Министерство высшего образования и науки

национальный исследовательский университет информационных технологий, механики и оптики

Мегафакультет компьютерных технологий и управления

факультет программной инженерии и компьютерной техники

Отчёт по лабораторной работе №2.3

«Использование встроенных средств автоматизации в приложении».

Студентка: Базилевич Мария Алексеевна

Группа № P3166

Преподаватель:

Климов Игорь Викторович

Санкт-Петербург

2022

Оглавление

[Задание 3](#_Toc122473000)

[Эскиз фрактала 3](#_Toc122473001)

[Ветка 3](#_Toc122473002)

[Снежинка 4](#_Toc122473003)

[Множество Кантора 5](#_Toc122473004)

[Снежинка Коха 6](#_Toc122473005)

[Описание расширения 7](#_Toc122473006)

[Ветка 7](#_Toc122473007)

[Снежинка 8](#_Toc122473008)

[Множество Кантора 9](#_Toc122473009)

[Снежинка Коха 10](#_Toc122473010)

[Фрактальное дерево 10](#_Toc122473011)

[Текст расширения (.inx и .py) 11](#_Toc122473012)

[Ветка 11](#_Toc122473013)

[Снежинка 13](#_Toc122473014)

[Множество Кантора 15](#_Toc122473015)

[Снежинка Коха 17](#_Toc122473016)

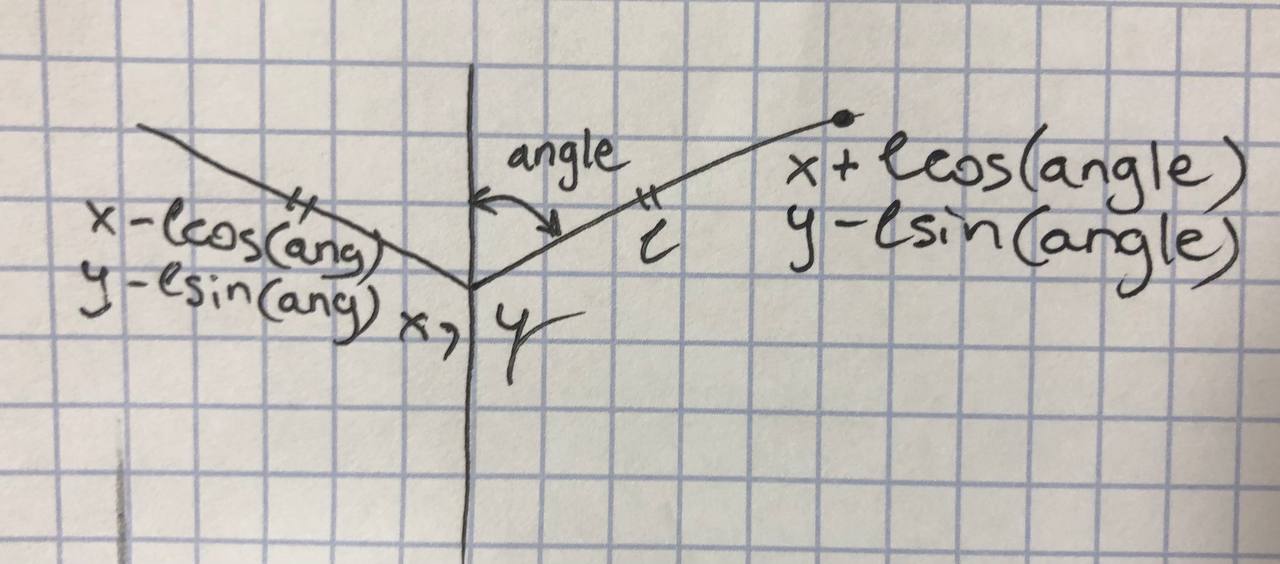
[Фрактальное дерево 19](#_Toc122473017)

# Задание

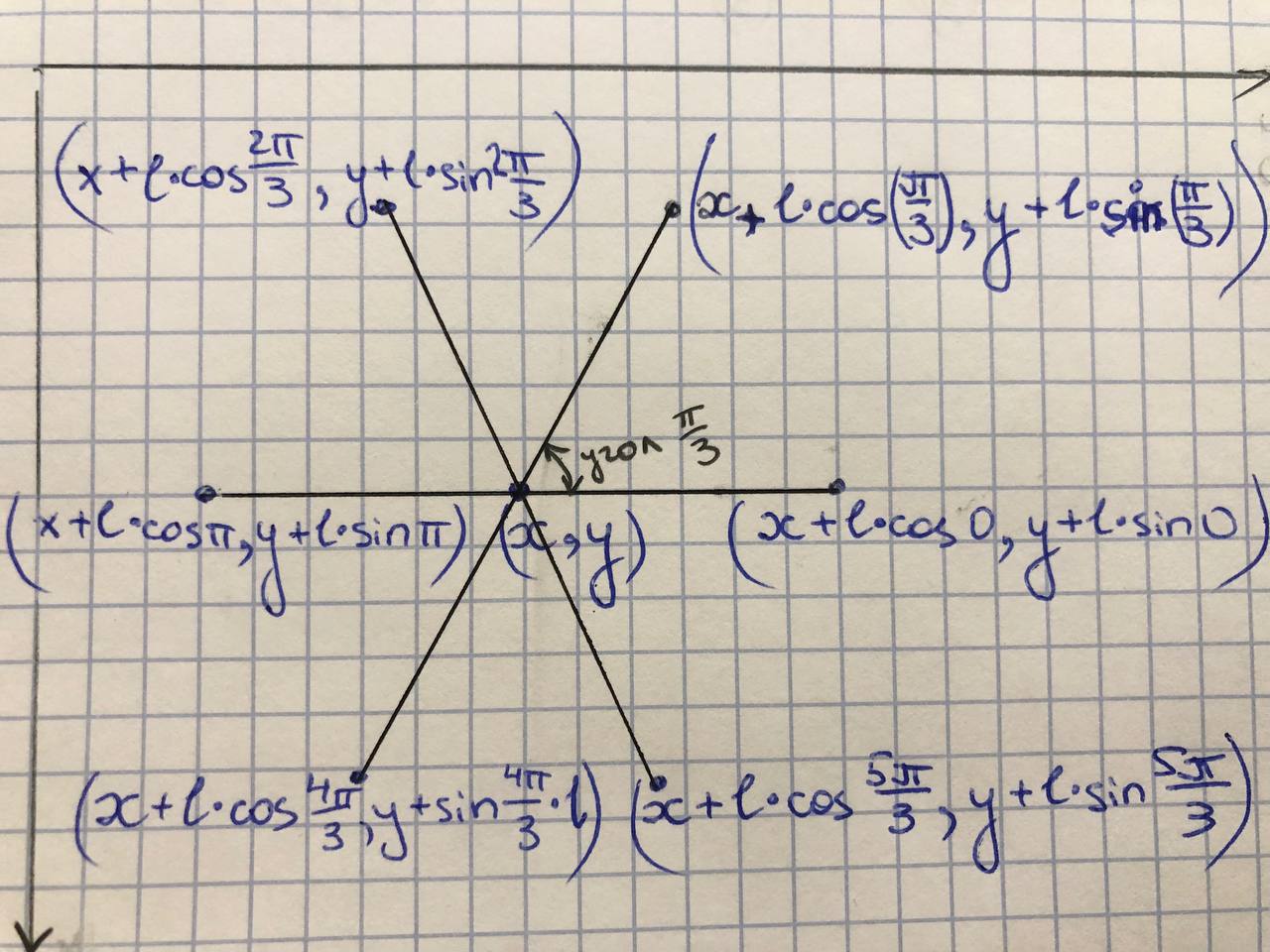
Разработайте расширения для построения указанных ниже изображений. Исходные данные для расширений определите самостоятельно таким образом, чтобы параметров для задания фрактального изображения было как можно больше (координаты, размеры, углы, глубина и т.п.).

