Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего профильного образования «Национальный исследовательский университет ИТМО»

Мегафакультет компьютерных технологий и управления

Факультет программной инженерии и компьютерной техники

Образовательная программа: «Компьютерные технологии в дизайне»

«Использование встроенных средств автоматизации в приложении»

Отчёт

Студент: Кешишев Максим Юрьевич

Группа: P3166

Преподаватель: Климов Игорь Викторович

Санкт-Петербург, 2023

Оглавление

[Текст задания 3](#_Toc153490120)

[Эскиз ветки 3](#_Toc153490121)

[Текст расширения для ветки 3](#_Toc153490122)

[Файл inx 4](#_Toc153490123)

[Файл py 4](#_Toc153490124)

[Снимки результатов ветки 6](#_Toc153490125)

[Эскиз снежинки 7](#_Toc153490126)

[Текст расширения для снежинки 7](#_Toc153490127)

[Файл inx 7](#_Toc153490128)

[Файл py 7](#_Toc153490129)

[Снимки результатов снежинки 9](#_Toc153490130)

[Эскиз множества кантора 9](#_Toc153490131)

[Текст расширения для множества кантора 10](#_Toc153490132)

[Файл inx 10](#_Toc153490133)

[Файл py 10](#_Toc153490134)

[Снимки результатов множества кантора 12](#_Toc153490135)

[Эскиз снежинки коха 12](#_Toc153490136)

[Текст расширения для снежинки коха 12](#_Toc153490137)

[Файл inx 12](#_Toc153490138)

[Файл py 13](#_Toc153490139)

[Снимки результатов снежинки коха 15](#_Toc153490140)

[Текст расширения для фрактального дерева 15](#_Toc153490141)

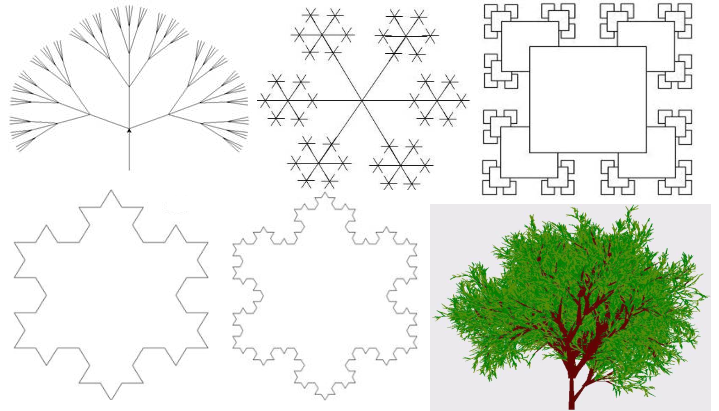
[Файл inx 15](#_Toc153490142)

[Файл py 16](#_Toc153490143)

