```
> restart:
  with(plots):

  #      !
    ,
  #          ,          0 . 1   0 . 0 5
  0.01  0.005
  ChZnPoZap:=2:
  ShagX:=evalf[1](1/10^ChZnPoZap);


  #
  Theta[0]:=0:
  Theta[1]:=Pi/3:
  Theta[2]:=-Pi/3:
  Theta[3]:=0:

  #
  for i from 0 to 3 do
  s[i]:=1/3:
  T[i]:=matrix(2,2,[s[i]*cos(Theta[i]),-s[i]*sin(Theta[i]),s[i]*sin
  (Theta[i]),s[i]*cos(Theta[i])]):
  end do:
  #
  v0:=matrix(2,1,[1,0]):


  #
  for j from 0 to 3 do
  A[j]:=evalf(evalm(add(evalm(T[i]&*v0),i=0..j))[1,1]);
  B[j]:=evalf(evalm(add(evalm(T[i]&*v0),i=0..j))[2,1]);
  C[j]:=evalf(s[j]*cos(Theta[j])):
  E[j]:=evalf(s[j]*sin(Theta[j])):
  end do:
  A[-1]:=0:
  B[-1]:=0:
  C[-1]:=0:
  E[-1]:=0:


  #
  KoxaPlolnostu:= proc(t)
  local zz,i,eq1,eq2,j,jj,TT:

  #
  ,         t
  j:=1:
  zz[1]:=0:
  while zz[j]<1+t do
  eq1[j],eq2[j],TT[j]:=KoxaUrav(zz[j]):
  zz[j+1]:=zz[j]+t:
  j:=j+1:
  end do:

  #
```

```
subs(x[0.]=0,y[0.]=0,solve({seq(eq1[jj],jj=1..j-1),seq(eq2[jj],
jj=1..j-1)},{seq(x[zz[jj]],jj=1..j-1),seq(y[zz[jj]],jj=1..j-1)}))
;

#    x , y      t
subs(%,[seq([x[zz[jj]],y[zz[jj]],zz[jj]],jj=1..j-1)]);
end proc:

#
  t
KoxaUrav := proc(t)
option remember;
local eq1,eq2,i,T,qq,k:
T:=t:
#
k:=4:
qq:=k*T-trunc(k*T):
eq1:=x[t]=A[trunc(k*T)-1]+C[trunc(k*T)]*x[qq]-E[trunc(k*T)]*y[qq]
:
eq2:=y[t]=B[trunc(k*T)-1]+E[trunc(k*T)]*x[qq]+C[trunc(k*T)]*y[qq]
:
eq1,eq2,T;
end proc:
SaveFractalNotWar:=KoxaPlolnostu(ShagX):
ChisloUzlovFractala:=nops(SaveFractalNotWar)-1:

KoxaFunction:= proc(t)
option remember;
local i:

for i from 0 to ChisloUzlovFractala do
#print(i/ChisloUzlovFractala):
if t=i/ChisloUzlovFractala then
RETURN([SaveFractalNotWar[i+1,1],SaveFractalNotWar[i+1,2]]);
fi:
od:
NULL:
end proc:
```

$$ShagX := 0.01 \tag{1}$$

```
> SaveFractalNotWar
```

$$[[0., 0., 0], [0.02407645757, 0.006013250501, 0.01], [0.04132231404, 0.01183094814, 0.02], [0.05212143792, 0.02875604833, 0.03], [0.07222937271, 0.01803975151, 0.04], [0.08333333332, 0., 0.05], [0.1030856252, 0.002004416833, 0.06], [0.1114968613, 0.01983883783, 0.07], [0.1239669421, 0.03549284442, 0.08], [0.1166452152, 0.05411925452, 0.09], [0.1250000000, 0.07216878368, 0.10], [0.1489071038, 0.06680085761, 0.11], [0.1563643138, 0.08626814501, 0.12], [0.1769690195, 0.08626814501, 0.13], [0.1844262295, 0.06680085761, 0.14], [0.2083333333, 0.07216878368, 0.15], [0.2166881181, 0.05411925452, 0.16], [0.2093663912, 0.03549284442, 0.17], [0.2218364720, 0.01983883783, 0.18], [0.2302477081, 0.002004416833, 0.19], [0.2500000000, 0., 0.20], [0.2611039606, 0.01803975151, 0.21], [0.2812118954, 0.02875604833, 0.22], [0.2920110193, 0.01183094814, 0.23], [0.3092568757, 0.006013250501, 0.24], [0.3333333333, 0., 0.25], [0.3401639344, \tag{2}$$

