INSTITUTO TECNOLÓGICO SUPERIOR PROGRESO

Carrera:

Ingeniería en sistemas computacionales

Materia:

Graficación

Actividad:

Reporte de practica 2-3

Docente:

Dr. Holzen Atocha Martínez García

Alumno:

David Ezequiel Caballero González

Fecha de entrega:

Lunes 8 de septiembre de 2025

Práctica 2-3

Introducción

La representación gráfica de datos es una de las aplicaciones más comunes de la programación visual, y en esta práctica se desarrolló un programa en Processing cuyo objetivo es generar un gráfico circular (pie chart), representando porcentajes mediante sectores de un círculo. Este tipo de representación permite visualizar de forma clara la proporción de distintos valores dentro de un conjunto total.

El código utiliza funciones propias de Processing como ***arc()*** para trazar los sectores circulares, y emplea cálculos con ángulos en radianes para determinar el tamaño de cada segmento. Además, cada sección se dibuja con un color aleatorio, facilitando la diferenciación visual de los datos.

- El código puede verse en el repositorio en GitHub: Graficación_Práctica 2_Práctica 2-3

Objetivos

Desarrollar un programa en Processing que represente datos en forma de gráfico circular (pie chart), asignando un sector a cada valor de un conjunto porcentual.

Marco teórico

Los gráficos circulares son herramientas de visualización que permiten mostrar cómo se distribuye un total en diferentes categorías, cada valor porcentual se transforma en un ángulo proporcional del círculo, de manera que la suma de todos los ángulos equivalga a 360°.

En Processing, la función ***arc(x, y, w, h, start, stop)*** permite dibujar arcos definidos por coordenadas, dimensiones y ángulos inicial y final en radianes, y para esta práctica, los porcentajes se convirtieron en ángulos multiplicando el valor por 360° y transformándolo a radianes con ***radians()***.

La variable ***lastAngulo*** se utiliza como acumulador para almacenar el ángulo en el que termina un segmento y en el que comienza el siguiente. De este modo, todos los sectores se van trazando de manera consecutiva hasta completar el círculo, y el uso de ***random(255)*** en ***fill()*** añade colores aleatorios, lo que mejora la legibilidad del gráfico.

Procedimiento

1. Se declara un arreglo de tipo ***float*** llamado ***valores*** que contiene los porcentajes que suman el 100% ***(25.0, 45.0, 5.0, 15.0 y 10.0)***.

```
float[] valores = {25.0, 45.0, 5.0, 15.0, 10.0};  // Suma 100%
```

2. En la función ***setup()***, se define el tamaño de la ventana con ***size(640, 360)***, se desactiva el borde de las figuras con ***noStroke()*** y se utiliza ***noLoop()*** para que el programa se ejecute una sola vez.

```
void setup() {
  size(640, 360);
  noStroke();
  noLoop();  // Ejecuta el codigo una vez
}
```

3. En la función ***draw()***, se establece un fondo negro con ***background(0)*** y se llama a la función ***pieChart()*** pasando como parámetros el centro de la ventana (***width/2, height/2***), el radio del círculo (***150***) y el arreglo ***valores***.

```
void draw() {
  background(0);
  pieChart(width/2, height/2, 150, valores);  // Centro y radio del circulo
}
```

4. Y en la función ***pieChart()***, se declara la variable ***lastAngulo*** para acumular el ángulo donde inicia cada nuevo segmento, se recorre el arreglo ***data*** con un bucle ***for***, de manera que para cada valor:
   - Se asigna un color aleatorio con ***fill(random(255), random(255), random(255))***.
   - Se convierte el porcentaje en ángulo en radianes utilizando la función ***radians()***.
   - Se dibuja un arco con ***arc()***, que corresponde al sector circular de cada porcentaje.
   - Se actualiza ***lastAngulo*** sumando el ángulo recién dibujado, de modo que el siguiente sector comenzara donde terminó el anterior.

