

PERTEMUAN 4 GODOT
LAPORAN PRAKTIKUM

Disusun untuk memenuhi tugas Mata Kuliah Komputer Grafik

Disusun oleh
Nazwa Fitriyani Zahra 211511051



PROGRAM STUDI D3 TEKNIK INFORMATIKA
JURUSAN TEKNIK KOMPUTER DAN INFORMATIKA
POLITEKNIK NEGERI BANDUNG
2022

DAFTAR ISI

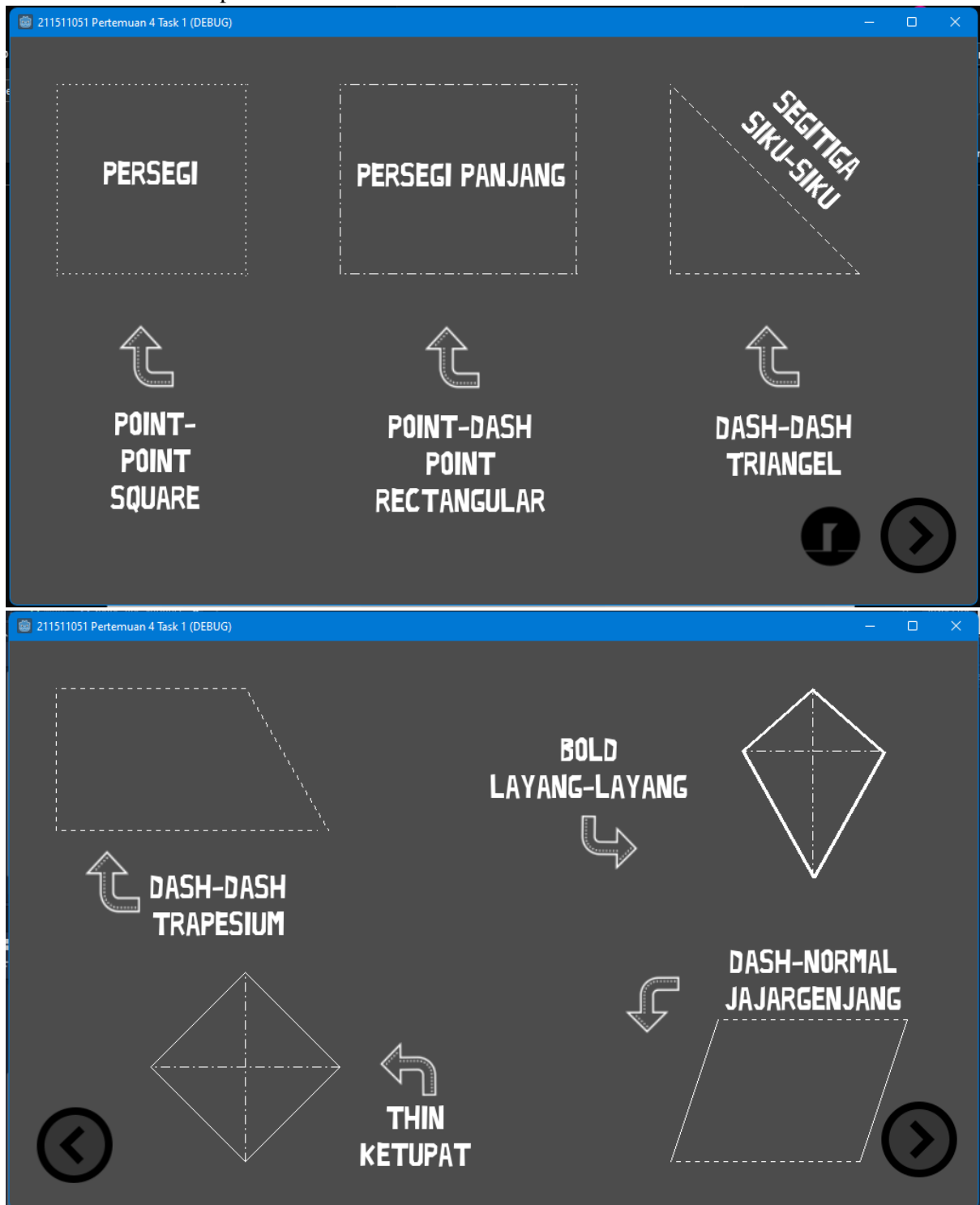
DAFTAR ISI.....	2
DAFTAR REFERENSI.....	3
A. Task 1.....	4
B. Task 2.....	11
C. Task 3.....	12
KOTRETAN.....	13
LESSON LEARN.....	15
CURHAT DOSEN	16