# Эскиз фрактала

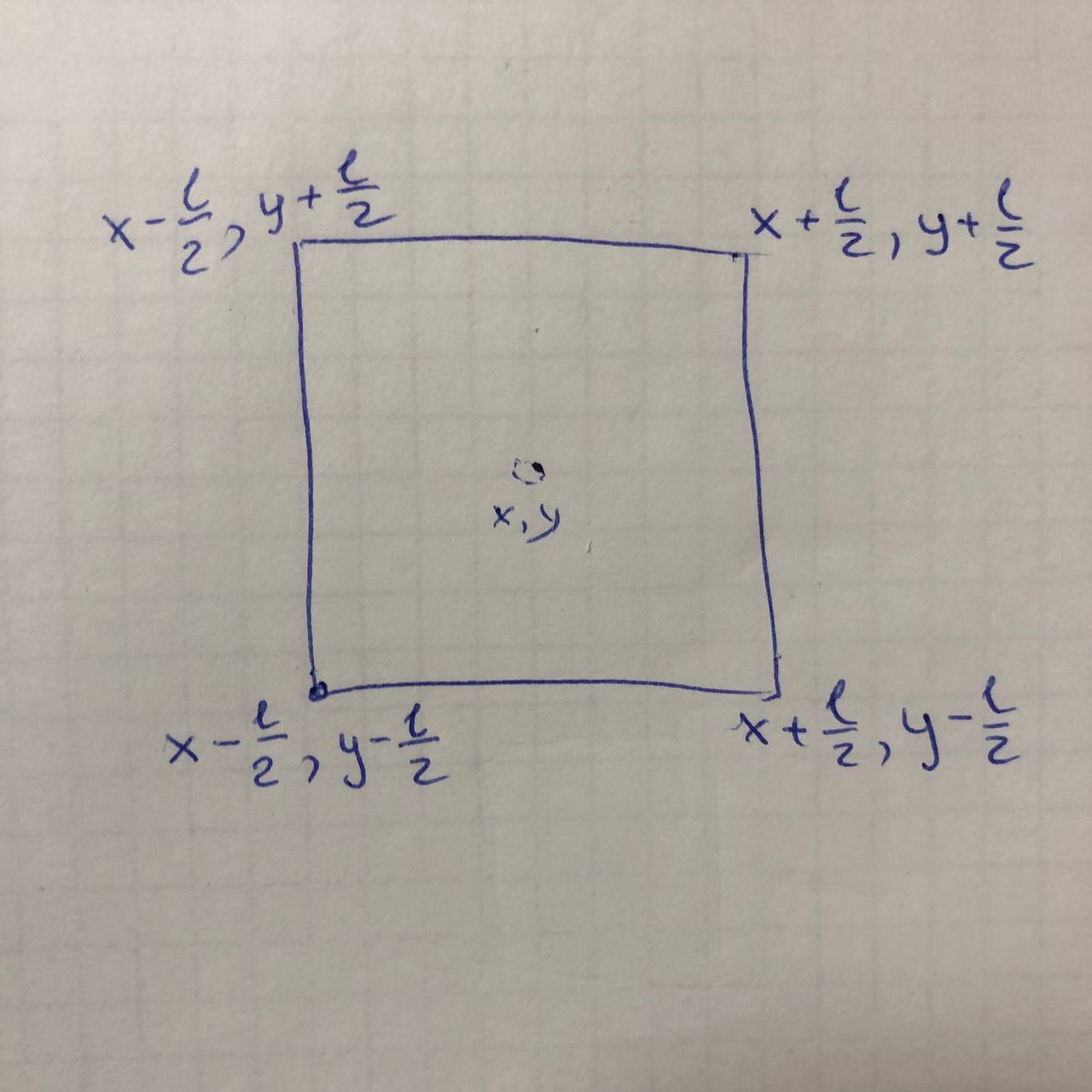
Ветка



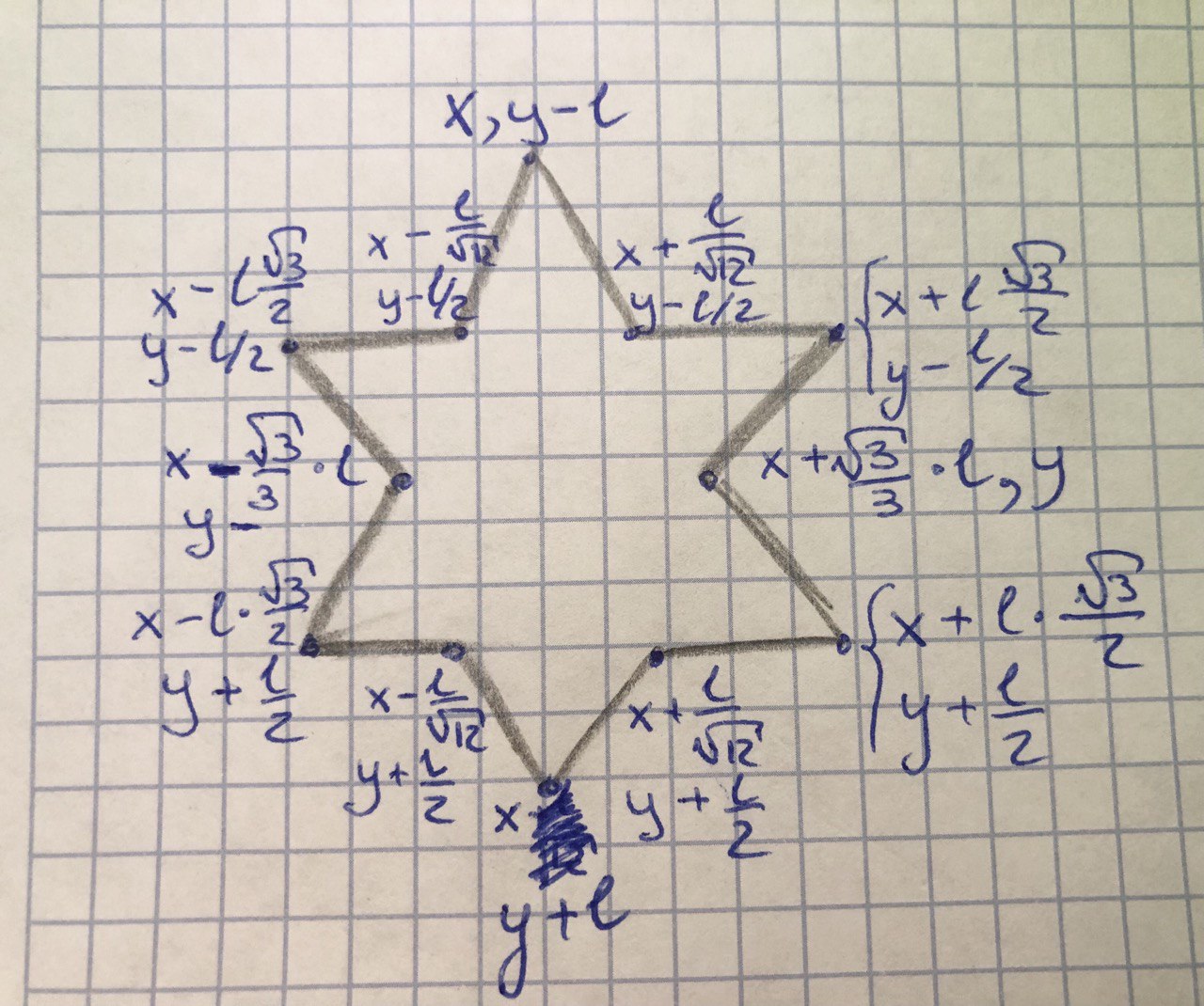
Снежинка



Множество Кантора

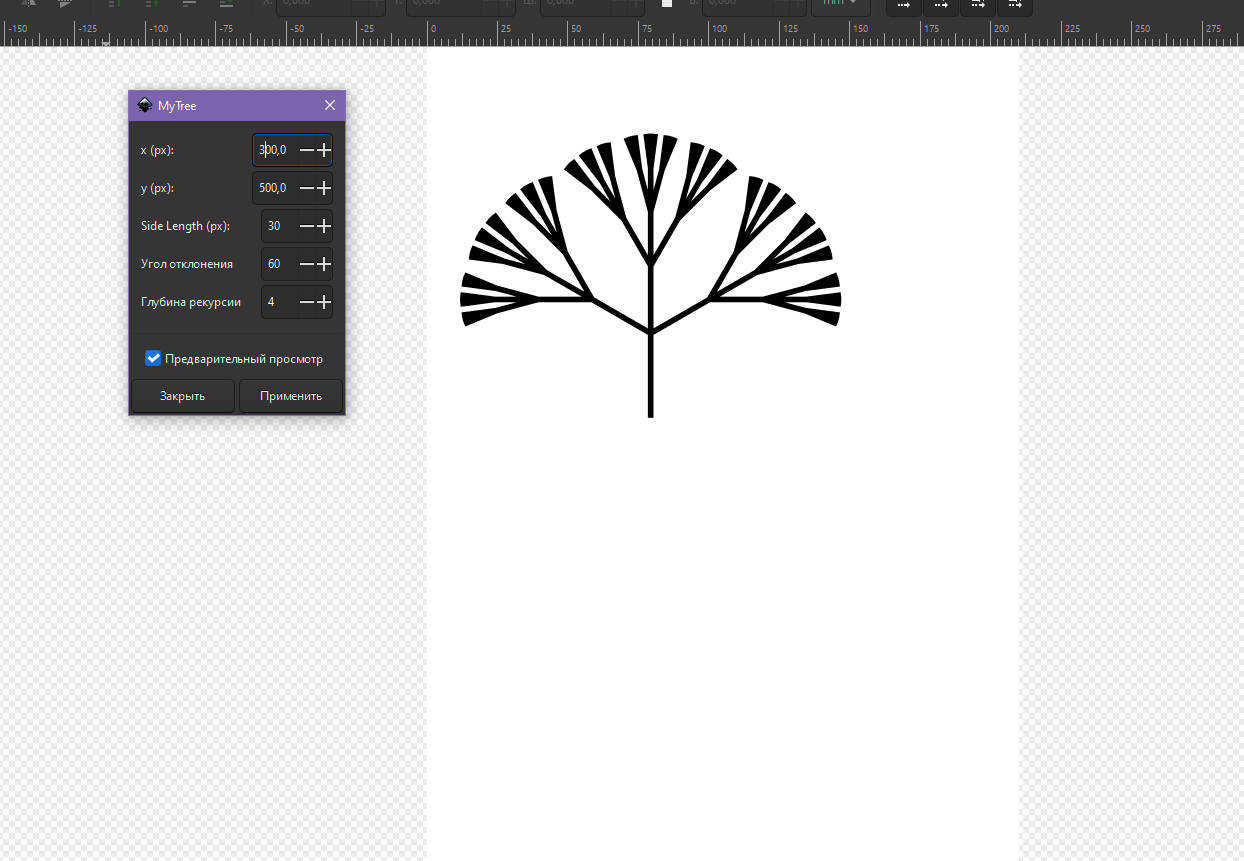


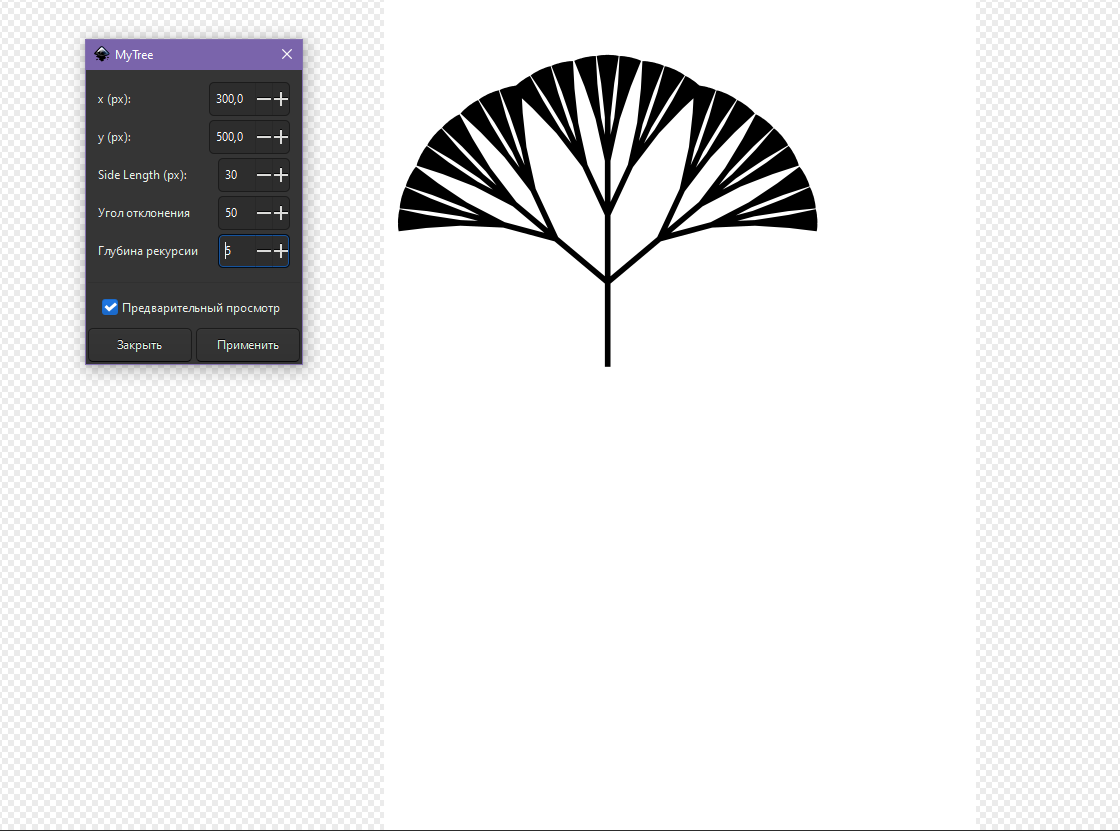
Снежинка Коха



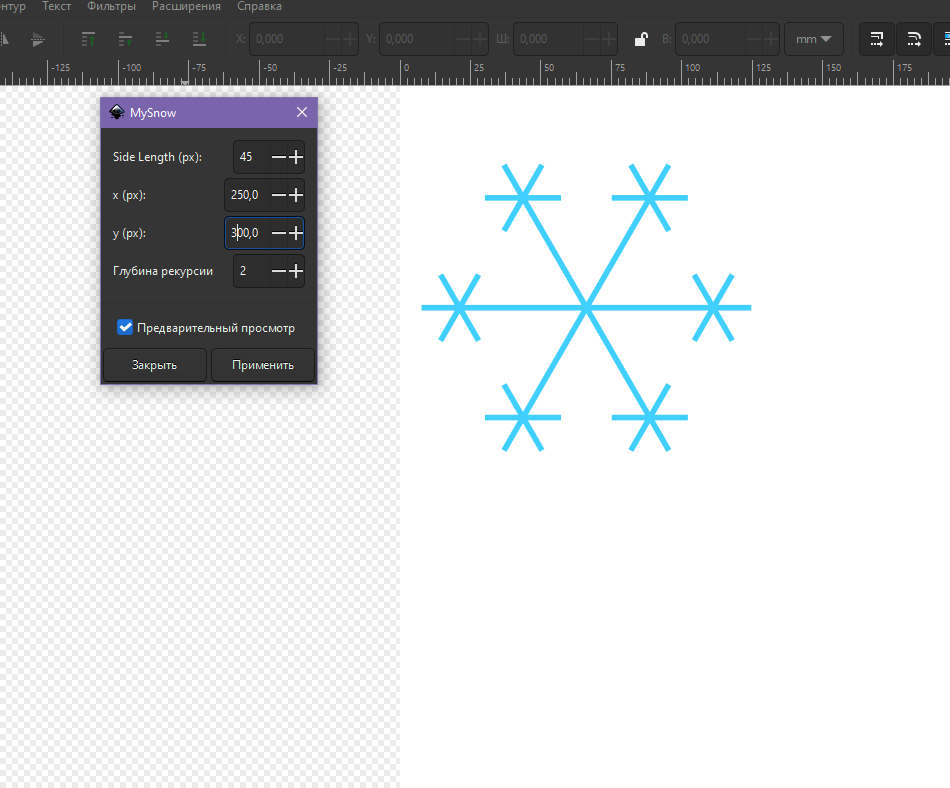
# Описание расширения

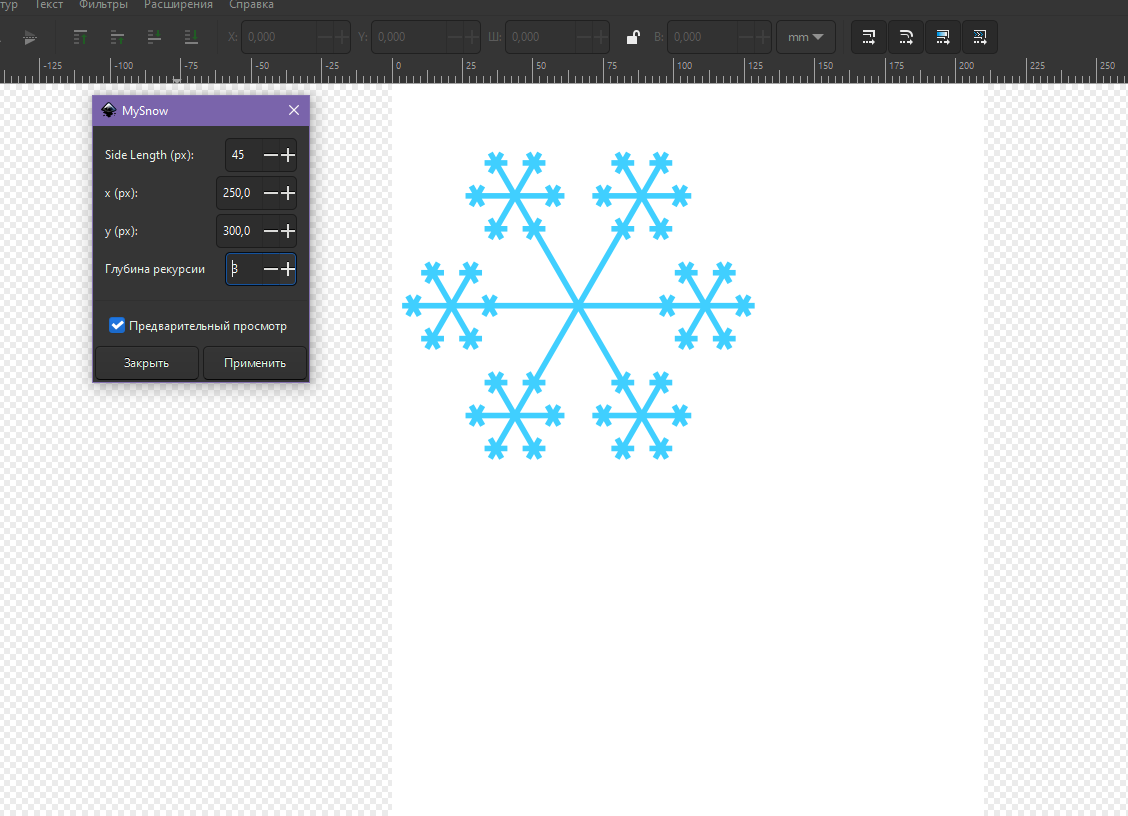
Ветка



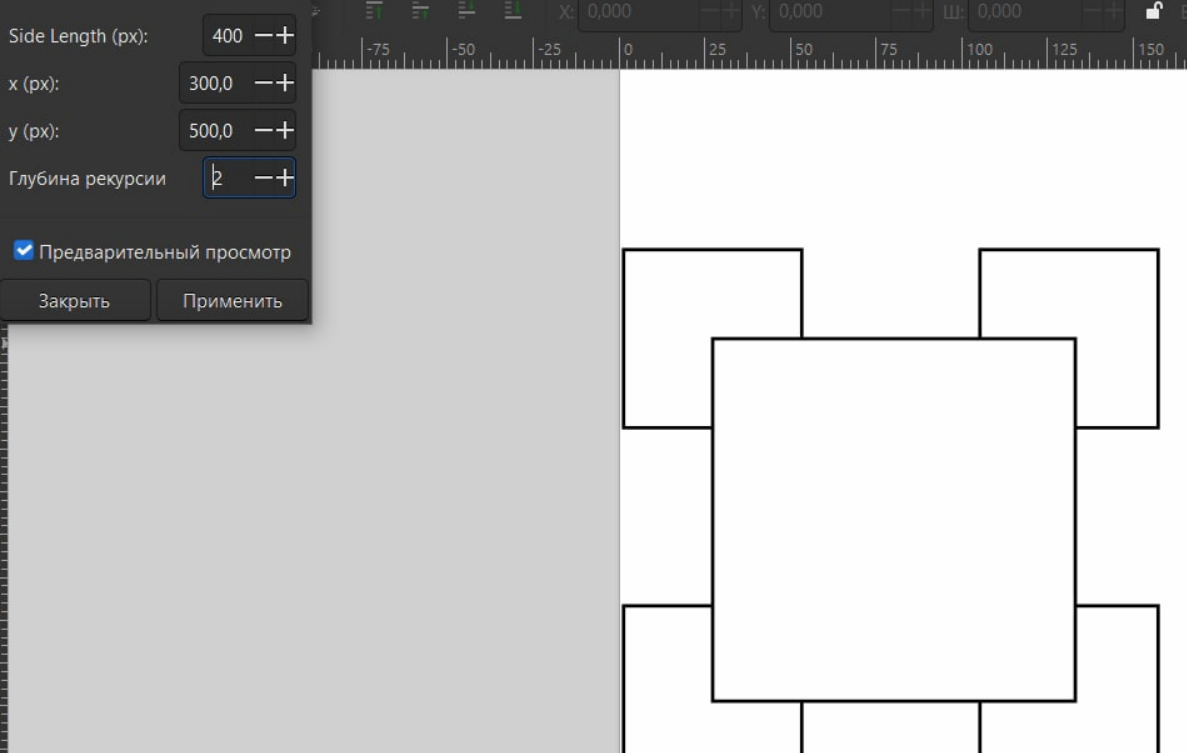


Снежинка

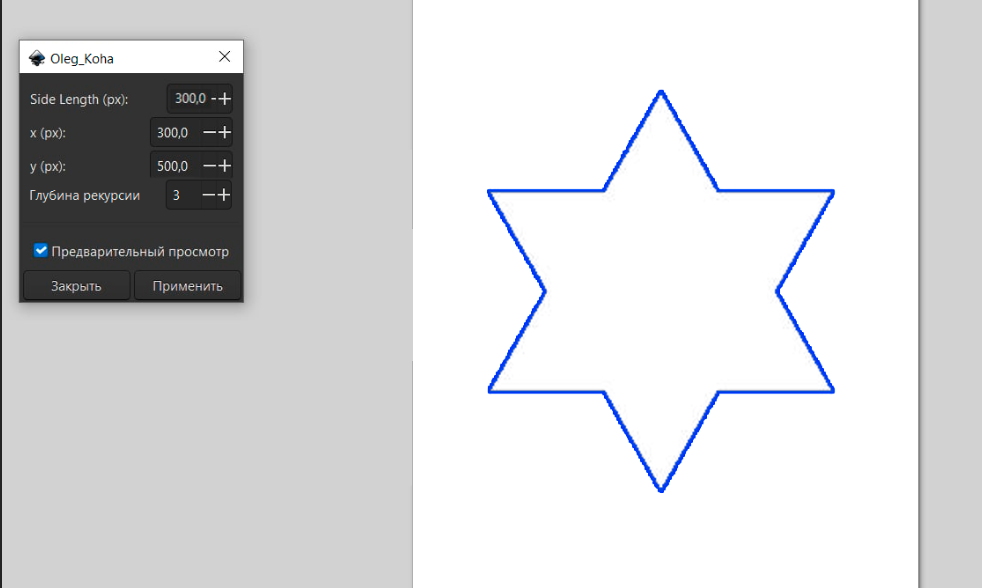




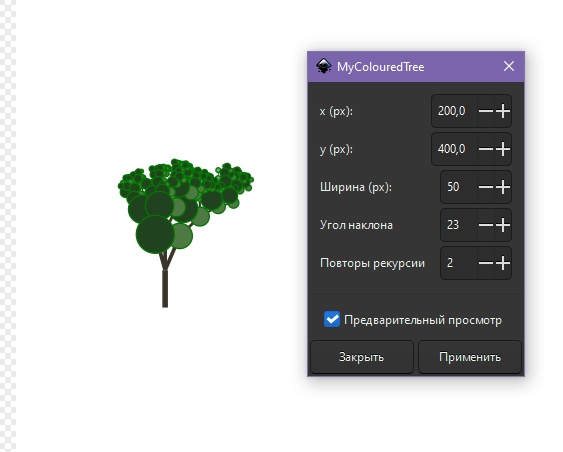
Множество Кантора

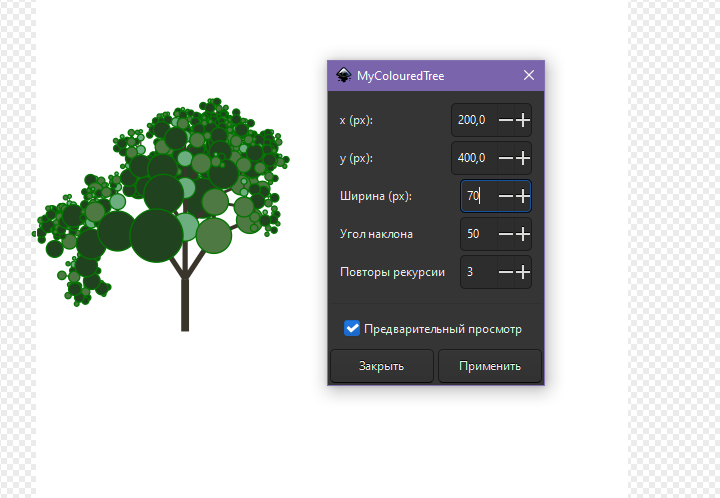


Снежинка Коха



Фрактальное дерево





