes:

$$\pi \text{ rad} = 180^\circ$$

Constantes útiles:

- PI
- TWO_PI
- HALF_PI

Un círculo completo en radianes:

$$2\pi$$

5.10 Modos de interpretación

rectMode()

	Ingeniería en Sistemas Computacionales	Guia p5.js Unidad 1
	Dr. Juan Gabriel Loaiza	02 / 2026

Controla cómo se interpretan los parámetros.

rectMode(CORNER) (por defecto)

(x,y) es la esquina superior izquierda.

rectMode(CENTER)

(x,y) es el centro.

Matemáticamente cambia el punto de referencia.

ellipseMode()

ellipseMode(CENTER)

Centro como referencia.

ellipseMode(CORNER)

(x,y) es la esquina del rectángulo contenedor.

5.11 Ejemplo Integrador: Dibujar una Casa

```
function setup() {
  createCanvas(600, 400);
}
```

```
function draw() {
  background(220);

  // Base
  fill(200, 150, 100);
  rect(200, 200, 200, 150);
```

	Ingeniería en Sistemas Computacionales Dr. Juan Gabriel Loaiza	Guia p5.js Unidad 1 02 / 2026
--	---	----------------------------------

// Techo

```
fill(150, 50, 50);
triangle(200, 200, 300, 120, 400, 200);
```

// Puerta

```
fill(100, 50, 20);
rect(280, 270, 40, 80);
```

// Sol

```
fill(255, 200, 0);
circle(500, 80, 80);
}
```

5.12 Análisis Geométrico del Ejercicio

Base de la casa:

$$(200,200) \rightarrow (400,350)$$

Techo:

Triángulo isósceles con vértice superior en:

$$(300, 120)$$

El sol:

Centro en (500,80), radio 40.

Estamos combinando:

- Rectángulo
- Triángulo

	Ingeniería en Sistemas Computacionales	Guia p5.js Unidad 1
	Dr. Juan Gabriel Loaiza	02 / 2026

- Círculo

Es composición geométrica.

5.13 Ejercicios Progresivos

Nivel 1 — Práctica básica

1. Dibujar 3 círculos alineados.
 2. Dibujar un triángulo equilátero aproximado.
 3. Dibujar un arco semicircular.
-

Nivel 2 — Geometría aplicada

4. Dibujar un objetivo (círculos concéntricos).
 5. Dibujar una ventana usando rectángulos y líneas.
 6. Dibujar una flor usando círculos distribuidos en ángulo.
-

Nivel 3 — Pensamiento matemático

7. Dibujar un polígono regular usando trigonometría:

$$x = a + r\cos(\theta)$$

$$y = b + r\sin(\theta)$$

5.14 Reflexión Final

En este capítulo aprendimos que:

- Cada figura es una función matemática.
- El plano digital es discreto.
- Círculo y elipse provienen de ecuaciones cuadráticas.

	Ingeniería en Sistemas Computacionales	Guia p5.js Unidad 1
	Dr. Juan Gabriel Loaiza	02 / 2026

- Triángulos y cuadriláteros son polígonos definidos por vértices.
- rectMode y ellipseMode cambian el sistema de referencia.

Programar figuras es aplicar:

- Geometría analítica
- Álgebra básica
- Trigonometría elemental

CAPÍTULO 7 — Arcos y Ángulos

7.1 ¿Qué es un arco?

Un **arco** es una parte de un círculo (o elipse).

En p5.js se dibuja con:

`arc(x, y, w, h, start, stop);`

- (x, y) → posición de referencia (por defecto: el **centro** del arco)
- w, h → ancho y alto del óvalo contenedor
- $start, stop$ → ángulo inicial y final (**en radianes**)

Si $w = h$, entonces estás trabajando con un **círculo**.

Si $w \neq h$, el arco será parte de una **elipse**.

7.2 ¿Por qué p5 usa radianes?

Porque en matemáticas (cálculo, física, trigonometría) es la unidad natural para trabajar con ángulos.

Definición matemática de radián

Un ángulo en radianes se define como:

$$\theta = \frac{s}{r}$$

	Ingeniería en Sistemas Computacionales Dr. Juan Gabriel Loaiza	Guia p5.js Unidad 1 02 / 2026
--	---	----------------------------------

donde:

- s = longitud del arco
- r = radio

En un círculo completo, la longitud de la circunferencia es:

$$s = 2\pi r$$

Por lo tanto el ángulo total es:

$$\theta = \frac{2\pi r}{r} = 2\pi$$

Un círculo completo equivale a:

$$2\pi \text{ rad}$$

7.3 Constantes útiles en p5.js

p5.js ya trae constantes listas:

- PI = $\pi \approx 3.14159 \rightarrow 180^\circ$
- TWO_PI = $2\pi \rightarrow 360^\circ$
- HALF_PI = $\pi/2 \rightarrow 90^\circ$

Tabla rápida (grado ↔ radián)

Grados Radianes

$$0^\circ \quad 0$$

$$90^\circ \quad \pi/2$$

$$180^\circ \quad \pi$$

$$270^\circ \quad 3\pi/2$$

	Ingeniería en Sistemas Computacionales	Guia p5.js Unidad 1
	Dr. Juan Gabriel Loaiza	02 / 2026