0.02385744915, 0.26], [0.3437485887, 0.04170164779, 0.27], [0.3344905839, 0.05951651351, 0.28], [0.3538251366, 0.07157234745, 0.29], [0.3750000000, 0.07216878367, 0.30], [0.3831402700, 0.09027697868, 0.31], [0.3719008264, 0.1064785333, 0.32], [0.3645790994, 0.1251049434, 0.33], [0.3447872916, 0.1280773469, 0.34], [0.3333333332, 0.1443375673, 0.35], [0.3499356455, 0.1623577636, 0.36], [0.3368050851, 0.1785495405, 0.37], [0.3471074379, 0.1963937392, 0.38], [0.3676952084, 0.1931182288, 0.39], [0.3749999999, 0.2165063511, 0.40], [0.3948087431, 0.2147170424, 0.41], [0.4072788240, 0.1990630358, 0.42], [0.4270706318, 0.2020354393, 0.43], [0.4467213115, 0.2004025729, 0.44], [0.4583333333, 0.2165063510, 0.45], [0.4482624305, 0.2351425388, 0.46], [0.4490358127, 0.2579146695, 0.47], [0.4690929413, 0.2588044351, 0.48], [0.4827541435, 0.2708309361, 0.49], [0.5000000000, 0.2886751347, 0.50], [0.5172458565, 0.2708309361, 0.51], [0.5309070587, 0.2588044351, 0.52], [0.5509641873, 0.2579146695, 0.53], [0.5517375695, 0.2351425388, 0.54], [0.5416666667, 0.2165063510, 0.55], [0.5532786885, 0.2004025729, 0.56], [0.5729293682, 0.2020354393, 0.57], [0.5927211760, 0.1990630358, 0.58], [0.6051912569, 0.2147170424, 0.59], [0.6250000001, 0.2165063511, 0.60], [0.6323047916, 0.1931182288, 0.61], [0.6528925621, 0.1963937392, 0.62], [0.6631949149, 0.1785495405, 0.63], [0.6500643545, 0.1623577636, 0.64], [0.6666666668, 0.1443375673, 0.65], [0.6552127084, 0.1280773469, 0.66], [0.6354209006, 0.1251049434, 0.67], [0.6280991736, 0.1064785333, 0.68], [0.6168597300, 0.09027697868, 0.69], [0.6250000000, 0.07216878367, 0.70], [0.6461748634, 0.07157234745, 0.71], [0.6655094161, 0.05951651351, 0.72], [0.6562514113, 0.04170164779, 0.73], [0.6598360656, 0.02385744915, 0.74], [0.6666666667, 0., 0.75], [0.6907431243, 0.006013250501, 0.76], [0.7079889807, 0.01183094814, 0.77], [0.7187881046, 0.02875604833, 0.78], [0.7388960394, 0.01803975151, 0.79], [0.7500000000, 0., 0.80], [0.7697522919, 0.002004416833, 0.81], [0.7781635280, 0.01983883783, 0.82], [0.7906336088, 0.03549284442, 0.83], [0.7833118819, 0.05411925452, 0.84], [0.7916666667, 0.07216878368, 0.85], [0.8155737705, 0.06680085761, 0.86], [0.8230309805, 0.08626814501, 0.87], [0.8436356862, 0.08626814501, 0.88], [0.8510928962, 0.06680085761, 0.89], [0.8750000000, 0.07216878368, 0.90], [0.8833547848, 0.05411925452, 0.91], [0.8760330579, 0.03549284442, 0.92], [0.8885031387, 0.01983883783, 0.93], [0.8969143748, 0.002004416833, 0.94], [0.9166666667, 0., 0.95], [0.9277706273, 0.01803975151, 0.96], [0.9478785621, 0.02875604833, 0.97], [0.9586776860, 0.01183094814, 0.98], [0.9759235424, 0.006013250501, 0.99], [1., 0, 1.00]]