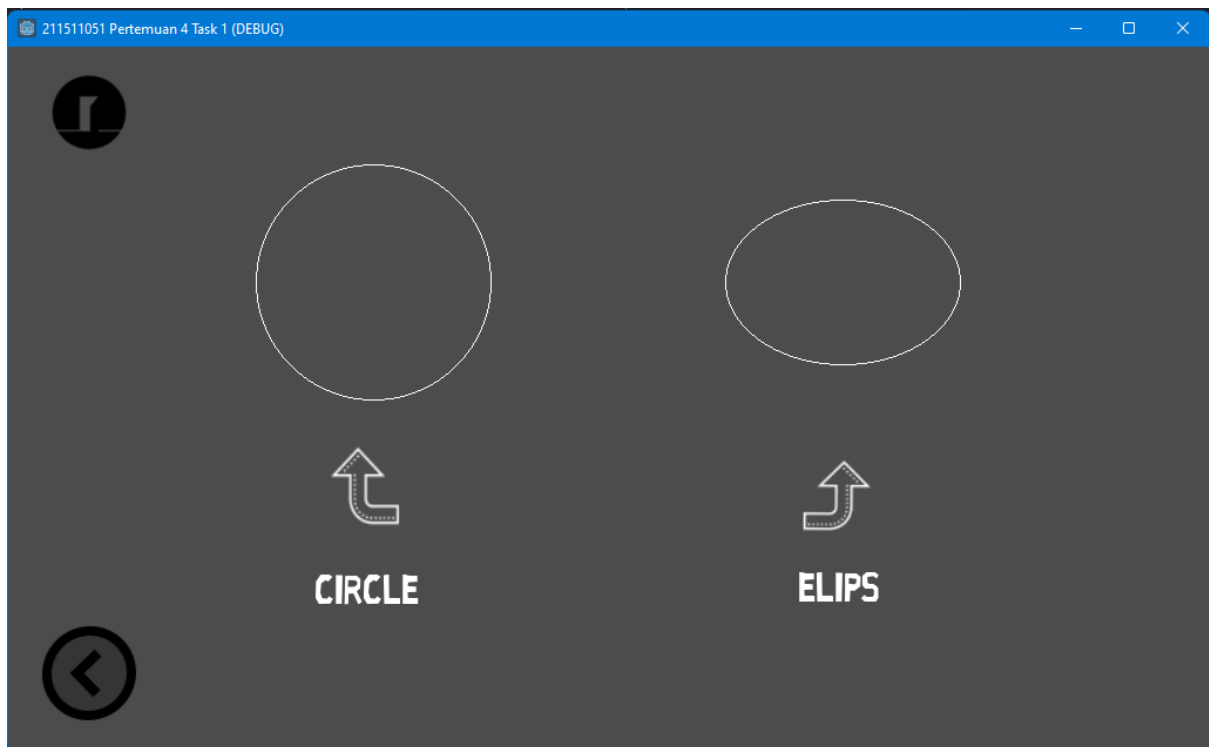
DAFTAR REFERENSI

- <https://docs.godotengine.org/en/3.5/index.html>
- <https://github.com/godotengine/godot-demo-projects>
- <https://docs.godotengine.org/en/3.5/community/tutorials.html>
- https://www.youtube.com/watch?v=ZUPBoqC_X_o
- YouTube Channel : Kelas Terbuka – Godot Tutorial
- https://docs.godotengine.org/en/stable/tutorials/2d/custom_drawing_in_2d.html
- https://docs.godotengine.org/en/stable/tutorials/2d/custom_drawing_in_2d.html

A. Task 1

Berikut ini hasil Manipulasi Garis dan Bentuk Dasar





Berikut Source Code dari hasil diatas

//line.gd

Setiap tipe line dibuat function baru seperti dibawah ini :

```

1  extends Node2D
2
3
4  # Declare member variables here. Examples:
5  # var a = 2
6  # var b = "text"
7
8
9  # Called when the node enters the scene tree for the first time.
10 ~ func _ready():
11     # Replace with function body.
12     var color = Color(1,1,1,1)
13
14 # Called every frame. 'delta' is the elapsed time since the previous frame.
15 #func _process(delta):
16     # pass
17 ~ func put_pixel(x, y, color):
18     draw_primitive(PoolVector2Array([Vector2(x, y)]),
19         PoolColorArray([color]),
20         PoolVector2Array())
21
22 ~ func put_pixel_all(dot: PoolVector2Array, color):
23     for i in dot.size():
24         put_pixel(dot[i].x, dot[i].y, color)
25
26

```

```

27 #line_dda
28 ▾ func lineDDA(xa : float, ya : float, xb : float, yb : float):
29   » var dx = xb - xa
30   » var dy = yb - ya
31   » var steps
32   » var xIncrement
33   » var yIncrement
34   » var x = xa
35   » var y = ya
36   » var res = PoolVector2Array()
37   »
38 ▾ » if (abs(dx) > abs(dy)) :
39   » » steps = abs(dx)
40 ▾ » else :
41   » » steps = abs(dy)
42   » »
43   » xIncrement = dx/ float(steps)
44   » yIncrement = dy/ float(steps)
45   » res.append(Vector2(round(x), round(y)))
46   »
47 ▾ » for k in steps:
48   » » x += xIncrement
49   » » y += yIncrement
50   » » res.append(Vector2(round(x), round(y)))
51   » #Dipanggil agar garis tampil
52   » put_pixel_all(res, color)
53   » return res
54

```

```

57 ▾ func lineBres(xa, ya, xb, yb):
58   » var dx = abs(xa - xb)
59   » var dy = abs(ya - yb)
60   » var p = 2 * dy - dx
61   » var twoDy = 2 * dy
62   » var twoDyDx = 2 * (dy - dx)
63   » var x
64   » var y
65   » var xEnd
66   » var k = 1
67   » var res = PoolVector2Array()
68   »
69 ▾ » if xa > xb :
70   » » x = xb
71   » » y = yb
72   » » xEnd = xa
73 ▾ » else :
74   » » x = xa
75   » » y = ya
76   » » xEnd = xb
77   »
78   » res.append(Vector2(round(x), round(y)))
79   »

```

```

80 ▾ » while x < xEnd :
81   » » x += 1
82 ▾ » if p < 0 :
83   » » » p += twoDy
84 ▾ » else :
85   » » » y += 1
86   » » » p += twoDyDx
87   » » res.append(Vector2(round(x), round(y)))
88   » » k += 1
89   » #Dipanggil agar garis tampil
90   » put_pixel_all(res, color)
91   » return res
92

```

```

95 ~ func dash_dash(xa : float, ya : float, xb : float, yb : float):
96 ~     var dx = xb - xa
97 ~     var dy = yb - ya
98 ~     var steps
99 ~     var xIncrement
100 ~     var yIncrement
101 ~     var x = xa
102 ~     var y = ya
103 ~     var res = PoolVector2Array()
104 ~
105 ~
106 ~     if (abs(dx) > abs(dy)) :
107 ~         steps = abs(dx)
108 ~     else :
109 ~         steps = abs(dy)
110 ~
111 ~     xIncrement = dx/ float(steps)
112 ~     yIncrement = dy/ float(steps)
113 ~     res.append(Vector2(round(x), round(y)))

```

```

114 ~     if(ya == yb):
115 ~         for k in steps :
116 ~             if(k % 5 == 0):
117 ~                 x += 5
118 ~                 x += xIncrement
119 ~                 y += yIncrement
120 ~                 res.append(Vector2(round(x), round(y)))
121 ~             if(x >= xb):
122 ~                 break
123 ~             print(k)
124 ~         elif(xa == xb):
125 ~             for k in steps :
126 ~                 if(k % 5 == 0):
127 ~                     y += 5
128 ~                     y += yIncrement
129 ~                     res.append(Vector2(round(x), round(y)))
130 ~                 if(x >= xb && y >= yb):
131 ~                     break
132 ~                 print(k)
133 ~             else:
134 ~                 for k in steps:
135 ~                     if(k % 5 == 0):
136 ~                         x += 3
137 ~                         y += 3
138 ~                         x += xIncrement
139 ~                         y += yIncrement
140 ~                         put_pixel(round(x), round(y), color)
141 ~                     if(x >= xb && y >= yb):
142 ~                         break
143 ~                     print(k)
144 ~             #Dipanggil agar garis tampil
145 ~             put_pixel all(res, color)