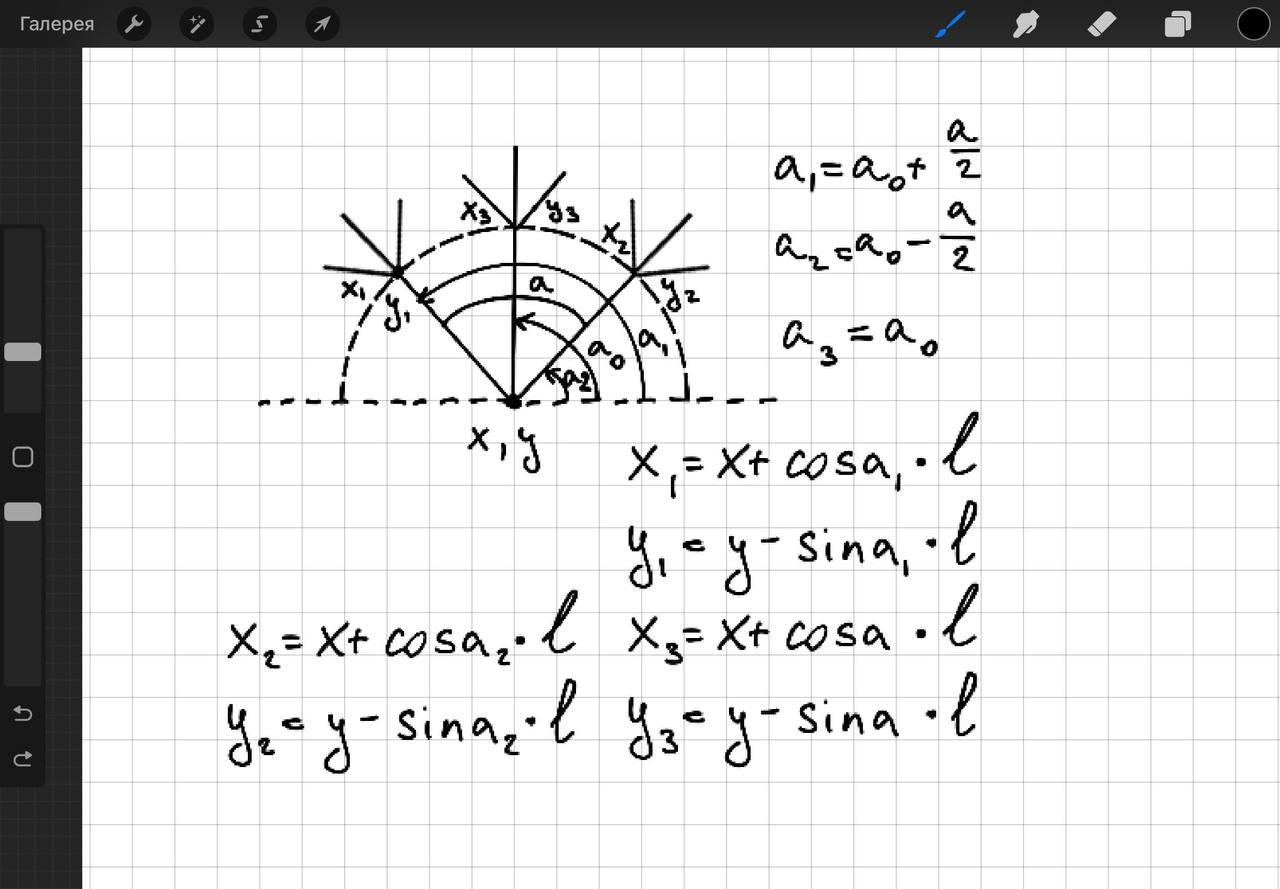
[Снимки результатов фрактального дерева 18](#_Toc153490144)

Текст задания

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 ветки



Текст расширения для ветки

Файл inx

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

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

<name>BETKA</name>

<id>BETKA</id>

<param name="l" type="float" min="10" max="1000" gui-text="Длина ветки">100.0</param>

<param name="x" type="float" min="0" max="1000" gui-text="Х центра">100.0</param>

<param name="y" type="float" min="0" max="1000" gui-text="Y центра">100.0</param>

<param name="n" type="int" min="1" max="10000" gui-text="Глубина рекурсии">3</param>

<effect>

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

<effects-menu>

<submenu name="Render"/>

</effects-menu>

</effect>

<script>

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

</script>

</inkscape-extension>

Файл py

#!/usr/bin/env python3

# coding=utf-8

import sys

import inkex

from lxml import etree

import random

import math

def generation\_tree(a0,a,x,y,l,s,n):

if n>0:

a1=a0+a/2

a2=a0-a/2

a3=a0

x1=x+math.cos(a1)\*l

y1=y-math.sin(a1)\*l

x2=x+math.cos(a2)\*l

y2=y-math.sin(a2)\*l

x3=x+math.cos(a3)\*l

y3=y-math.sin(a3)\*l

s=s+" M " + str(x1) + "," + str(y1)+ " L " + str(x) + "," + str(y) + " L " + str(x2) + "," + str(y2)

s=s+" M " + str(x3) + "," + str(y3)+ " L " + str(x) + "," + str(y)

s=s+generation\_tree(a1,a-a/4,x1,y1,l//2,'',n-1)

s=s+generation\_tree(a2,a-a/4,x2,y2,l//2,'', n-1)

s=s+generation\_tree(a3,a-a/4,x3,y3,l//2,'', n-1)

return s

def draw\_BETKA(x,y,l,a0,a,n,grp):

style = {"stroke": "#000000", "stroke-width": '1px', "fill": "none"}

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

elem.update(

\*\*{

"style": style,

"inkscape:label": 'BETKA',

"d": generation\_tree(3.14/2,3.14/2,x,y,l,'',n)

