

```
1 import turtle
2 count = 0
3 wn = turtle.Screen()
4 turtle.hideturtle()
5 turtle.penup()
6 turtle.backward(400)
7 turtle.pendown()
8
9 def drawseed(length):
10     len = length/3
11     turtle.forward(len)
12     turtle.left(60)
13     turtle.forward(len)
14     turtle.right(120)
15     turtle.forward(len)
16     turtle.left(60)
17     turtle.forward(len)
18
19 def recursive(count, length):
20     count -= 1
21     len = length/3
22     if count>1:
23         recursive(count, len)
24         turtle.left(60)
25         recursive(count, len)
26         turtle.right(120)
27         recursive(count, len)
28         turtle.left(60)
29         recursive(count, len)
30     else:
31         drawseed(len)
32
33 try:
34     count = int(input("Psudeo Order: "))
35     recursive(count, 2400)
36     print("Finished")
37 except ValueError:
38     raise ValueError("Order must be positive interger")
39
```

```

1 import sys
2 import turtle
3 Gcount = 0
4 wn = turtle.Screen()
5 turtle.hideturtle()
6 turtle.penup()
7 turtle.backward(250)
8 turtle.left(90)
9 turtle.forward(200)
10 turtle.right(135)
11 turtle.pendown()
12
13 def rRecursive(count, length):
14     count -= 1
15     newLength = ((length**2)/2)**0.5
16     if count > 0:
17         turtle.left(45)
18         lRecursive(count, newLength)
19         turtle.right(90)
20         rRecursive(count, newLength)
21         turtle.left(45)
22     else:
23         turtle.left(45)
24         turtle.forward(length)
25         turtle.right(90)
26         turtle.forward(length)
27         turtle.left(45)
28
29 def lRecursive(count, length):
30     count -= 1
31     newLength = ((length**2)/2)**0.5
32     if count > 0:
33         turtle.right(45)
34         lRecursive(count, newLength)
35         turtle.left(90)
36         rRecursive(count, newLength)
37         turtle.right(45)
38     else:
39         turtle.right(45)
40         turtle.forward(length)
41         turtle.left(90)
42         turtle.forward(length)
43         turtle.right(45)
44
45 try:
46     Gcount = int(input("Pseudo Order: "))
47     filename = "Order{0}PseudoDragonCurve.ps".format(Gcount)
48     print(filename)
49     rRecursive(Gcount, 350)
50     print("Finished")
51     turtle.getscreen().getcanvas().postscript(file=filename)
52 except ValueError:
53     raise ValueError("Order must be positive interger")
54
55 sys.exit()
56

```

```
1 import sys
2 import turtle
3 Gcount = 0
4 wn = turtle.Screen()
5 turtle.hideturtle()
6 turtle.penup()
7 turtle.backward(350)
8 turtle.left(90)
9 turtle.backward(300)
10 turtle.right(30)
11 turtle.pendown()
12
13 def seed(sideLength):
14     for i in range(3):
15         turtle.forward(sideLength)
16         turtle.right(120)
17
18 def recursive(count, sideLength):
19     count -= 1
20     if count > 0:
21         recursive(count, sideLength/2)
22         turtle.forward(sideLength/2)
23         recursive(count, sideLength/2)
24         turtle.right(120)
25         turtle.forward(sideLength/2)
26         turtle.left(120)
27         recursive(count, sideLength/2)
28         turtle.left(120)
29         turtle.forward(sideLength/2)
30         turtle.right(120)
31     else:
32         seed(sideLength)
33
34 try:
35     Gcount = int(input("Pseudo Order: "))
36     filename = "Order{0}PseudoSierpinskiTriangle.ps".format(Gcount)
37     print(filename)
38     recursive(Gcount, 700)
39     print("Finished")
40     turtle.getscreen().getcanvas().postscript(file=filename)
41 except ValueError:
42     raise ValueError("Order must be positive interger")
43
44 sys.exit()
45
```