# Текст расширения (.inx и .py)

Ветка

#!/usr/bin/env python

# coding=utf-8

import sys

import inkex

from math import sin, cos, pi, radians

from lxml import etree

def MyTree\_string(x,y,l,angle,n,angle2):

if n==0:

return ""

angle2=angle2\*0.5

s=f"M {x} {y} L {x+l\*cos(angle)} {y-l\*sin(angle)} "

s=s+MyTree\_string(x+l\*cos(angle), y-l\*sin(angle), l\*0.8, angle+angle2,n-1,angle2)

s=s+f"M {x} {y} L {x+l\*cos(angle)} {y-l\*sin(angle)} "

s=s+MyTree\_string(x+l\*cos(angle), y-l\*sin(angle), l\*0.8, angle-angle2,n-1,angle2)

s=s+f"M {x} {y} L {x+l\*cos(angle)} {y-l\*sin(angle)} "

s=s+MyTree\_string(x+l\*cos(angle), y-l\*sin(angle), l\*0.8, angle,n-1,angle2)

return s

def draw\_SVG\_MyTree(x,y,l,angle,n,angle2):

style = {'stroke': 'black', 'stroke-width': '2', 'fill': 'none'}

elem = inkex.PathElement()

elem.update(\*\*{

'style': style,

'inkscape:label': 'MyTree',

'd': MyTree\_string(x,y,l,angle,n,angle2)

})

return elem

class MyTree(inkex.EffectExtension):

def add\_arguments(self, pars):

pars.add\_argument("--c\_x", type=float, default=10, help="x (px)")

pars.add\_argument("--c\_y", type=float, default=40, help="y (px)")