```

```

148 ~ func point_dash(xa : float, ya : float, xb : float, yb : float):
149 ~     var dx = xb - xa
150 ~     var dy = yb - ya
151 ~     var steps
152 ~     var xIncrement
153 ~     var yIncrement
154 ~     var x = xa
155 ~     var y = ya
156 ~     var res = PoolVector2Array()
157 ~     var tampung = 0
158 ~
159 ~     if (abs(dx) > abs(dy)) :
160 ~         steps = abs(dx)
161 ~     else :
162 ~         steps = abs(dy)
163 ~
164 ~     xIncrement = dx/ float(steps)
165 ~     yIncrement = dy/ float(steps)
166 ~     res.append(Vector2(round(x), round(y)))

```

```

167 ~> | if(ya == yb):
168 ~> | | for k in steps:
169 ~> | | | if(int(x) % 10 == 0):
170 ~> | | | | x+= 4
171 ~> | | | | tampung = x
172 ~> | | | if(x == tampung + 2):
173 ~> | | | | x+= 4
174 ~> | | | x += xIncrement
175 ~> | | | y += yIncrement
176 ~> | | | res.append(Vector2(round(x), round(y)))
177 ~> | | | if(x >= xb && y >= yb):
178 ~> | | | | break
179 ~> | | else :
180 ~> | | | for k in steps:
181 ~> | | | | if(int(y) % 10 == 0):
182 ~> | | | | y+= 4
183 ~> | | | | tampung = y
184 ~> | | | if(y == tampung + 2):
185 ~> | | | | y+= 4
186 ~> | | | y += yIncrement
187 ~> | | | res.append(Vector2(round(x), round(y)))
188 ~> | | | if(x >= xb && y >= yb):
189 ~> | | | | break
190 ~> | | #Dipanggil agar garis tampil
191 ~> | | put_pixel_all(res, color)
192 ~> | | return res
193 ~> |

```

```

194 ~> func point_point(xa : float, ya : float, xb : float, yb : float):
195 ~> | var dx = xb - xa
196 ~> | var dy = yb - ya
197 ~> | var steps
198 ~> | var xIncrement
199 ~> | var yIncrement
200 ~> | var x = xa
201 ~> | var y = ya
202 ~> | var res = PoolVector2Array()
203 ~> |
204 ~> | if (abs(dx) > abs(dy)) :
205 ~> | | steps = abs(dx)
206 ~> | else :
207 ~> | | steps = abs(dy)
208 ~> |
209 ~> | xIncrement = dx/ float(steps)
210 ~> | yIncrement = dy/ float(steps)
211 ~> | res.append(Vector2(round(x), round(y)))
212 ~> |

```

```

213 ~> | if(ya == yb):
214 ~> | | for k in steps :
215 ~> | | | if(k % 2 == 0):
216 ~> | | | | x += 5
217 ~> | | | x += xIncrement
218 ~> | | | y += yIncrement
219 ~> | | | res.append(Vector2(round(x), round(y)))
220 ~> | | | if(x >= xb && y >= yb):
221 ~> | | | | break
222 ~> | | | print(k)
223 ~> | | elif(xa == xb):
224 ~> | | | for k in steps :
225 ~> | | | | if(k % 2 == 0):
226 ~> | | | | y += 5
227 ~> | | | y += yIncrement
228 ~> | | | res.append(Vector2(round(x), round(y)))
229 ~> | | | if(x >= xb && y >= yb):
230 ~> | | | | break
231 ~> | | | print(k)
232 ~> |
233 ~> | #Dipanggil agar garis tampil
234 ~> | put_pixel_all(res, color)
235 ~> | return res
236 ~> |

```



```

237 ~ func bold(xa : float, ya : float, xb : float, yb : float):
238   ~ var dx = xb - xa
239   ~ var dy = yb - ya
240   ~ var steps
241   ~ var xIncrement
242   ~ var yIncrement
243   ~ var x = xa
244   ~ var y = ya
245   ~ var res = PoolVector2Array()
246
247 ~ if (abs(dx) > abs(dy)) :
248   ~ steps = abs(dx)
249 ~ else :
250   ~ steps = abs(dy)
251   ~
252   ~ xIncrement = dx/ float(steps)
253   ~ yIncrement = dy/ float(steps)
254   ~ res.append(Vector2(round(x), round(y)))
255 ~ for i in steps:
256   ~ x += xIncrement #11
257   ~ y += yIncrement #10
258   ~ res.append(Vector2(round(x), round(y)))
259 ~ for n in 4:
260 ~   ~ if dx > dy:
261     ~ res.append(Vector2(round(x), round(y+n)))
262 ~   ~ else:
263     ~ res.append(Vector2(round(x+n), round(y)))
264   ~
265   ~ #Dipanggil agar garis tampil
266   ~ put_pixel_all(res, color)
267   ~ return res

```

//shape.gd

Dari line.gd yang telah dibuat, dikembangkan untuk digunakan dalam membuat bidang.