> 0, 0.012, 0.016000, 0.044000, 0.044000, 0.045000, 0.053000, 0.055000, 0.070000, 0.071000, 0.073000, 0.076000, 0.080000, 0.082000, 0.10600, 0.10800, 0.11000, 0.14000, 0.14100, 0.16900, 0.16900, 0.17000, 0.17800, 0.17900, 0.18000, 0.19500, 0.19700, 0.20500, 0.20700, 0.23100, 0.23200, 0.24000, 0.24100, 0.24300, 0.24400, 0.25600, 0.25800, 0.27300, 0.27600, 0.27800, 0.28500, 0.28800, 0.28900, 0.33600, 0.33700, 0.33700, 0.33900, 0.34000, 0.34700, 0.34800, 0.35500, 0.35700, 0.36100, 0.36200, 0.36300, 0.38700, 0.38700, 0.38800, 0.38900,

```
    0.39300, 0.39400, 0.41800, 0.41900, 0.42000, 0.42800, 0.42900, 0.43100, 0.43500, 0.43600,
    0.43900, 0.44000, 0.44200, 0.44800, 0.45500, 0.45700, 0.48100, 0.48200, 0.48300, 0.48300,
    0.48600, 0.48700, 0.48800, 0.51200, 0.51200, 0.51400, 0.51700, 0.51800, 0.52000, 0.52700,
    0.52800, 0.53500, 0.53600, 0.53800, 0.53900, 0.58600, 0.58700, 0.59000, 0.59700, 0.59800,
    0.60500, 0.60700, 0.60800, 0.61100, 0.61300, 0.63700, 0.63800, 0.63900, 0.64300, 0.64400,
    0.66800, 0.67000, 0.67800, 0.68000, 0.71100, 0.71500, 0.72200, 0.72400, 0.72500, 0.72600,
    0.77500, 0.77600, 0.77700, 0.77800, 0.78500, 0.78600, 0.78900, 0.83600, 0.83700, 0.83800,
    0.84100, 0.84700, 0.84700, 0.84900, 0.85300, 0.86600, 0.87100, 0.88700, 0.88800, 0.88900,
    0.89300, 0.89400, 0.91800, 0.92000, 0.92800, 0.93000, 0.94500, 0.94700, 0.95500, 0.95600,
    0.95700, 0.98100, 0.99000, 0.99, 1.0
> with(Statistics):


  #
  FromA:={0, 0.12e-1, 0.16000e-1, 0.44000e-1, 0.44000e-1,
  0.45000e-1, 0.53000e-1, 0.55000e-1, 0.70000e-1, 0.71000e-1,
  0.73000e-1, 0.76000e-1, 0.80000e-1, 0.82000e-1, .10600, .10800,
  .11000, .14000, .14100, .16900, .16900, .17000, .17800, .17900,
  .18000, .19500, .19700, .20500, .20700, .23100, .23200, .24000,
  .24100, .24300, .24400, .25600, .25800, .27300, .27600, .27800,
  .28500, .28800, .28900, .33600, .33700, .33700, .33900, .34000,
  .34700, .34800, .35500, .35700, .36100, .36200, .36300, .38700,
  .38700, .38800, .38900, .39300, .39400, .41800, .41900, .42000,
  .42800, .42900, .43100, .43500, .43600, .43900, .44000, .44200,
  .44800, .45500, .45700, .48100, .48200, .48300, .48300, .48600,
  .48700, .48800, .51200, .51200, .51400, .51700, .51800, .52000,
  .52700, .52800, .53500, .53600, .53800, .53900, .58600, .58700,
  .59000, .59700, .59800, .60500, .60700, .60800, .61100, .61300,
  .63700, .63800, .63900, .64300, .64400, .66800, .67000, .67800,
  .68000, .71100, .71500, .72200, .72400, .72500, .72600, .77500,
  .77600, .77700, .77800, .78500, .78600, .78900, .83600, .83700,