pars.add\_argument("--s\_l", type=int, default=100.0, help="Side Length")

pars.add\_argument("--angle", type=int, default=0, help="Угол отклонения")

pars.add\_argument("--n", type=int, default=0, help="Глубина рекурсии")

def effect(self):

cur = self.svg.get\_current\_layer()

x = self.svg.unittouu(str(self.options.c\_x) + 'px')

y = self.svg.unittouu(str(self.options.c\_y) + 'px')

l = self.options.s\_l

angle = self.options.angle

n = self.options.n

cur.add(draw\_SVG\_MyTree(x,y,l,pi/2,n+1,radians(angle\*2)))

if \_\_name\_\_ == '\_\_main\_\_':

MyTree().run()

<?xml version="1.0" encoding="UTF-8"?>

<inkscape-extension xmlns="http://www.inkscape.org/namespace/inkscape/extension">

<name>MyTree</name>

<id>MyTree</id>

<param name="c\_x" type="float" min="0.01" max="10000" \_gui-text="x (px):">10.0</param>

<param name="c\_y" type="float" min="0.01" max="10000" \_gui-text="y (px):">40.0</param>

<param name="s\_l" type="int" min="1" max="1000" gui-text="Side Length (px):">150.0</param>

<param name="angle" type="int" min="0" max="360" gui-text="Угол отклонения">0</param>

<param name="n" type="int" min="1" max="1000" gui-text="Глубина рекурсии">0</param>