Bidang dibuat dengan mengextend fungsi yang ada di line.gd. Berikut adalah source code pembuatan bidang 2d

```

28 ~ func trapesium_siku_siku(titik_awal: Vector2, atas, tinggi, bawah):
29   ~ var res = PoolVector2Array()
30   ~ dash_dash(titik_awal.x, titik_awal.y, titik_awal.x + atas, titik_awal.y) #sisi atas
31   ~ dash_dash(titik_awal.x, titik_awal.y + tinggi, titik_awal.x + bawah + round(abs(bawah-atas)/2), titik_awal.y + tinggi) #sisi bawah
32   ~ dash_dash(titik_awal.x, titik_awal.y, titik_awal.x, titik_awal.y + tinggi) #sisi kiri
33   ~ dash_dash(titik_awal.x + atas, titik_awal.y, titik_awal.x + bawah, titik_awal.y + tinggi) #sisi kanan
34   ~ return res
35
36 ~ func jajargenjang(titik_awal: Vector2, panjang, tinggi, geser):
37   ~ var res = PoolVector2Array()
38   ~ dash_dash(titik_awal.x, titik_awal.y, titik_awal.x + panjang, titik_awal.y) #sisi atas
39   ~ dash_dash(titik_awal.x - geser, titik_awal.y + tinggi, (titik_awal.x + panjang - geser), titik_awal.y + tinggi) #sisi bawah
40   ~ res.append_array(lineDDA(titik_awal.x, titik_awal.y, (titik_awal.x - geser), titik_awal.y + tinggi)) #sisi kiri
41   ~ res.append_array(lineDDA(titik_awal.x + panjang, titik_awal.y, ((titik_awal.x + panjang) - geser), titik_awal.y + tinggi)) #sisi k
42   ~ return res
43
44 ~ func ketupat (titik_awal: Vector2, diagonal):
45   ~ var res = PoolVector2Array()
46   ~ point_dash(titik_awal.x, titik_awal.y, titik_awal.x, titik_awal.y + diagonal)
47   ~ point_dash(titik_awal.x - (diagonal/2), titik_awal.y + (diagonal/2), titik_awal.x + (diagonal/2), titik_awal.y + (diagonal/2))
48   ~ res.append_array(lineDDA(titik_awal.x, titik_awal.y, titik_awal.x - (diagonal/2), titik_awal.y + (diagonal/2)))
49   ~ res.append_array(lineDDA(titik_awal.x, titik_awal.y, titik_awal.x + (diagonal/2), titik_awal.y + (diagonal/2)))
50   ~ res.append_array(lineDDA(titik_awal.x - (diagonal/2), titik_awal.y + (diagonal/2), titik_awal.x, titik_awal.y + diagonal))
51   ~ res.append_array(lineDDA(titik_awal.x + (diagonal/2), titik_awal.y + (diagonal/2), titik_awal.x, titik_awal.y + diagonal))
52   ~ return res
53

```

```

54 ▾ func layanglayang (titik_awal: Vector2, diagonal_a, diagonal_b):
55   ▸ var res = PoolVector2Array()
56   ▸ point_dash(titik_awal.x, titik_awal.y, titik_awal.x, titik_awal.y + diagonal_a)
57   ▸ point_dash(titik_awal.x - (diagonal_b/2), titik_awal.y + (diagonal_a/3), titik_awal.x + (diagonal_b/2), titik_awal.y + (diagonal_a/3))
58   ▸ res.append_array(bold(titik_awal.x, titik_awal.y, titik_awal.x - (diagonal_b/2), titik_awal.y + (diagonal_a/3)))
59   ▸ res.append_array(bold(titik_awal.x, titik_awal.y, titik_awal.x + (diagonal_b/2), titik_awal.y + (diagonal_a/3)))
60   ▸ res.append_array(bold(titik_awal.x - (diagonal_b/2), titik_awal.y + (diagonal_a/3), titik_awal.x, titik_awal.y + diagonal_a))
61   ▸ res.append_array(bold(titik_awal.x + (diagonal_b/2), titik_awal.y + (diagonal_a/3), titik_awal.x, titik_awal.y + diagonal_a))
62   ▸ return res
63
64 ▾ func circleMidPoint(xCenter, yCenter, radius, color):
65   ▸ var x = 0
66   ▸ var y = radius
67   ▸ var p = 1 - radius
68
69   ▸ circlePlotPoints(xCenter, yCenter, x, y, color)
70   ▸
71   ▾ while (x < y):
72     ▸ x += 1
73     ▸ if (p < 0):
74       ▸ p += 2*x + 1
75     ▸ else:
76       ▸ y -= 1
77       ▸ p += 2*(x-y) + 1
78     ▸ circlePlotPoints(xCenter, yCenter, x, y, color)
79
80 ▾ func circlePlotPoints(xCenter, yCenter, x, y, color):
81   ▸ put_pixel(xCenter + x, yCenter + y, color) #1
82   ▸ put_pixel(xCenter - x, yCenter + y, color) #2
83   ▸ put_pixel(xCenter + x, yCenter - y, color) #3
84   ▸ put_pixel(xCenter - x, yCenter - y, color) #4
85   ▸ put_pixel(xCenter + y, yCenter + x, color) #5
86   ▸ put_pixel(xCenter - y, yCenter + x, color) #6
87   ▸ put_pixel(xCenter + y, yCenter - x, color) #7
88   ▸ put_pixel(xCenter - y, yCenter - x, color) #8
89   ▸
90 ▾ func ellipseMidPoint(xCenter, yCenter, Rx, Ry, color):
91   ▸ var Rx2 = Rx*Rx
92   ▸ var Ry2 = Ry*Ry
93   ▸ var twoRx2 = 2*Rx2
94   ▸ var twoRy2 = 2*Ry2
95   ▸ var p
96   ▸ var x = 0
97   ▸ var y = Ry
98   ▸ var px = 0
99   ▸ var py = twoRx2*y
100  ▸
101
102  ▸ ellipsePlotPoints(xCenter, yCenter, x, y, color)
103