Grados Radianes

$$360^\circ \quad 2\pi$$

7.4 Matemática sencilla detrás de un arco

Un círculo con centro (a, b) y radio r se describe con:

$$\begin{aligned}x &= a + r\cos(\theta) \\y &= b + r\sin(\theta)\end{aligned}$$

Donde:

- θ es el ángulo
- cos y sin devuelven la posición en X y Y

Un **arco** simplemente dibuja esa curva, pero solo para:

$$\theta \in [start, stop]$$

7.5 Ejemplo básico de arc()

```
function setup() {
  createCanvas(400, 400);
}

function draw() {
  background(220);

  // arco de 0 a PI (media circunferencia)
  arc(200, 200, 200, 200, 0, PI);
}
```

	Ingeniería en Sistemas Computacionales	Guia p5.js Unidad 1
	Dr. Juan Gabriel Loaiza	02 / 2026

☞ Nota: 0 y PI aquí generan medio círculo.

7.6 Detalle importante: orientación del ángulo en p5.js

En matemática tradicional:

- el ángulo 0° apunta hacia la derecha
- los ángulos crecen en sentido antihorario

En p5.js pasa algo similar, pero como el eje Y crece hacia abajo, visualmente se siente "invertido".

Regla práctica para enseñar:

- 0 rad apunta a la derecha (3 en un reloj)
 - HALF_PI apunta hacia abajo (6 en un reloj)
 - PI apunta a la izquierda (9 en un reloj)
 - -HALF_PI apunta hacia arriba (12 en un reloj)
-

7.7 Ejercicio del capítulo: Reloj analógico

Para un reloj necesitamos:

- Un círculo (carátula)
 - Marcas o números (opcional)
 - Manecillas (líneas con ángulos)
 - Un arco opcional para "segundero" o progreso
-

7.8 Matemática de las manecillas (lo importante)

Un punto en un círculo se calcula con trigonometría:

$$x = cx + r\cos(\theta)$$

$$y = cy + r\sin(\theta)$$

Esto se usa para dibujar una manecilla desde el centro:

	Ingeniería en Sistemas Computacionales	Guia p5.js Unidad 1
	Dr. Juan Gabriel Loaiza	02 / 2026

Línea: $(cx, cy) \rightarrow (x, y)$

7.9 Código sencillo: reloj analógico (básico)

Este reloj dibuja carátula y una manecilla que gira con el tiempo.

```
function setup() {
    createCanvas(400, 400);
}
```

```
function draw() {
    background(240);
```

```
    let cx = width / 2;
    let cy = height / 2;
    let r = 150;
```

// 1) Carátula

```
    stroke(0);
    strokeWeight(3);
    noFill();
    circle(cx, cy, r * 2);
```

// 2) Ángulo que avanza (simulación simple)

```
// frameCount va aumentando 1 por frame
let ang = frameCount * 0.02;
```

	Ingeniería en Sistemas Computacionales	Guia p5.js Unidad 1
	Dr. Juan Gabriel Loaiza	02 / 2026

// 3) Punto final de la manecilla usando cos/sin

```
let x2 = cx + (r * 0.8) * cos(ang);
let y2 = cy + (r * 0.8) * sin(ang);
```

// 4) Manecilla

```
strokeWeight(5);
line(cx, cy, x2, y2);
```

// 5) Arco de progreso (opcional)

```
strokeWeight(8);
arc(cx, cy, r * 2.1, r * 2.1, 0, ang);
}
```

7.10 Explicación matemática del ejemplo

- frameCount es el “tiempo discreto” (Cap. 2)
- Multiplicar por 0.02 hace que el ángulo aumente poco a poco:

$$\theta = 0.02 \cdot t$$

- $\cos(\theta)$ y $\sin(\theta)$ convierten el ángulo en coordenadas sobre el círculo.
 - El radio controla la longitud de la manecilla.
-

7.11 Ejercicios progresivos

Nivel 1 (básico)

1. Dibujar un arco de 0 a HALF_PI
2. Dibujar un arco de PI a TWO_PI

	Ingeniería en Sistemas Computacionales	Guia p5.js Unidad 1
	Dr. Juan Gabriel Loaiza	02 / 2026

Nivel 2 (intermedio)

3. Hacer 12 marcas alrededor del reloj usando un ciclo y trigonometría.
4. Agregar una segunda manecilla más corta que gire más lento.

Nivel 3 (reto)

5. Usar second(), minute(), hour() para que sea un reloj real:
 - segundos → 0 a TWO_PI
 - minutos → 0 a TWO_PI
 - horas (12h) → 0 a TWO_PI

(Puedo darte esa versión realista si la quieres para el libro).

7.12 Cierre conceptual

En este capítulo aprendimos que:

- p5 usa **radianes** para ángulos.
- Un círculo completo es TWO_PI.
- Los arcos son porciones de la circunferencia.
- Con trigonometría se convierten ángulos en puntos:

$$(x, y) = (cx + r\cos \theta, cy + r\sin \theta)$$

- Esto es la base de animaciones circulares, relojes, órbitas, etc.