<effect>

<object-type>all</object-type>

<effects-menu>

<submenu name="Render"/>

</effects-menu>

</effect>

<script>

<command location="inx" interpreter="python">MyTree.py</command>

</script>

</inkscape-extension>

Снежинка

#!/usr/bin/env python

# coding=utf-8

import sys

import inkex

from math import sin, cos, pi

from lxml import etree

def MySnow\_string(x,y,l,n):

if n==0:

return ""

s=f"M {x} {y} L {x+l\*cos(pi/3)} {y+l\*sin(pi/3)} "

s=s+MySnow\_string(x+l\*cos(pi/3), y+l\*sin(pi/3), l\*0.3,n-1)

s=s+f"M {x} {y} L {x+l\*cos(2\*pi/3)} {y+l\*sin(2\*pi/3)} "

s=s+MySnow\_string(x+l\*cos(2\*pi/3), y+l\*sin(2\*pi/3), l\*0.3,n-1)

s=s+f"M {x} {y} L {x+l\*cos(4\*pi/3)} {y+l\*sin(4\*pi/3)} "

s=s+MySnow\_string(x+l\*cos(4\*pi/3), y+l\*sin(4\*pi/3), l\*0.3,n-1)

s=s+f"M {x} {y} L {x+l\*cos(5\*pi/3)} {y+l\*sin(5\*pi/3)} "

s=s+MySnow\_string(x+l\*cos(5\*pi/3), y+l\*sin(5\*pi/3), l\*0.3,n-1)

s=s+f"M {x} {y} L {x+l\*cos(0)} {y+l\*sin(0)} "

s=s+MySnow\_string(x+l\*cos(0), y+l\*sin(0), l\*0.3,n-1)

s=s+f"M {x} {y} L {x+l\*cos(pi)} {y+l\*sin(pi)} "

s=s+MySnow\_string(x+l\*cos(pi), y+l\*sin(pi), l\*0.3,n-1)

return s

def draw\_SVG\_MySnow(x,y,l,n):

style = {'stroke': '#40cfff', 'stroke-width': '2', 'fill': 'none'}

elem = inkex.PathElement()

elem.update(\*\*{

'style': style,

'inkscape:label': 'MySnow',

'd': MySnow\_string(x,y,l,n)

})

return elem

class MySnow(inkex.EffectExtension):

def add\_arguments(self, pars):

pars.add\_argument("--s\_l", type=int, default=100.0, help="Side Length")

pars.add\_argument("--c\_x", type=float, default=10, help="x (px)")

pars.add\_argument("--c\_y", type=float, default=40, help="y (px)")

pars.add\_argument("--n", type=int, default=0, help="Глубина рекурсии")

def effect(self):

cur = self.svg.get\_current\_layer()

l = self.options.s\_l

x = self.svg.unittouu(str(self.options.c\_x) + 'px')

y = self.svg.unittouu(str(self.options.c\_y) + 'px')

n = self.options.n

cur.add(draw\_SVG\_MySnow(x,y,l,n))

if \_\_name\_\_ == '\_\_main\_\_':

MySnow().run()

<?xml version="1.0" encoding="UTF-8"?>

<inkscape-extension xmlns="http://www.inkscape.org/namespace/inkscape/extension">

<name>MySnow</name>

<id>MySnow</id>

<param name="s\_l" type="int" min="1" max="1000" gui-text="Side Length (px):">150.0</param>

<param name="c\_x" type="float" min="0.01" max="10000" \_gui-text="x (px):">10.0</param>

<param name="c\_y" type="float" min="0.01" max="10000" \_gui-text="y (px):">40.0</param>

<param name="n" type="int" min="1" max="1000" gui-text="Глубина рекурсии">0</param>

<effect>

<object-type>all</object-type>

<effects-menu>

<submenu name="Render"/>

</effects-menu>

</effect>

<script>

<command location="inx" interpreter="python">MySnow.py</command>

</script>

</inkscape-extension>

Множество Кантора

#!/usr/bin/env python

# coding=utf-8

import sys

import inkex

import math

from lxml import etree

def draw\_MyKantor(l, x, y, n):

sub\_grp = inkex.Group()

if n>1:

sub\_grp.add(draw\_MyKantor(l//2, x-l//2, y-l//2, n-1))

sub\_grp.add(draw\_MyKantor(l//2, x+l//2, y-l//2, n-1))

sub\_grp.add(draw\_MyKantor(l//2, x-l//2, y+l//2, n-1))

sub\_grp.add(draw\_MyKantor(l//2, x+l//2, y+l//2, n-1))

style = {'stroke':'black', 'stroke-width': '1', 'fill' : 'white' }

elem = inkex.Rectangle()

elem.update(\*\*{

'style': style,

'inkscape:label': 'MyKantor',

'x': str(x - l//2),

'y': str(y - l//2),

'width': str(l),

'height': str(l) })

sub\_grp.add(elem)

return sub\_grp

class MyKantor(inkex.EffectExtension):

def add\_arguments(self, pars):

pars.add\_argument("--s\_l", type=int, default=100.0, help="Side Length")

pars.add\_argument("--c\_x", type=float, default=10.0, help="x(px)")

pars.add\_argument("--c\_y", type=float, default=40.0, help="y(px)")

pars.add\_argument("--n", type=int, default=4, help="Глубина рекурсии")

def effect(self):

cur = self.svg.get\_current\_layer()

a = self.svg.unittouu(str(self.options.s\_l) + 'px')

x = self.svg.unittouu(str(self.options.c\_x) + 'px')

y = self.svg.unittouu(str(self.options.c\_y) + 'px')

n = self.options.n

cur.add(draw\_MyKantor(a,x,y,n))

if \_\_name\_\_ == '\_\_main\_\_':

MyKantor().run()

<?xml version="1.0" encoding="UTF-8"?>

<inkscape-extension xmlns="http://www.inkscape.org/namespace/inkscape/extension">

<\_name>MyKantor</\_name>

<id>MyKantor</id>

<param name="s\_l" type="int" min="1" max="1000" gui-text="Side Length (px):">124.0</param>

<param name="c\_x" type="float" min="0.01" max="10000" \_gui-text="x (px):">10.0</param>

<param name="c\_y" type="float" min="0.01" max="10000" \_gui-text="y (px):">40.0</param>

<param name="n" type="int" min="1" max="1000" gui-text="Глубина рекурсии">0</param>