```

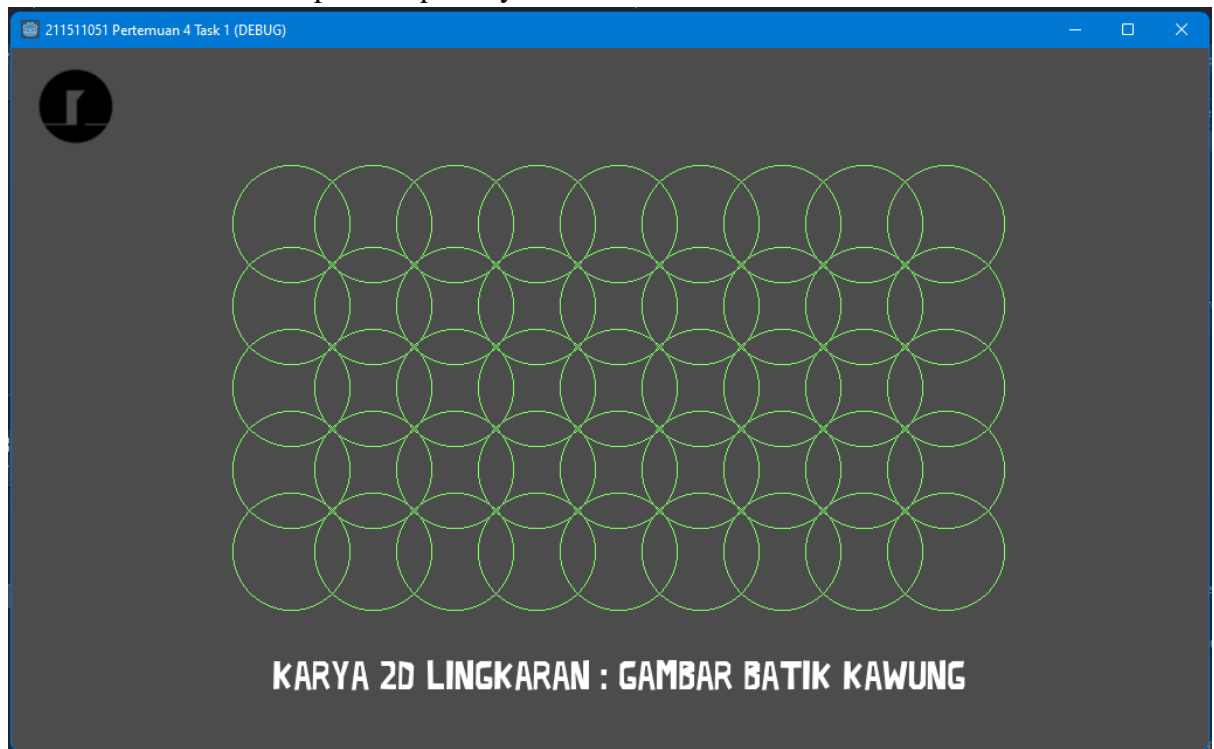
```

104 > | p = round(Ry2 - (Rx2*Ry) + (0.25 * Rx2))
105 > | while (px < py):
106 > | | x += 1
107 > | | px += twoRy2
108 > | | if (p < 0):
109 > | | | p += Ry2 + px
110 > | | else:
111 > | | | y -= 1
112 > | | | py -= twoRx2
113 > | | | p += Ry2 + px - py
114 > | | ellipsePlotPoints(xCenter, yCenter, x, y, color)
115 > | |
116 > | # Region 2 #
117 > | p = round(Ry2*(x+0.5)*(x+0.5) + Rx2*(y-1)*(y-1) - Rx2*Ry2)
118 > | while (y > 0):
119 > | | y -= 1
120 > | | py -= twoRx2
121 > | | if (p > 0):
122 > | | | p += Rx2 - py
123 > | | else :
124 > | | | x += 1
125 > | | | px += twoRy2
126 > | | | p += Rx2 - py + px
127 > | | ellipsePlotPoints(xCenter, yCenter, x, y, color)
128 > | |
129 > | func ellipsePlotPoints(xCenter, yCenter, x, y, color):
130 > | | put_pixel(xCenter + x, yCenter + y, color)
131 > | | put_pixel(xCenter - x, yCenter + y, color)
132 > | | put_pixel(xCenter + x, yCenter - y, color)
133 > | | put_pixel(xCenter - x, yCenter - y, color)
134 > |

```

B. Task 2

Berikut adalah hasil output dari pertanyaan Task 2



Karya 2D Lingkaran yang saya buat adalah berbentuk batik kawung

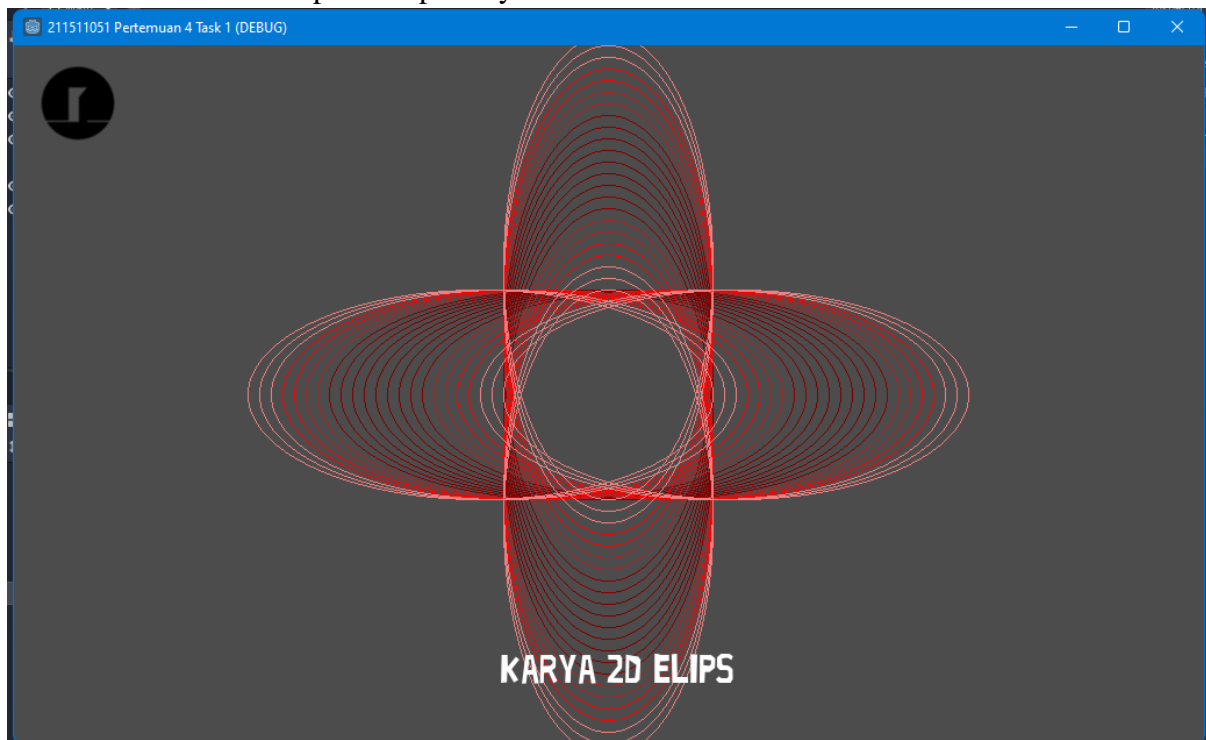
Dari library line dan shape yang telah dibuat sebelumnya, saya membuat function baru dengan nama lingkaranoverlapping yang mengextend function dari shape.gd. Berikut source codenya :

