Modul 9 : Processing untuk Struktur Data

9.1 Tujuan

Mahasiswa mampu menggunakan software processing untuk struktur data dan penggambaran elemen.

9.2 Alat & Bahan

1. Komputer/Laptop
2. Software Processing (download di processing.org)

9.3 Prosedur Praktikum

1. Conditional

```
size(640, 360);
background(0);

for(int i = 10; i < width; i += 10) {
  // If 'i' divides by 20 with no remainder draw
  // the first line, else draw the second line
  if((i % 20) == 0) {
    stroke(255);
    line(i, 80, i, height/2);
  } else {
    stroke(153);
    line(i, 20, i, 180);
  }
}
```

2. Coordinates

```
// Sets the screen to be 640 pixels wide and 360 pixels high
size(640, 360);

// Set the background to black and turn off the fill color
background(0);
noFill();

// The two parameters of the point() method each specify coordinates.
// The first parameter is the x-coordinate and the second is the Y
stroke(255);
point(width * 0.5, height * 0.5);
point(width * 0.5, height * 0.25);

// Coordinates are used for drawing all shapes, not just points.
// Parameters for different functions are used for different purposes.
// For example, the first two parameters to line() specify
// the coordinates of the first endpoint and the second two parameters
// specify the second endpoint
stroke(0, 153, 255);
line(0, height*0.33, width, height*0.33);

// By default, the first two parameters to rect() are the
```

```
// coordinates of the upper-left corner and the second pair
// is the width and height
stroke(255, 153, 0);
rect(width*0.25, height*0.1, width * 0.5, height * 0.8);
```

3. Logical Operator

```
size(640, 360);
background(126);

boolean test = false;

for (int i = 5; i <= height; i += 5) {
  // Logical AND
  stroke(0);
  if((i > 35) && (i < 100)) {
    line(width/4, i, width/2, i);
    test = false;
  }

  // Logical OR
  stroke(76);
  if ((i <= 35) || (i >= 100)) {
    line(width/2, i, width, i);
    test = true;
  }

  // Testing if a boolean value is "true"
  // The expression "if(test)" is equivalent to "if(test == true)"
  if (test) {
    stroke(0);
    point(width/3, i);
  }

  // Testing if a boolean value is "false"
  // The expression "if(!test)" is equivalent to "if(test == false)"
  if (!test) {
    stroke(255);
    point(width/4, i);
  }
}
```

4. Width and Height

```
void setup() {
  size(640, 360);
}

void draw() {
  background(127);
  noStroke();
  for (int i = 0; i < height; i += 20) {
    fill(129, 206, 15);
    rect(0, i, width, 10);
    fill(255);
    rect(i, 0, 10, height);
  }
```

```
}
```

5. No Loop

```
float y;

// The statements in the setup() function
// execute once when the program begins
void setup()
{
  size(640, 360);  // Size should be the first statement
  stroke(255);     // Set line drawing color to white
  noLoop();

  y = height * 0.5;
}

// The statements in draw() are executed until the
// program is stopped. Each statement is executed in
// sequence and after the last line is read, the first
// line is executed again.
void draw()
{
  background(0);   // Set the background to black
  y = y - 1;
  if (y < 0) { y = height; }
  line(0, y, width, y);
}
```

6. Looping

```
float y = 100;

// The statements in the setup() function
// run once when the program begins
void setup() {
  size(640, 360);  // Size should be the first statement
  stroke(255);     // Set stroke color to white
  noLoop();

  y = height * 0.5;
}

// The statements in draw() are run until the
// program is stopped. Each statement is run in
// sequence and after the last line is read, the first
// line is run again.
void draw() {
  background(0);   // Set the background to black
  line(0, y, width, y);

  y = y - 1;
  if (y < 0) {
    y = height;
  }
}

void mousePressed() {
  loop();
```

```
}
```

7. Redraw

```
float y;

// The statements in the setup() function
// execute once when the program begins
void setup() {
  size(640, 360);  // Size should be the first statement
  stroke(255);     // Set line drawing color to white
  noLoop();
  y = height * 0.5;
}

// The statements in draw() are executed until the
// program is stopped. Each statement is executed in
// sequence and after the last line is read, the first
// line is executed again.
void draw() {
  background(0);   // Set the background to black
  y = y - 4;
  if (y < 0) { y = height; }
  line(0, y, width, y);
}

void mousePressed() {
  redraw();
}
```

8. Functions

```
void setup() {
  size(640, 360);
  background(51);
  noStroke();
  noLoop();
}
void draw() {
  drawTarget(width*0.25, height*0.4, 200, 4);
  drawTarget(width*0.5, height*0.5, 300, 10);
  drawTarget(width*0.75, height*0.3, 120, 6);
}

void drawTarget(float xloc, float yloc, int size, int num) {
  float grayvalues = 255/num;
  float steps = size/num;
  for (int i = 0; i < num; i++) {
    fill(i*grayvalues);
    ellipse(xloc, yloc, size - i*steps, size - i*steps);
  }
}
```