  .83800, .84100, .84700, .84700, .84900, .85300, .86600, .87100,
  .88700, .88800, .88900, .89300, .89400, .91800, .92000, .92800,
  .93000, .94500, .94700, .95500, .95600, .95700, .98100, .99000,
  .99, 1.0  }:




  #
  delta:=evalf(0.1):
  SubdivizionTrue[1]:=0:
  i:=1:

  #
  while SubdivizionTrue[i]<1 do
  SubdivizionTrue[i+1]:=evalf[2](SubdivizionTrue[i]+delta/4):
  i:=i+1:
  od:
  N:=i:print(%):

  (*
  for i from 1 to 147 do
  SubdivizionTrue[i]:=FromA[i]:
  od:
```

```
N:=147:

*)


k:=0:
while k<300 do
i:='i':
j:='j':
SubdivizionShtrih1:=0:
#
#
x1:=evalf(Sample(RandomVariable(Uniform(0, 1)), 1)[1]):
y1:=evalf(Sample(RandomVariable(Uniform(0, 1)), 1)[1]):

#
while evalf[5](x1)=evalf[5](y1) do
y1:=evalf(Sample(RandomVariable(Uniform(0, 1)), 1)[1]):
od:

y:=max(x1,y1); x:=min(x1,y1);
#
j:=1:
for i from 1 to N do
if SubdivizionTrue[i]>=x and SubdivizionTrue[i]<=y then
SubdivizionShtrih1[j]:=SubdivizionTrue[i]:
j:=j+1:
#
if j=2 then
iStart:=i:
fi:

fi:
od:
#   ,
m:=j-3:
#
iFinish:=iStart+m+1:



keyOnlyOneWay:=0:

#            3

if m>1 then
#-------------------------------------------------------------
- - - - - -
RandomVariableForProbability:=Sample(RandomVariable(Uniform(0, 1)
), 1)[1]:
pC:=min(1,delta/(y-x));
#     ,
if RandomVariableForProbability<evalf(pC) and keyOnlyOneWay=0
then
#       (   )
```

```
key:=-1:
while key<>SubdivizionShtrih1[m+2] do
SubdivizionShtrih1Shift:=0:
SubdivizionShtrih1Shift[1]:=SubdivizionShtrih1[1]:
RandomVariableForShift:=Sample(RandomVariable(Uniform(-delta/10,
delta/10)), m+5):
for i from 2 to m+1 do
SubdivizionShtrih1Shift[i]:=abs(evalf[5](trunc(10^ChZnPoZap*
(SubdivizionShtrih1[i]+RandomVariableForShift[i]))/10^ChZnPoZap))
:
od:
SubdivizionShtrih1Shift[m+2]:=SubdivizionShtrih1[m+2]:
SubdivizionShtrih1Shift:=sort([seq(SubdivizionShtrih1Shift[pp],
pp=1..m+2)]);
#print([seq(SubdivizionShtrih1Shift[qqq],qqq=1..m+2)]):
key:=SubdivizionShtrih1Shift[m+2]:
od:

sssss:=0:


#
,
diamShtrih1Shift:=evalf(max(seq(SubdivizionShtrih1Shift[ii]-
SubdivizionShtrih1Shift[ii-1],ii=2..m+2))):
if diamShtrih1Shift<=delta then
for i from 1 to m+2 do
SubdivizionShtrih2[i]:=SubdivizionShtrih1Shift[i]:
od:

#
for i from 1 to iStart do
Subdivizion1[i]:=SubdivizionTrue[i]:
od:
j:=2:
for i from iStart+1 to iFinish-1 do
Subdivizion1[i]:=SubdivizionShtrih2[j]:
j:=j+1:
od:

for i from iFinish to N do
Subdivizion1[i]:=SubdivizionTrue[i]:
od:

(*
print(x,y,iStart,iFinish,m,j-1);
print(seq([SubdivizionTrue[ii],ii],ii=iStart..iFinish)):
print(seq([Subdivizion1[ii],ii],ii=iStart..iFinish)):
k:=10000:
*)



#
sigma1:=add(evalf((sqrt((KoxaFunction(Subdivizion1[jj+1])[1]-