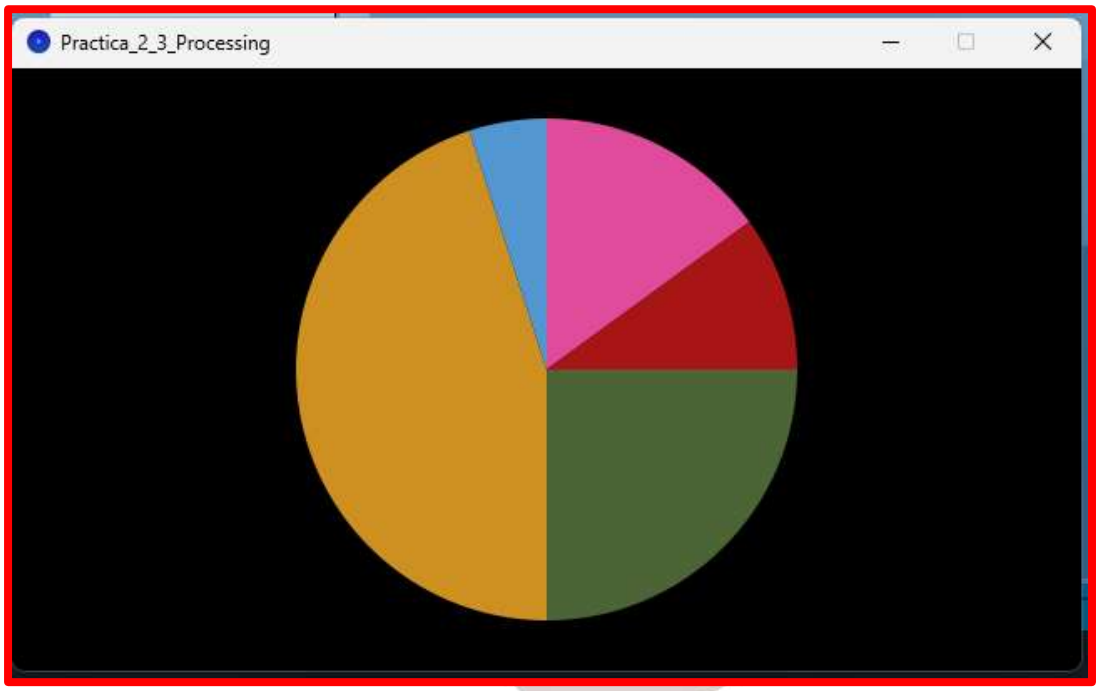
```
void pieChart(float cx, float cy, float radio, float[] data) {
  float lastAngulo = 0;
  for (int i = 0; i < data.length; i++) {
    // Color aleatorio para cada segmento
    fill(random(255), random(255), random(255));

    // Convertir porcentaje a ángulo en radianes
    float angulo = radians(data[i] * 360 / 100.0);

    // Dibujar arco
    arc(cx, cy, radio * 2, radio * 2, lastAngulo, lastAngulo + angulo);

    lastAngulo += angulo;
  }
}
```

Materiales y equipos utilizados

- Laptop
- Processing
- Página oficial de Processing

Resultados

El programa generó un gráfico circular en el que se representaron cinco valores que suman el 100% y cada sector tuvo un tamaño proporcional a su porcentaje, con colores aleatorios que facilitaron distinguirlos visualmente.



Análisis

La práctica permitió comprender cómo los porcentajes pueden transformarse en ángulos y representarse gráficamente con Processing. Se comprobó que la suma de los ángulos equivalió a un círculo completo y que el uso de ***arc()*** simplifica la construcción de gráficos estadísticos y el resultado demuestra que Processing no solo es útil para graficar funciones matemáticas, sino también para representar datos de manera visual e interactiva.

La asignación de colores aleatorios añadió dinamismo, aunque también se podría mejorar definiendo paletas de colores más específicas según el tipo de datos.

Discusión

A diferencia de las prácticas anteriores, donde se trabajó con funciones trigonométricas para representar ondas o polígonos, aquí se aplicaron los conceptos de ángulos y proporciones para mostrar datos. La práctica resalta cómo las matemáticas, en este caso la regla de tres simple y la conversión a radianes, se aplican de manera directa en la programación gráfica.

Este ejercicio también refleja la importancia de la visualización de información en la vida cotidiana, ya que los gráficos circulares se usan ampliamente en áreas como estadística, negocios y ciencias sociales.

Conclusión

El programa cumplió con éxito el objetivo de representar porcentajes en un gráfico circular y se comprobó que Processing es una herramienta sencilla y eficaz para visualizar datos, permitiendo transformar un arreglo numérico en una figura clara y comprensible.
La práctica refuerza la conexión entre matemáticas y programación, mostrando cómo conceptos básicos como porcentajes y ángulos pueden trasladarse al ámbito computacional para generar visualizaciones útiles.

Referencias

- Processing Foundation. (n.d.). Pie Chart / Examples. Processing.org. https://processing.org/examples/piechart.html
- Processing Foundation. (n.d.). arc() / Reference. Processing.org. https://processing.org/reference/arc_.html