}

)

elem1 = grp.add(inkex.PathElement())

elem1.update(

\*\*{

"style": style,

"inkscape:label": 'BETKA',

"d": " M " + str(x) + "," + str(y)+ " L " + str(x) + "," + str(y+l)

}

)

return elem

return elem1

class BETKA(inkex.EffectExtension):

def add\_arguments(self, pars):

pars.add\_argument("--n", type=int, default=3.0, help="Глубина рекурсии")

pars.add\_argument("--l", type=float, default=100.0, help="Длина ветки")

pars.add\_argument("--x", type=float, default=100.0, help="Х центра")

pars.add\_argument("--y", type=float, default=100.0, help="Y центра")

def effect(self):

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

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

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

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

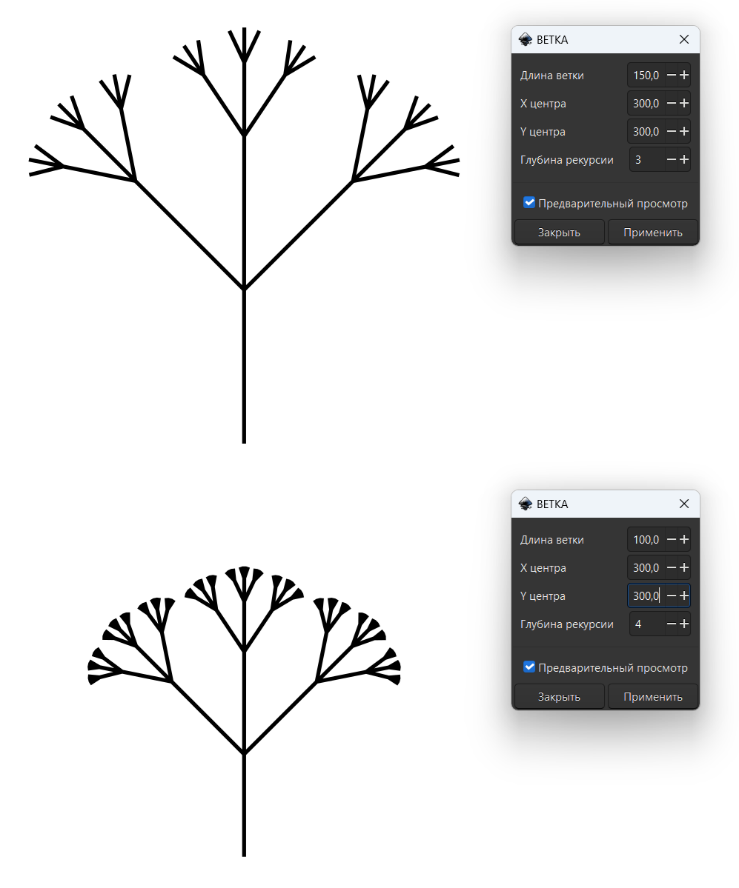
n = self.options.n

draw\_BETKA(x,y,l,3.14/2,3.14/2,n,grp)