<effect>

<object-type>all</object-type>

<effects-menu>

<submenu \_name="Render"/>

</effects-menu>

</effect>

<script>

<command reldir="extensions" interpreter="python">MyKantor.py</command>

</script>

</inkscape-extension>

Снежинка Коха

#!/usr/bin/env python

# coding=utf-8

import sys

import inkex

from lxml import etree

from math import sqrt

def MyKoha(x,y,l):

s='M'+str(x)+','+str(y-l)+'L'+str(x+l/sqrt(12))+','+str(y-l/2)+'L'+str(x+l\*sqrt(3)/2)+','+str(y-l/2)+'L'+str(x+l\*sqrt(3)/3)+','+str(y)+'L'+str(x+l\*sqrt(3)/2)+','+str(y+l/2)+'L'+str(x+l/sqrt(12))+','+str(y+l/2)+'L'+str(x)+','+str(y+l)+'L'+str(x-l/sqrt(12))+','+str(y+l/2)+'L'+str(x-l\*sqrt(3)/2)+','+str(y+l/2)+'L'+str(x-l\*sqrt(3)/3)+','+str(y)+'L'+str(x-l\*sqrt(3)/2)+','+str(y-l/2)+'L'+str(x-l/sqrt(12))+','+str(y-l/2)+'z'

if (l>10):

s=s+MyKoha(x,y-l\*2/3,l/3)

s=s+MyKoha(x+l\*sqrt(3)/3,y-l/3,l/3)

s=s+MyKoha(x+l\*sqrt(3)/3,y+l/3,l/3)

s=s+MyKoha(x,y+l\*2/3,l/3)

s=s+MyKoha(x-l\*sqrt(3)/3,y+l/3,l/3)

s=s+MyKoha(x-l\*sqrt(3)/3,y-l/3,l/3)

return s

def draw\_SVG\_MyKoha(x,y,l,cur):

style = {'stroke': 'black', 'stroke-width': '0.5', 'fill': 'black'}

elem = cur.add(inkex.PathElement())

elem.update(\*\*{

'style': style,

'inkscape:label': 'MyKOHA',

'd': MyKoha(x,y,l)})

return elem

class MyKOHA(inkex.EffectExtension):

def add\_arguments(self, pars):

pars.add\_argument("--s\_l", type=int, default=100, help="Side Length")

pars.add\_argument("--c\_x", type=float, default=10, help="x (px)")

pars.add\_argument("--c\_y", type=float, default=40, help="y (px)")

def effect(self):

cur = self.svg.get\_current\_layer()

x = self.svg.unittouu(str(self.options.c\_x) + 'px')

y = self.svg.unittouu(str(self.options.c\_y) + 'px')

l = self.svg.unittouu(str(self.options.s\_l) + 'px')

draw\_SVG\_MyKoha(x,y,l,cur)

if \_\_name\_\_ == '\_\_main\_\_':

MyKOHA().run()

<?xml version="1.0" encoding="UTF-8"?>

<inkscape-extension xmlns="http://www.inkscape.org/namespace/inkscape/extension">

<name>MyKoha</name>

<id>MyKoha</id>

<param name="s\_l" type="int" min="1" max="1000" gui-text="Side Length (px):">150.0</param>

<param name="c\_x" type="float" min="0.01" max="10000" \_gui-text="x (px):">10.0</param>

<param name="c\_y" type="float" min="0.01" max="10000" \_gui-text="y (px):">40.0</param>

<effect>

<object-type>all</object-type>

<effects-menu>

<submenu name="Render"/>

</effects-menu>

</effect>

<script>

<command location="inx" interpreter="python">MyKoha.py</command>

</script>

</inkscape-extension>

Фрактальное дерево

#!/usr/bin/env python

# coding=utf-8

import sys

import inkex

from math import sin, cos, pi, radians

from lxml import etree

def MyTree\_string(x,y,l,angle,n,angle2):

if n==0:

return ""

angle2=angle2\*0.5

s=f"M {x} {y} L {x+l\*cos(angle)} {y-l\*sin(angle)} "

s=s+MyTree\_string(x+l\*cos(angle), y-l\*sin(angle), l\*0.8, angle+angle2,n-1,angle2)

s=s+f"M {x} {y} L {x+l\*cos(angle)} {y-l\*sin(angle)} "

s=s+MyTree\_string(x+l\*cos(angle), y-l\*sin(angle), l\*0.8, angle-angle2,n-1,angle2)

s=s+f"M {x} {y} L {x+l\*cos(angle)} {y-l\*sin(angle)} "

s=s+MyTree\_string(x+l\*cos(angle), y-l\*sin(angle), l\*0.8, angle,n-1,angle2)

return s

def draw\_SVG\_MyTree(x,y,l,angle,n,angle2):

style = {'stroke': 'purple', 'stroke-width': '2', 'fill': 'none'}

elem = inkex.PathElement()

elem.update(\*\*{

'style': style,

'inkscape:label': 'MyTree',

'd': MyTree\_string(x,y,l,angle,n,angle2)

})

return elem

class MyTree(inkex.EffectExtension):

def add\_arguments(self, pars):

pars.add\_argument("--c\_x", type=float, default=10, help="x (px)")

pars.add\_argument("--c\_y", type=float, default=40, help="y (px)")

pars.add\_argument("--s\_l", type=int, default=50, help="Side Length")

pars.add\_argument("--angle", type=int, default=45, help="Угол отклонения")

pars.add\_argument("--n", type=int, default=1, help="Глубина рекурсии")

def effect(self):

cur = self.svg.get\_current\_layer()

x = self.svg.unittouu(str(self.options.c\_x) + 'px')

y = self.svg.unittouu(str(self.options.c\_y) + 'px')

l = self.options.s\_l

angle = self.options.angle

n = self.options.n

cur.add(draw\_SVG\_MyTree(x,y,l,pi/2,n+1,radians(angle\*2)))

if \_\_name\_\_ == '\_\_main\_\_':

MyTree().run()

<?xml version="1.0" encoding="UTF-8"?>

<inkscape-extension xmlns="http://www.inkscape.org/namespace/inkscape/extension">

<name>MyTreeBranch</name>

<id>MyTreeBranch</id>

<param name="c\_x" type="float" min="0.01" max="10000" \_gui-text="x (px):">10.0</param>

<param name="c\_y" type="float" min="0.01" max="10000" \_gui-text="y (px):">40.0</param>

<param name="s\_l" type="int" min="1" max="1000" gui-text="Side Length (px):">50.0</param>

<param name="angle" type="int" min="0" max="360" gui-text="Угол отклонения">45</param>

<param name="n" type="int" min="1" max="1000" gui-text="Глубина рекурсии">1</param>

<effect>

<object-type>all</object-type>

<effects-menu>

<submenu name="Render"/>

</effects-menu>

</effect>

<script>

<command location="inx" interpreter="python">MyTreeBranch.py</command>

</script>

</inkscape-extension>