//call_overlapping.gd

```
1 extends 'res://SCRIPT/shape.gd'
2 func _ready():
3     pass # Replace with function body.
4
5
6 func lingkaran_overlapping():
7     var x = 170
8     var y = 150
9     var i = 0
10    while (i<5):
11        lingkaran_horizontal(x, y)
12        y = y + 70
13        i = i + 1
14    #lingkaran_horizontal(x, y+70)
15    #lingkaran_horizontal(x, y+140)
16    #lingkaran_horizontal(x, y+210)
17    #lingkaran_horizontal(x, y+280)
18 func lingkaran_horizontal(x, y):
19     var i = 0
20
21     while (i < 9) :
22         x = x + 70
23         circleMidPoint(x, y, 50, Color(0.5,1,0.4,1))
24         i = i + 1
25
26
27 func _draw():
28     lingkaran_overlapping()
29
```

C. Task 3

Berikut adalah hasil output dari pertanyaan Task 3



Karya 2D Elips yang saya buat adalah berupa bunga kelopak 4

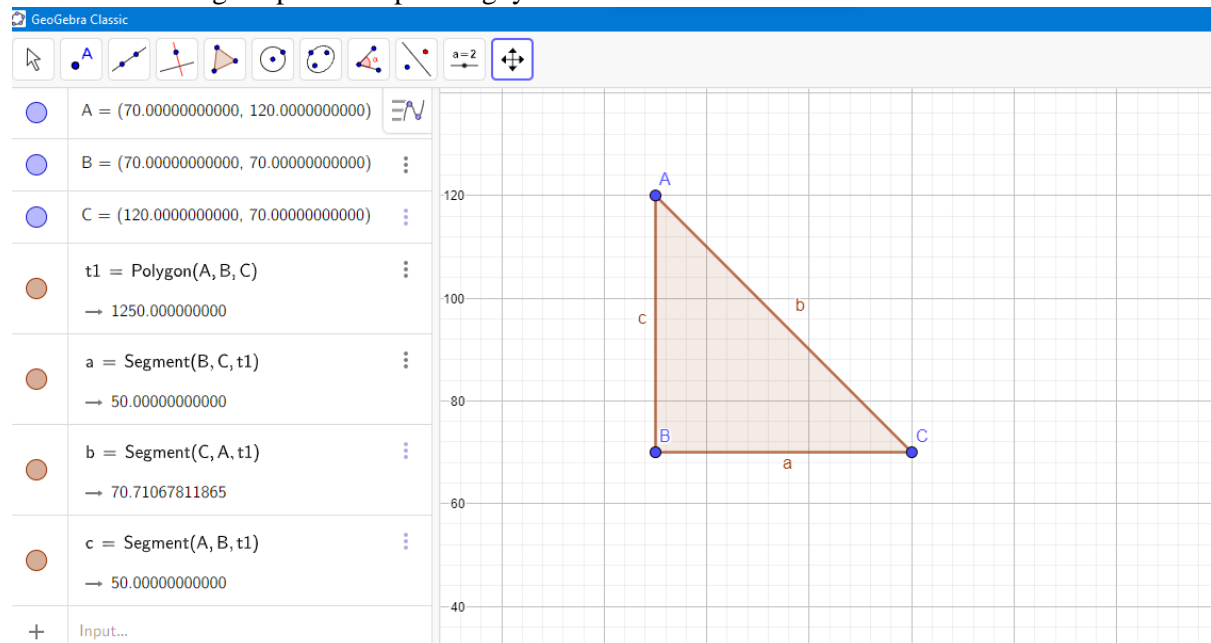
Dari library shape yang telah dibuat sebelumnya, saya membuat function baru dengan nama Elips2D yang mana isinya untuk membentuk karya elips seperti diatas ini. Berikut source codenya :