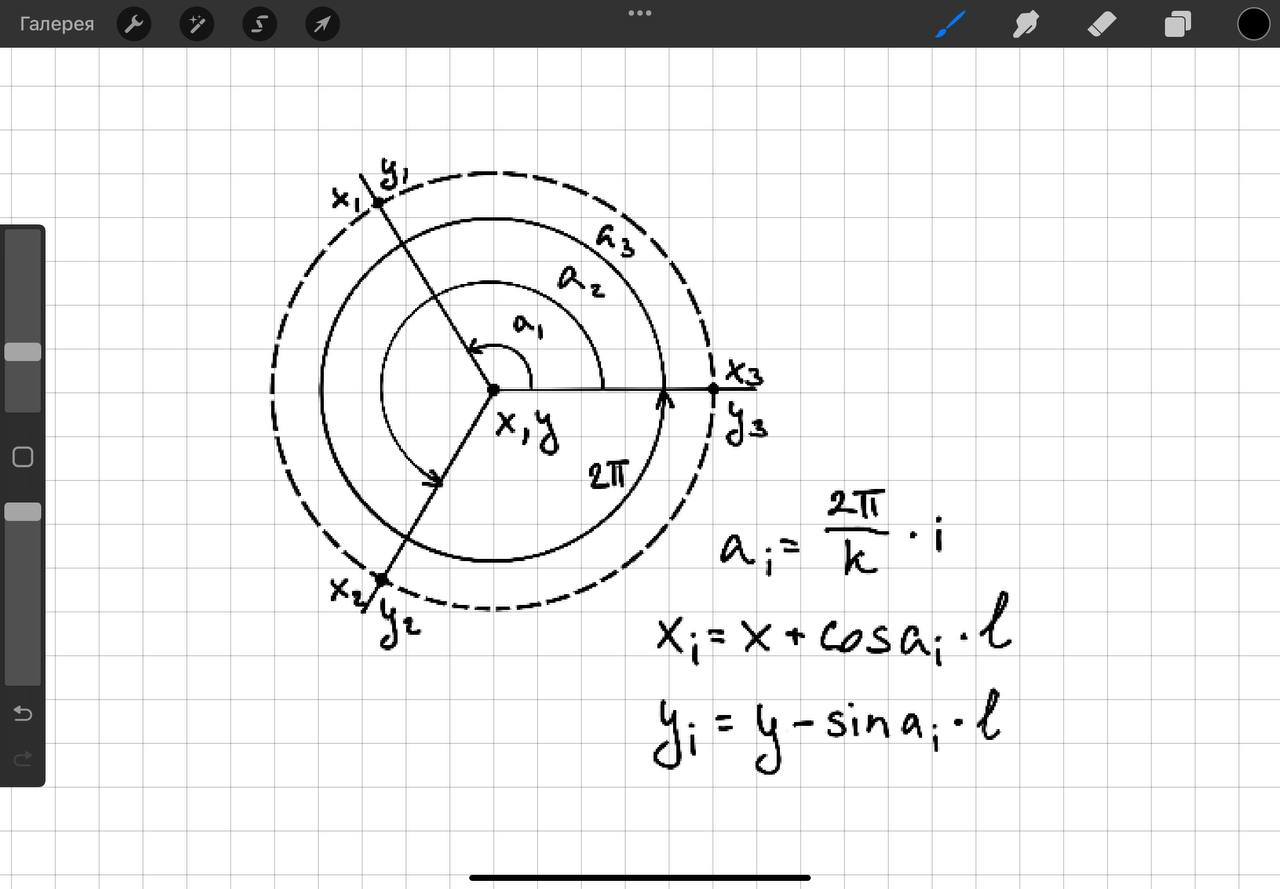
if \_\_name\_\_ == "\_\_main\_\_":

BETKA().run()

Снимки результатов ветки



Эскиз снежинки



Текст расширения для снежинки

Файл inx

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

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

<name>CHEIIINHKA</name>

<id>CHEIIINHKA</id>

<param name="l" type="float" min="10" max="1000" gui-text="Длина ветки">100.0</param>

<param name="x" type="float" min="0" max="1000" gui-text="X центра">100.0</param>

<param name="y" type="float" min="0" max="1000" gui-text="Y центра">100.0</param>

<param name="n" type="int" min="1" max="10000" gui-text="Глубина рекурсии">3</param>

<param name="k" type="int" min="1" max="10000" gui-text="Количество веток">6</param>

<effect>

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

<effects-menu>

<submenu name="Render"/>

</effects-menu>

</effect>

<script>

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

</script>

</inkscape-extension>

Файл py

#!/usr/bin/env python3

# coding=utf-8

import sys

import inkex

from lxml import etree

import random

import math

def generation\_CHEIIINHKA(a,x,y,l,s,n,k):

if n>0:

for i in range(k):

an=6.28/k\*i

xn=x+math.cos(an)\*l

yn=y-math.sin(an)\*l

s=s+" M " + str(xn) + "," + str(yn)+ " L " + str(x) + "," + str(y)

s=s+generation\_CHEIIINHKA(an,xn,yn,l//3,'',n-1,k)

return s

def draw\_CHEIIINHKA(x,y,l,n,grp,k):

style = {"stroke": "#000000", "stroke-width": '1px', "fill": "none"}

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

elem.update(

\*\*{

"style": style,

"inkscape:label": 'snow',

"d": generation\_CHEIIINHKA(6.28,x,y,l,'',n,k)

}

)

return elem

class CHEIIINHKA(inkex.EffectExtension):

def add\_arguments(self, pars):

pars.add\_argument("--n", type=int, default=3.0, help="Глубина рекурсии")

pars.add\_argument("--l", type=float, default=100.0, help="Длина ветки")

pars.add\_argument("--x", type=float, default=100.0, help="X центра")

pars.add\_argument("--y", type=float, default=100.0, help="Y центра")

pars.add\_argument("--k", type=int, default=6.0, help="Количество веток")

def effect(self):