KoxaFunction(Subdivizion1[jj])[1])^2+(KoxaFunction(Subdivizion1
[jj+1])[2]-KoxaFunction(Subdivizion1[jj])[2])^2))^(ln(4)/ln(3))),
```

```
jj=1..N-1):
sigma:=add(evalf((sqrt((KoxaFunction(SubdivizionTrue[jj+1])[1]-
KoxaFunction(SubdivizionTrue[jj])[1])^2+(KoxaFunction
(SubdivizionTrue[jj+1])[2]-KoxaFunction(SubdivizionTrue[jj])[2])
^2))^(ln(4)/ln(3))),jj=1..N-1):



#
       k
if sigma1<sigma then
k:=k+1:
if Subdivizion1[N]<0.99 then
p r i n t ( " ! " ) :
print(x,y,iStart,iFinish,m,j-1);
print(seq([SubdivizionTrue[ii],ii],ii=iStart..iFinish)):
print(seq([Subdivizion1[ii],ii],ii=iStart..iFinish)):
k:=10000:
fi:
for i from 1 to N do
SubdivizionTrue[i]:=Subdivizion1[i]:
od:
p r i n t ( "        " ) :
print(sigma1):
p r i n t ( "   " ) :
print(k):
keyOnlyOneWay:=0:
fi:



keyOnlyOneWay:=0:
fi:
fi:

#----------------------------------------------------------------
---------- B
#
RandomVariableForProbability:=Sample(RandomVariable(Uniform(0, 1)
), 1)[1]:
pD:=min(1,delta/(y-x));
#     ,
if RandomVariableForProbability<evalf(pD) and keyOnlyOneWay=0
then
#       (   )
SubdivizionShtrih1Delete[1]:=SubdivizionShtrih1[1]:
SubdivizionShtrih1Delete[2]:=SubdivizionShtrih1[m+2]:


#
,
diamShtrih1Delete:=evalf(SubdivizionShtrih1Delete[2]-
SubdivizionShtrih1Delete[1]):
if diamShtrih1Delete<=delta then
SubdivizionShtrih2[1]:=SubdivizionShtrih1Delete[1]:
SubdivizionShtrih2[2]:=SubdivizionShtrih1Delete[2]:
```

```
#
for i from 1 to iStart-1 do
Subdivizion1[i]:=SubdivizionTrue[i]:
od:

Subdivizion1[iStart]:=SubdivizionShtrih2[1]:
Subdivizion1[iStart+1]:=SubdivizionShtrih2[2]:

jjj:=iStart+2:
for i from iFinish+1 to N do
Subdivizion1[jjj]:=SubdivizionTrue[i]:
jjj:=jjj+1:
od:
jjj:=jjj-1:



#
sigma1:=add(evalf((sqrt((KoxaFunction(Subdivizion1[jj+1])[1]-
KoxaFunction(Subdivizion1[jj])[1])^2+(KoxaFunction(Subdivizion1
[jj+1])[2]-KoxaFunction(Subdivizion1[jj])[2])^2))^(ln(4)/ln(3))),
jj=1..jjj-1):
sigma:=add(evalf((sqrt((KoxaFunction(SubdivizionTrue[jj+1])[1]-
KoxaFunction(SubdivizionTrue[jj])[1])^2+(KoxaFunction
(SubdivizionTrue[jj+1])[2]-KoxaFunction(SubdivizionTrue[jj])[2])
^2))^(ln(4)/ln(3))),jj=1..N-1):
#sigma1:=0:
#sigma:=1:


#
       k
if sigma1<sigma then
k:=k+1:
SubdivizionTrue:=0:
for i from 1 to jjj do
SubdivizionTrue[i]:=Subdivizion1[i]: #print(%);
od:#1 to j
p r i n t ( "       " ) :
print(sigma1):
N:=jjj: print(N):
p r i n t ( "  " ) :
print(k):
keyOnlyOneWay:=0:
fi:#sigma1<sigma
fi: #diamShtrih1Delete<=delta

keyOnlyOneWay:=0:
fi:#RandomVariableForProbability<evalf(pD)


#----------------------------------------------------------------