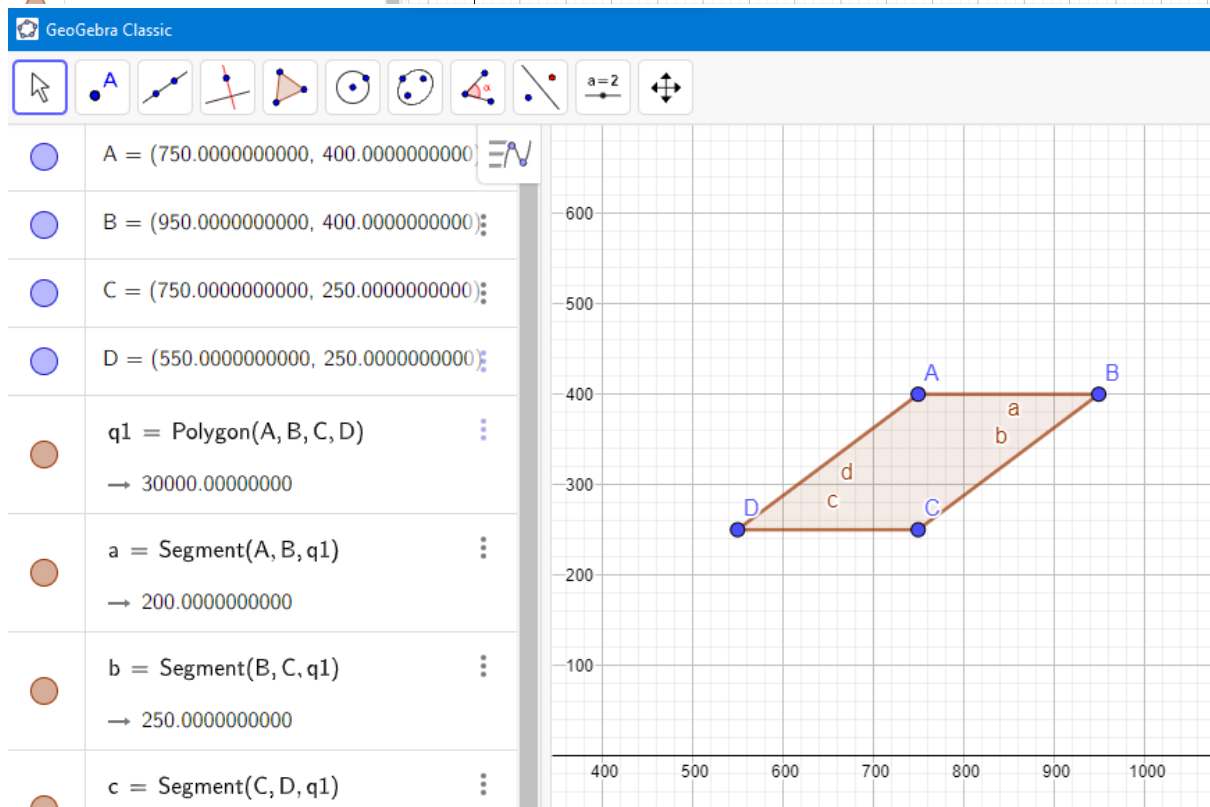
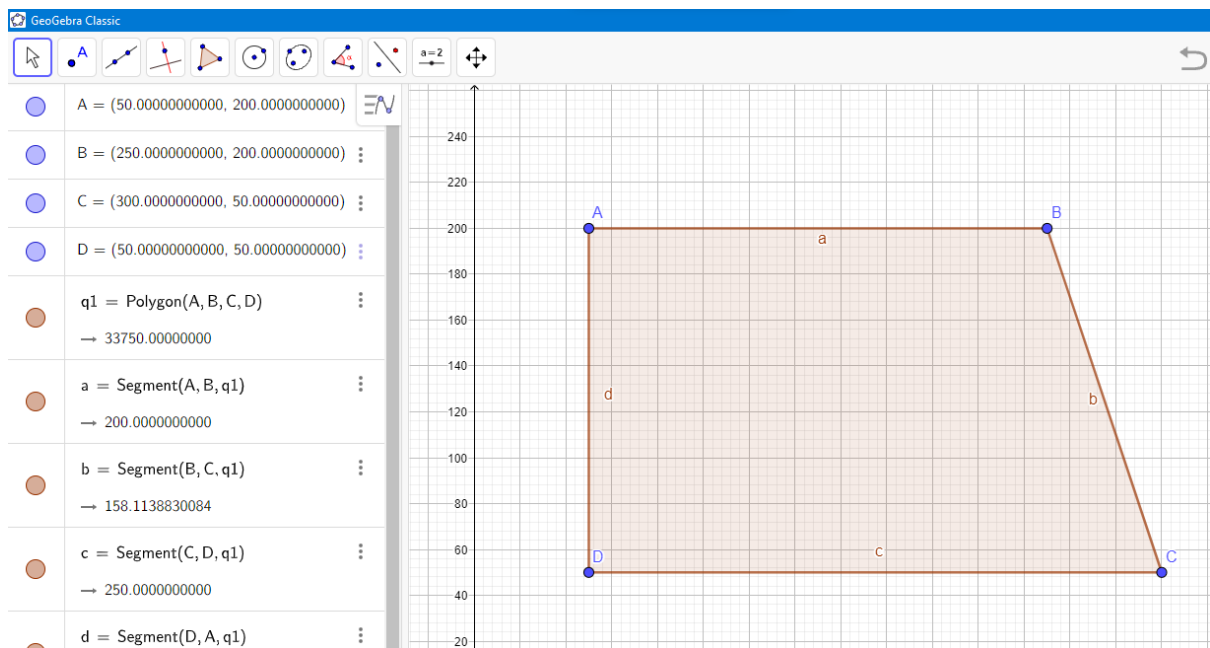
//call_elips