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

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

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

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

n = self.options.n

k = self.options.k

draw\_CHEIIINHKA(x,y,l,n,grp,k)

if \_\_name\_\_ == "\_\_main\_\_":

CHEIIINHKA().run()

Снимки результатов снежинки

Изображение выглядит как снимок экрана, диаграмма, линия, дизайн

Автоматически созданное описание

Эскиз множества кантора

Изображение выглядит как текст, диаграмма, снимок экрана, График

Автоматически созданное описание

Текст расширения для множества кантора

Файл inx

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

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

<name>KAHTOP</name>

<id>KAHTOP</id>

<param name="l" type="float" min="10" max="10000" gui-text="Сторона квадрата">100.0</param>

<param name="x" type="float" min="10" max="10000" gui-text="X центра">100.0</param>

<param name="y" type="float" min="10" max="10000" gui-text="Y центра">100.0</param>

<param name="n" type="int" min="1" max="10000" gui-text="Глубина рекурсии">3</param>

<effect>

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

<effects-menu>

<submenu name="Render"/>

</effects-menu>

</effect>

<script>

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

</script>

</inkscape-extension>

Файл py

#!/usr/bin/env python3

# coding=utf-8

import sys

import inkex

import random

def generate\_KAHTOP(x, y, l, parent, n):

if n>0:

x1=x-l

y1=y-l

x2=x+l

y2=y-l

x3=x+l

y3=y+l

x4=x-l

y4=y+l

group = parent.add(inkex.Group())

if l > 2:

generate\_KAHTOP(x1, y1, l//2, group, n-1)

generate\_KAHTOP(x2, y2, l//2, group, n-1)

generate\_KAHTOP(x3, y3, l//2, group, n-1)

generate\_KAHTOP(x4, y4, l//2, group, n-1)

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

elem.update(

\*\*{

"style": {"stroke": "#000000", "stroke-width": 1, "fill": "white"},

"inkscape:label": 'KAHTOP',

"d": ' M ' + str(x1) + ',' + str(y1) + ' L ' + str(x2) + ',' + str(y2) + ' L ' + str(x3) + ',' + str(y3) + ' L ' + str(x4) + ',' + str(y4) + 'z'

}

)

def draw\_KAHTOP(x, y, parent, l, n):

group = generate\_KAHTOP(x, y, l, parent, n)

return group

class KAHTOP(inkex.EffectExtension):

def add\_arguments(self, pars):

pars.add\_argument("--n", type=int, default=3.0, help="Глубина рекурсии")

pars.add\_argument("--l", type=float, default=100.0, help="Сторона квадрата")

pars.add\_argument("--x", type=float, default=100.0, help="X центра")

pars.add\_argument("--y", type=float, default=100.0, help="Y центра")

def effect(self):