- - - - - - - - - -    (                                    )
#
RandomVariableForProbability:=Sample(RandomVariable(Uniform(0, 1)
), 1)[1]:
#RandomVariableForProbability:=0:
```

```
pI:=min(1,delta/(y-x));
#     ,
if RandomVariableForProbability<evalf(pI) and keyOnlyOneWay=0
then

#
i:=1:
j:=1:
while j<=m do
if SubdivizionShtrih1[j+1]-SubdivizionShtrih1[j]>delta/10 then
SubdivizionShtrih1Insert[i]:=SubdivizionShtrih1[j]:
i:=i+1:
SubdivizionShtrih1Insert[i]:=SubdivizionShtrih1Insert[i-1]:
while abs(SubdivizionShtrih1Insert[i]-SubdivizionShtrih1Insert
[i-1])=0 do
SubdivizionShtrih1Insert[i]:=evalf[5](trunc(10^ChZnPoZap*(Sample
(RandomVariable(Uniform(SubdivizionShtrih1[j], SubdivizionShtrih1
[j+1])), 1)[1]))/10^ChZnPoZap):
od:
i:=i+1:
else
SubdivizionShtrih1Insert[i]:=SubdivizionShtrih1[j]:
i:=i+1:
fi:
j:=j+1:
od:
SubdivizionShtrih1Insert[i]:=SubdivizionShtrih1[j]:
NN:=i;


#
,
diamShtrih1Insert:=evalf(max(seq(SubdivizionShtrih1Insert[ii]-
SubdivizionShtrih1Insert[ii-1],ii=2..NN))):
if diamShtrih1Insert<=delta then

for i from 1 to NN do
SubdivizionShtrih2[i]:=SubdivizionShtrih1Insert[i]:
end do:


#
for i from 1 to iStart do
Subdivizion1[i]:=SubdivizionTrue[i]:
od:
j:=iStart:
for i from 1 to NN do
Subdivizion1[j]:=SubdivizionShtrih2[i]:
j:=j+1:
end do:


for i from iFinish to N do
Subdivizion1[j]:=SubdivizionTrue[i]:
j:=j+1:
od:
j:=j-1:
```

```
#
sigma1:=add(evalf((sqrt((KoxaFunction(Subdivizion1[jj+1])[1]-
KoxaFunction(Subdivizion1[jj])[1])^2+(KoxaFunction(Subdivizion1
[jj+1])[2]-KoxaFunction(Subdivizion1[jj])[2])^2))^(ln(4)/ln(3))),
jj=1..j-1):
sigma:=add(evalf((sqrt((KoxaFunction(SubdivizionTrue[jj+1])[1]-
KoxaFunction(SubdivizionTrue[jj])[1])^2+(KoxaFunction
(SubdivizionTrue[jj+1])[2]-KoxaFunction(SubdivizionTrue[jj])[2])
^2))^(ln(4)/ln(3))),jj=1..N-1):
#sigma1:=0:
#sigma:=1:


#
        k
if sigma1<sigma then
k:=k+1:
SubdivizionTrue:=0:
for i from 1 to j do
SubdivizionTrue[i]:=Subdivizion1[i]: #print(%);
od:#1 to j
p r i n t ( "        C " ) :
print(sigma1):
N:=j:print(N):
p r i n t ( "   " ) :
print(k):
keyOnlyOneWay:=0:
fi:#sigma1<sigma


keyOnlyOneWay:=1:
fi: #diamShtrih1Insert<=delta
fi:#RandomVariableForProbability<evalf(pI)



fi:#m>3
keyOnlyOneWay:=0:

if SubdivizionTrue[N]<0.98 then
p r i n t ( " ! " ) :
k:=100:
fi:

od:# k<...
                                50
                "       C "
                              0.7712571473
                                53
                           "   "
                                 1
                "       "
                              0.7665173727
                           "   "
```