9. Recursion

```
void setup() {
  size(640, 360);
  background(51);
  noStroke();
```

```
  noLoop();
}

void draw() {
  drawTarget(width*0.25, height*0.4, 200, 4);
  drawTarget(width*0.5, height*0.5, 300, 10);
  drawTarget(width*0.75, height*0.3, 120, 6);
}

void drawTarget(float xloc, float yloc, int size, int num) {
  float grayvalues = 255/num;
  float steps = size/num;
  for (int i = 0; i < num; i++) {
    fill(i*grayvalues);
    ellipse(xloc, yloc, size - i*steps, size - i*steps);
  }
}
```

10. Create Graphics

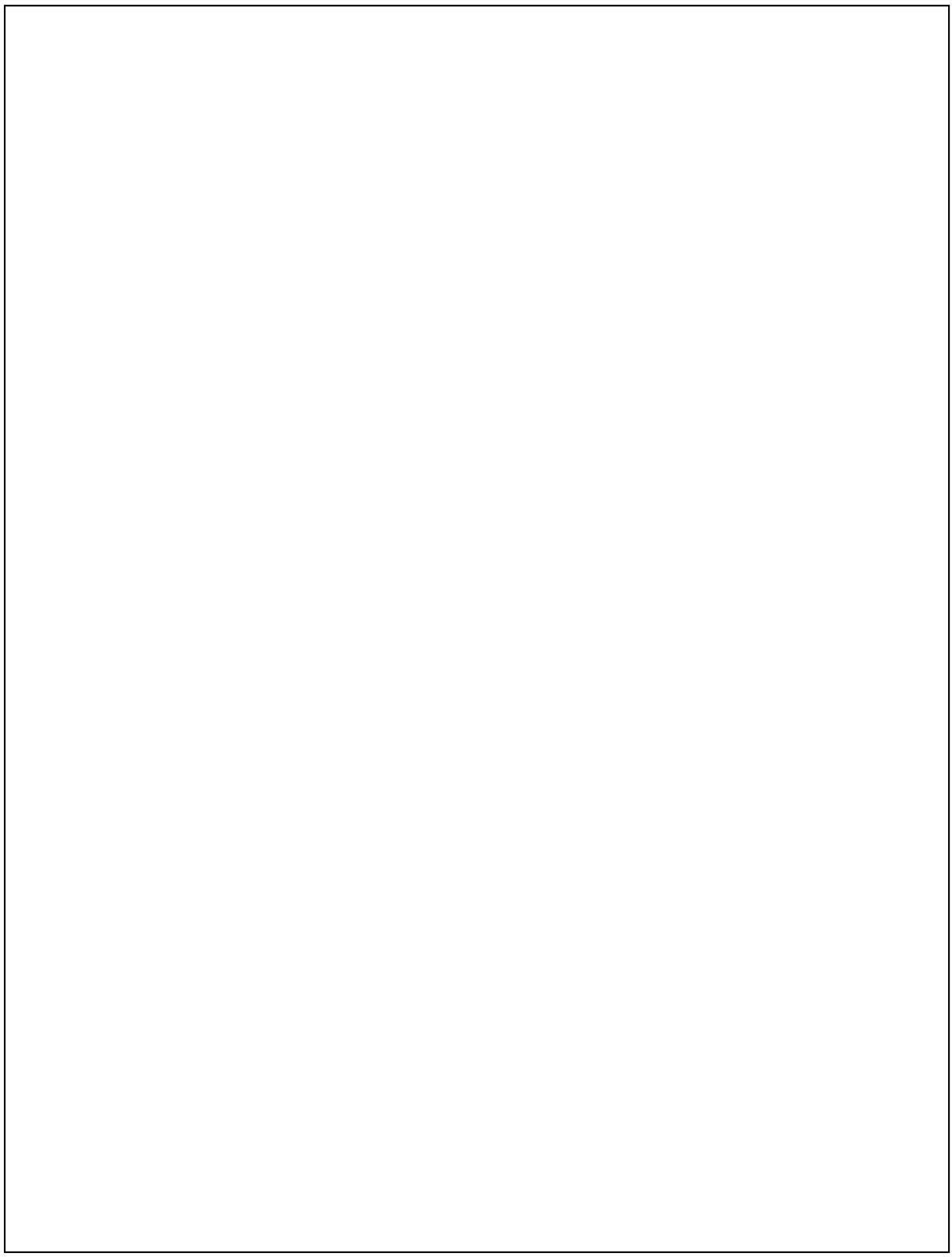
```
PGraphics pg;

void setup() {
  size(640, 360);
  pg = createGraphics(400, 200);
}

void draw() {
  fill(0, 12);
  rect(0, 0, width, height);
  fill(255);
  noStroke();
  ellipse(mouseX, mouseY, 60, 60);

  pg.beginDraw();
  pg.background(51);
  pg.noFill();
  pg.stroke(255);
  pg.ellipse(mouseX-120, mouseY-60, 60, 60);
  pg.endDraw();

  // Draw the offscreen buffer to the screen with image()
  image(pg, 120, 60);
}
```

9.4 Latihan

1. Buat 3 lingkaran dengan ukuran yang sama
2. 3 lingkaran ini bergerak berputar pada satu poros

9.5 Jurnal

Capture dan dan beri komentar dan keterangan Hasil Eksekusi syntax:



DAFTAR PUSTAKA

- https://processing.org/