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

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

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

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

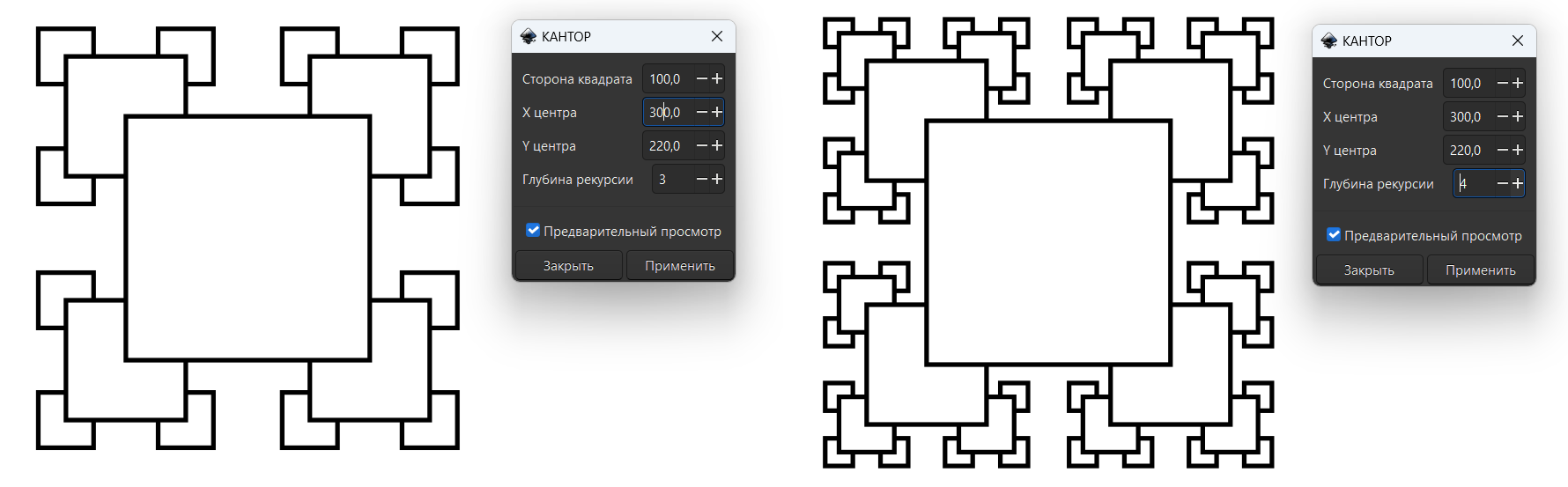
n = self.options.n

draw\_KAHTOP(x, y, grp, l, n)

if \_\_name\_\_ == "\_\_main\_\_":

KAHTOP().run()

Снимки результатов множества кантора



Эскиз снежинки коха

Изображение выглядит как диаграмма, линия, График, текст

Автоматически созданное описание

Текст расширения для снежинки коха

Файл inx

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

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

<name>KOX</name>

<id>KOX</id>

<param name="l" type="float" min="10" max="1000" gui-text="Сторона снежинки">100.0</param>

<param name="x" type="float" min="0" max="1000" gui-text="X центра">100.0</param>

<param name="y" type="float" min="0" max="1000" gui-text="Y центра">100.0</param>

<param name="n" type="int" min="1" max="10000" gui-text="Глубина рекурсии">3</param>

<effect>

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

<effects-menu>

<submenu name="Render"/>

</effects-menu>

</effect>

<script>

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

</script>

</inkscape-extension>

Файл py

#!/usr/bin/env python

# coding=utf-8

import sys

import inkex

import math

def generation\_KOX(x,y,l,n, grp):

s=""

if n == 0:

draw\_KOX(l, x, y, grp)

else:

s = s + generation\_KOX(x-(l/2), y+(l/2), l/2, n-1, grp)

s = s + generation\_KOX(x+(l/2), y+(l/2), l/2, n-1, grp)

s = s + generation\_KOX(x+(l/2), y-(l/2), l/2, n-1, grp)

s = s + generation\_KOX(x-(l/2), y-(l/2), l/2, n-1, grp)

return s

def draw\_KOX(cur, d):

style = {'stroke': '#000000', 'stroke-width': '1', 'fill': 'none'}

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

elem.update(

\*\*{

"style": style,

"inkscape:label": 'KOX',

"d": d

}

)

return elem

class KOX(inkex.EffectExtension):

def add\_arguments(self, pars):

pars.add\_argument("--n", type=int, default=3.0, help="Глубина рекурсии")

pars.add\_argument("--l", type=float, default=600.0, help="Сторона снежинки")

pars.add\_argument("--x", type=float, default=100.0, help="X центра")

pars.add\_argument("--y", type=float, default=600.0, help="Y центра")

self.\_\_pos =[0,0]

self.\_\_heading = 0

self.\_\_path = ""

self.\_\_draw = True

self.\_\_new = True

def forward(self, mag):

self.setpos((self.\_\_pos[0] + math.cos(math.radians(self.\_\_heading)) \* mag,

self.\_\_pos[1] + math.sin(math.radians(self.\_\_heading)) \* mag))

def right(self, deg):

self.\_\_heading -= deg

def left(self, deg):

self.\_\_heading += deg

def setpos(self, arg):

if self.\_\_new:

self.\_\_path += "M" + ",".join([str(i) for i in self.\_\_pos])

self.\_\_new = False

self.\_\_pos = arg

if self.\_\_draw:

self.\_\_path += "L" + ",".join([str(i) for i in self.\_\_pos])

def getPath(self):

return self.\_\_path

def KOX\_curve(self, l, n):

if n == 0:

self.forward(l)

else:

self.KOX\_curve(l/3, n-1)

self.left(60)

self.KOX\_curve(l/3, n-1)

self.right(120)

self.KOX\_curve(l/3, n-1)

self.left(60)

self.KOX\_curve(l/3, n-1)

def drawer\_KOX(self,l, n):

for i in range(3):

self.KOX\_curve(l, n)

self.right(120)

def effect(self):