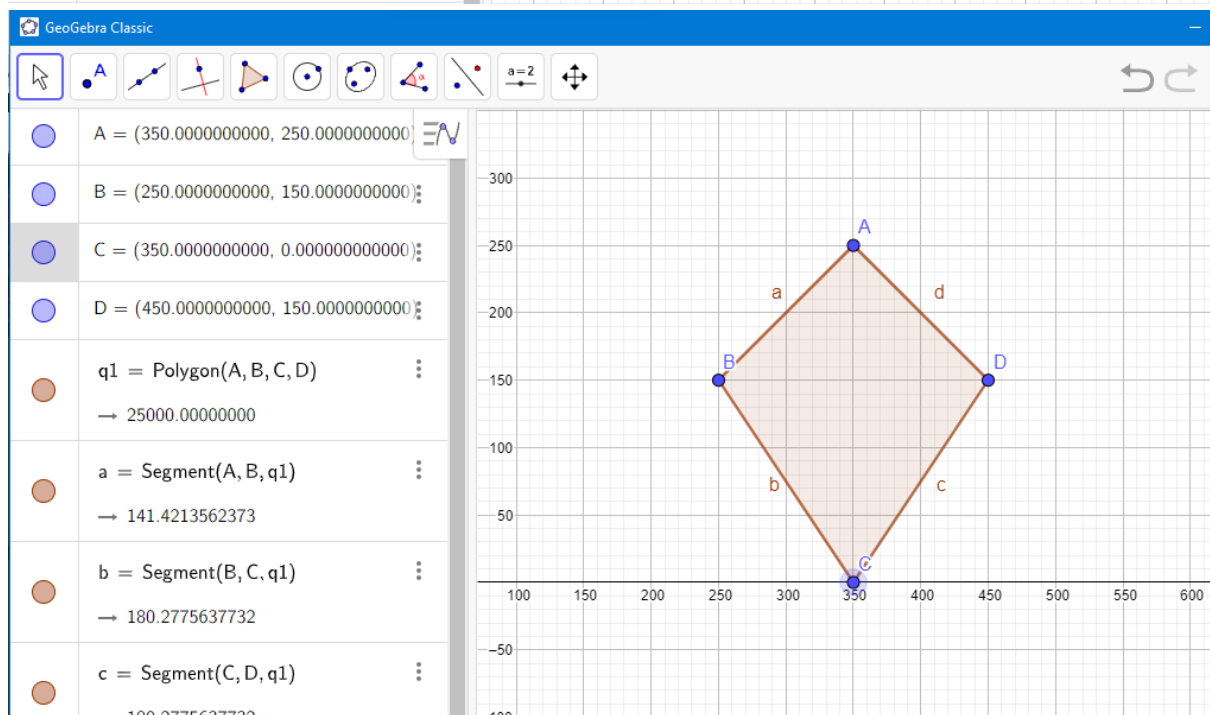
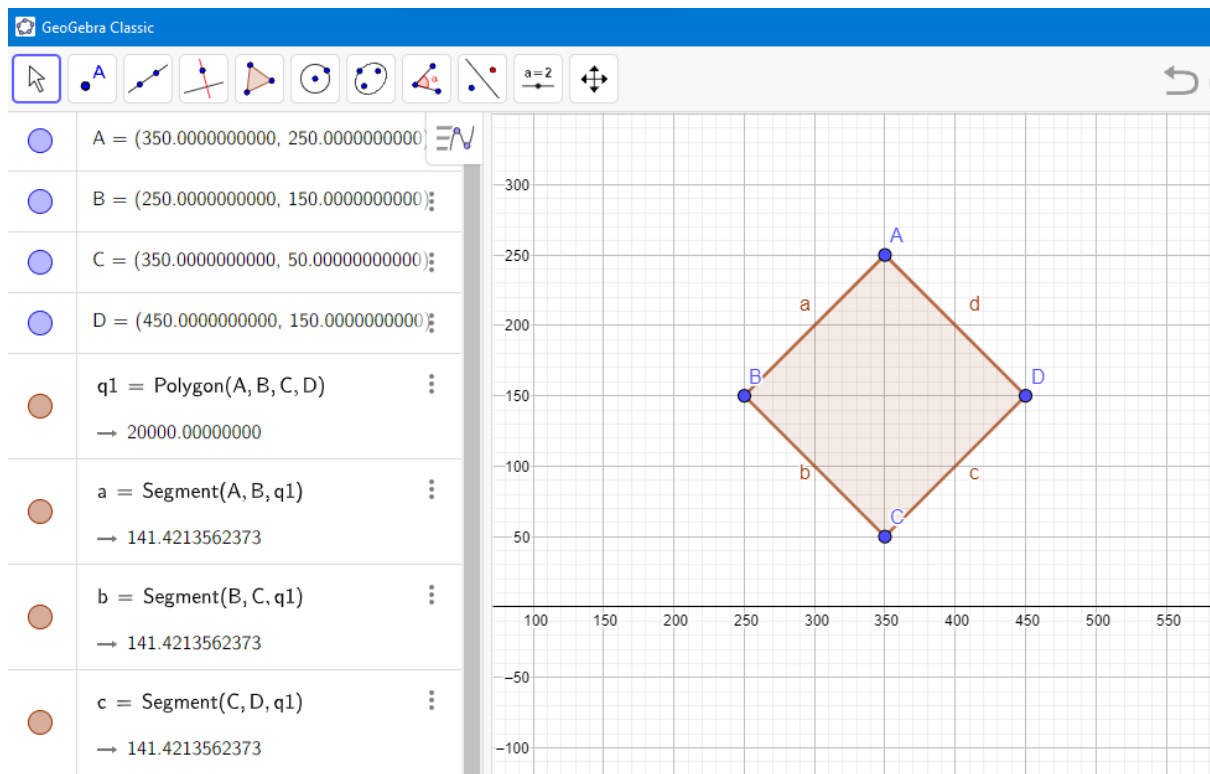
```
17 func elips2D():
18     var i = 0
19     var l = 10
20     var color = Color(0.501961, 0, 0, 1)
21     ellipseMidPoint(get_viewport().size.x/2, get_viewport().size.y/2, 200, 90, color)
22     ellipseMidPoint(get_viewport().size.x/2, get_viewport().size.y/2, 90, 200, color)
23     while (i <= 10):
24         if (i == 2):
25             color = Color( 0.545098, 0, 0, 1 )
26         elif(i == 4):
27             color = Color( 0.698039, 0.133333, 0.133333, 1 )
28         elif(i == 6):
29             color = Color( 1, 0, 0, 1 )
30         elif(i == 8):
31             color = Color( 0.941176, 0.501961, 0.501961, 1 )
32         ellipseMidPoint(get_viewport().size.x/2+l, get_viewport().size.y/2, 200, 90, color)
33         ellipseMidPoint(get_viewport().size.x/2-l, get_viewport().size.y/2, 200, 90, color)
34         ellipseMidPoint(get_viewport().size.x/2, get_viewport().size.y/2-l, 90, 200, color)
35         ellipseMidPoint(get_viewport().size.x/2, get_viewport().size.y/2+l, 90, 200, color)
36         l = l + 10
37         i = i + 1
38
39 func _draw():
40     elips2D()
41
```

KOTRETAN

Kontretan selama pembuatan beberapa bidang dasar dengan mencoba di geogebra agar lebih gampang dalam membuat garis pada setiap bidangnya







LESSON LEARN

Setelah saya mengerjakan tugas ini, materi yang diajarkan pada mata kuliah teori menjadi lebih jelas lagi. Karena di tugas sebelumnya ditugaskan mengenai manipulasi garis, jadi function yang telah digunakan bisa digunakan di dalam tugas pertemuan minggu 4 ini. Yang menjadi kendala adalah mengenai logikanya

yang sering error jadi hasilnya tidak sesuai dengan keinginan. Selain itu juga kesalahan yang sering terjadi adalah kesalahan penempatan looping yang terjadi berulang kali sehingga ketika dijalankan akan not responding aplikasinya.

CURHAT DOSEN

Untuk tugas godot ini terasa lebih terbayang karena sebelumnya telah mengerjakan task mengenai manipulasi garis. Yang paling sulit itu menurut saya adalah bagaimana mengkonversi logika yang kita punya ke dalam godot script. Mungkin hal yang membuat mudah itu ketika sudah ada contoh dari logika program bahasa lain, contohnya yang elips dan lingkaran ini. Selama mengkonversi kita jadi sambil mikir tentang logikanya dan jadi lebih paham lagi ttg perilaku gdscript kalau dibedakan dengan bahasa pemrograman lain.