CAPÍTULO 8 — Suavizado y Calidad Visual

8.1 ¿Qué es el suavizado?

Cuando dibujamos líneas o figuras curvas en una pantalla, estamos representando formas continuas (matemáticas) usando una cuadrícula discreta de píxeles.

	Ingeniería en Sistemas Computacionales	Guia p5.js Unidad 1
	Dr. Juan Gabriel Loaiza	02 / 2026

Esto genera un problema visual llamado:

Aliasing

El aliasing produce bordes “dentados” o escalonados.

Ejemplo típico:
una línea diagonal se ve como una escalera.

8.2 Concepto Matemático del Problema

En matemáticas, una recta es continua:

$$y = mx + b$$

Pero en pantalla solo podemos dibujar puntos enteros:

$$x, y \in \mathbb{Z}$$

Esto significa que estamos aproximando una función continua con valores discretos.

Es un problema de:

Muestreo d

iscreto d

8.3 ¿Qué hace smooth()?

`smooth();`

Activa el suavizado (anti-aliasing).

En términos técnicos:

El navegador aplica interpolación de color en los bordes.

En lugar de tener:



Se generan píxeles intermedios con colores mezclados.

	Ingeniería en Sistemas Computacionales	Guia p5.js Unidad 1
	Dr. Juan Gabriel Loaiza	02 / 2026

Matemáticamente, es una interpolación:

$$Color_{nuevo} = \alpha C_1 + (1 - \alpha)C_2$$

Donde:

- C_1 es el color del objeto
- C_2 es el color del fondo
- $\alpha \in [0,1]$

Esto genera transición suave.

8.4 ¿Qué hace noSmooth()?

`noSmooth();`

Desactiva el suavizado.

Cada píxel se dibuja sin interpolación.

Esto es útil cuando:

- Se quiere estilo pixel-art
 - Se trabaja con gráficos retro
 - Se necesita precisión de píxel
-

8.5 Ejemplo Comparativo

```
function setup() {
  createCanvas(400, 200);
}
```

```
function draw() {
  background(240);
```

	Ingeniería en Sistemas Computacionales	Guia p5.js Unidad 1
	Dr. Juan Gabriel Loaiza	02 / 2026

```
// Con suavizado
smooth();
strokeWeight(8);
line(50, 150, 150, 50);
```

```
// Sin suavizado
noSmooth();
line(250, 150, 350, 50);
}
```

Visualmente:

- La primera línea se ve suave.
 - La segunda se ve escalonada.
-

8.6 Explicación Geométrica

Una línea diagonal ideal cumple:

$$y = -x + c$$

Pero en píxeles solo podemos activar puntos como:

$$(10,15), (11,14), (12,13) \dots$$

Esto crea escalones.

El suavizado reduce la diferencia de intensidad entre píxeles vecinos.

8.7 Relación con Resolución

Si aumentamos la resolución del canvas:

```
createCanvas(1200, 800);
```

	Ingeniería en Sistemas Computacionales	Guia p5.js Unidad 1
	Dr. Juan Gabriel Loaiza	02 / 2026

Los escalones se notan menos.

Porque el muestreo es más fino.

Esto se relaciona con el concepto de:

Frecuencia de muestreo

En procesamiento digital:

Mayor resolución → menor error de aproximación.

8.8 Cuándo usar cada uno

Usar smooth() cuando:

- Se trabaja con ilustraciones suaves
 - Se quiere apariencia moderna
 - Se usan curvas y círculos grandes

Usar noSmooth() cuando:

- Se busca estilo retro
 - Se hacen gráficos pixelados
 - Se requiere precisión geométrica

8.9 Ejercicio práctico

1. Dibujar un conjunto de líneas diagonales con y sin suavizado.
 2. Dibujar un círculo pequeño con y sin suavizado.
 3. Crear un patrón tipo “pixel art” usando noSmooth().

8.10 Reflexión

Con este capítulo cerramos los fundamentos visuales de p5.js.

Hemos aprendido que:

	Ingeniería en Sistemas Computacionales	Guia p5.js Unidad 1
	Dr. Juan Gabriel Loaiza	02 / 2026

- Las figuras matemáticas son continuas.
- La pantalla es discreta.
- El suavizado reduce errores de discretización.
- Los gráficos digitales son aproximaciones numéricas.

En términos matemáticos:

La graficación por computadora es:

Geometría c ontinua → a

Conclusión General

A lo largo de los capítulos hemos visto que:

- El canvas es un plano cartesiano discreto.
- Las figuras son ecuaciones matemáticas.
- La animación es una sucesión en tiempo discreto.
- La interacción introduce sistemas dinámicos.
- El color es un vector tridimensional.
- La trigonometría controla el movimiento circular.
- El suavizado mejora la aproximación visual.

La programación gráfica es, en esencia:

Matemáticas aplicadas en la programación

PROYECTO FINAL — Mini Proyecto Integrador

Diseñar una escena animada interactiva que incluya:

- Múltiples figuras
- Animación
- Interacción con el mouse
- Uso correcto de color y estilos
- Control del ciclo (loop / noLoop)