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

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

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

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

n = self.options.n

self.\_\_pos =[x,y]

self.drawer\_KOX(l,n)

d = self.getPath()

draw\_KOX(grp, d)

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

KOX().run()

Снимки результатов снежинки коха

Изображение выглядит как снимок экрана, текст, линия

Автоматически созданное описание

Текст расширения для фрактального дерева

Файл inx

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

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

<name>DEPEBO</name>

<id>DEPEBO</id>

<param name="l" type="float" min="10" max="10000" gui-text="Длина ветви">100.0</param>

<param name="x" type="float" min="10" max="10000" gui-text="X центра">100.0</param>

<param name="y" type="float" min="10" max="10000" gui-text="Y центра">100.0</param>

<effect>

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

<effects-menu>

<submenu name="Render"/>

</effects-menu>

</effect>

<script>

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

</script>

</inkscape-extension>

Файл py

#!/usr/bin/env python3

# coding=utf-8

import sys

import inkex

import math

import random

from random import randint

pi = math.pi

def generate\_DEPEBO(a0, a, x, y, l, parent):

a1 = a0 + a / 2

a2 = a0 - a / 2

a3 = a0

x1 = x + math.cos(a1) \* l

y1 = y - math.sin(a1) \* l

x2 = x + math.cos(a2) \* l

y2 = y - math.sin(a2) \* l

x3 = x + math.cos(a0) \* l

y3 = y - math.sin(a0) \* l

group = parent.add(inkex.Group())

elem1 = group.add(inkex.PathElement())

elem1.update(

\*\*{

"style": {"stroke": "#8B4513", "stroke-width": 2, "fill": "none"},

"inkscape:label": 'DEPEBO',

"d": ' M ' + str(x1) + ',' + str(y1) + ' L ' + str(x) + ',' + str(y) + ' L ' + str(x2) + ',' + str(y2) + ' M ' + str(x) + ',' + str(y) + ' L ' + str(x3) + ',' + str(y3)

}

)

if l > 2:

generate\_DEPEBO(a1, (3/4) \* a, x1, y1, random.randint(0, int(l)), group)

generate\_DEPEBO(a2, (3/4) \* a, x2, y2, random.randint(0, int(l)), group)

generate\_DEPEBO(a3, (3/4) \* a, x3, y3, random.randint(0, int(l)), group)

else:

elem2 = group.add(inkex.PathElement())

elem2.update(

\*\*{

"style": {"stroke": "Orange", "stroke-width": 2, "fill": "Orange"},

"inkscape:label": 'DEPEBO',

"d": ' M ' + str(x+1) + ',' + str(y) + ' L ' + str(x+2) + ',' + str(y-2) + ' L ' + str(x) + ',' + str(y-1) + ' L ' + str(x-2) + ',' + str(y-2) + ' L ' + str(x-1) + ',' + str(y) + ' L ' + str(x-1) + ',' + str(y+2) + ' L ' + str(x) + ',' + str(y+1) + ' L ' + str(x+2) + ',' + str(y+2) + 'z'

}

)

def draw\_DEPEBO(x, y, parent, l):

group = generate\_DEPEBO(pi/2, random.uniform(0.1, pi/2), x, y, l, parent)

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

elem.update(

\*\*{

"style": {"stroke": "#8B4513", "stroke-width": 4, "fill": "none"},

"inkscape:label": 'DEPEBO',

"d": ' M ' + str(x) + ',' + str(y) + ' L ' + str(x) + ',' + str(y+l)

}

)

return group, elem

class DEPEBO(inkex.EffectExtension):

def add\_arguments(self, pars):

pars.add\_argument("--x", type=float, default=100.0, help="X центра")

pars.add\_argument("--y", type=float, default=100.0, help="Y центра")

pars.add\_argument("--l", type=float, default=100.0, help="Длина ветви")

def effect(self):

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

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

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

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

draw\_DEPEBO(x, y, grp, l)

if \_\_name\_\_ == "\_\_main\_\_":

DEPEBO().run()

Снимки результатов фрактального дерева