2
" "
0.7564586104
" "
3
" C "
0.7532323695
65
" "
4
" C "
0.7419064175
76
" "
5
" "
0.7291861936
74
" "
6
" "
0.7283327334
" "
7
" "
0.7277521416
" "
8
" "
0.7251211936
" "
9
" "
0.7220192622
" "
10
" "
0.7207937992
" "
11
" "
0.7185157624

65
" "
12
" "
0.7161494566
" "
13
" "
0.7159729004
" "
14
" C "
0.7134213943
69
" "
15
" "
0.6982447604
67
" "
16
" "
0.6971380988
" "
17
" "
0.6933415298
64
" "
18
" C "
0.6908080523
69
" "
19
" C "
0.6897390106
70
" "
20
" "
0.6861890473

62
" "
21
" "
0.6844378397
58
" "
22
" "
0.6825385006
" "
23
" "
0.6805314102
54
" "
24
" C "
0.6798331489
56
" "
25
" "
0.6731539321
" "
26
" C "
0.6661338641
66
" "
27
" "
0.6633971703
" "
28
" "
0.6564707265
" "
29
" "
0.6553261483
64

" "
30
" "
0.6518081893
" "
31
" "
0.6475643193
" "
32
" C "
0.6462951981
66
" "
33
" "
0.6457985141
" "
34
" "
0.6406375039
" "
35
" C "
0.6402211464
67
" "
36
" C "
0.6382716552
72
" "
37
" "
0.6378810478
" "
38
" C "
0.6351547003
74
" "
39

" "
0.6334771478
62
" "
40
" "
0.6320740872
" "
41
" "
0.6320200990
" "
42
" C "
0.6297305952
65
" "
43
" "
0.6290166574
61
" "
44
" C "
0.6286086108
62
" "
45
" "
0.6276032690
" "
46
" "
0.6263032382
60
" "
47
" "
0.6261820450
" "
48
" C "

0.6243433384
61
" "
49
" "
0.6236930430
" "
50
" "
0.6193430236
59
" "
51
" C "
0.6185590353
60
" "
52
" "
0.6105599377
58
" "
53
" "
0.6091795031
" "
54
" "
0.6088628620
" "
55
" "
0.6021175590
46
" "
56
" "
0.6015297447
" "
57
" "
0.5997234475

" "
58
" "
0.5977795706
43
" "
59
" "
0.5960598856
" "
60
" "
0.5955492360
" "
61
" C "
0.5949316378
44
" "
62
" "
0.5948364218
" "
63
" "
0.5939028558
" "
64
" C "
0.5934797172
45
" "
65
" "
0.5930457130
" "
66
" "
0.5928636095
" "
67
" C "

0.5922369039
46
" "
68
" C "
0.5899999741
47
" "
69
" "
0.5848625781
" "
70
" C "
0.5834011301
49
" "
71
" C "
0.5830987579
50
" "
72
" "
0.5830239517
43
" "
73
" "
0.5829774697
" "
74
" "
0.5828760207
" "
75
" "
0.5818621503
" "
76
" C "
0.5817011487

44
" "
77
" "
0.5814735117
" "
78
" C "
0.5791120259
45
" "
79
" "
0.5785682971
" "
80
" C "
0.5714273559
47
" "
81
" C "
0.5702033739
48
" "
82
" C "
0.5691138576
49
" "
83
" C "
0.5683791699
50
" "
84
" "
0.5683624711
" "
85
" C "
0.5681390181

51
" "
86
" C "
0.5676822835
52
" "
87
" C "
0.5667676645
53
" "
88
" "
0.5667151319
" "
89
" C "
0.5657414056
54
" "
90
" C "
0.5650074494
55
" "
91
" C "
0.5622008519
56
" "
92
" C "
0.5615050515
57
" "
93
" "
0.5600930378
" "
94
" C "

0.5589373644
58
" "
95
" "
0.5588091779
52
" "
96
" C "
0.5581114879
53
" "
97
" C "
0.5569606823
54
" "
98
" "
0.5569237254
" "
99
" "
0.5562779291
" "
100
" C "
0.5551167979
55
" "
101
" "
0.5544247110
" "
102
" "
0.5526569912
" "
103
" "
0.5523311033

" "
104
" "
0.5511910861
" "
105
" C "
0.5500485457
56
" "
106
" C "
0.5497446620
57
" "
107
" C "
0.5484152866
59
" "
108
" C "
0.5465093095
60
" "
109
" "
0.5465054126
" "
110
" C "
0.5460669906
61
" "
111
" C "
0.5458081404
62
" "
112
" "
0.5456884783

" "
113
" C "
0.544598962163
" "
114
" "
0.544061202364
" "
115
" C "
0.543363512264
" "
116
" C "
0.542336530066
" "
117
" "
0.542329205264
" "
118
" C "
0.541594410067
" "
119
" "
0.541410878664
" "
120
" "
0.541222234464
" "
121
" C "
0.540510291068
" "

122
" "
0.5404345016
" "
123
" "
0.5383779970
" "
124
" "
0.5383070922
" "
125
" "
0.5381673714
" "
126
" "
0.5377942866
" "
127
" C "
0.5373790754
69
" "
128
" "
0.5369943022
" "
129
" C "
0.5369022298
70
" "
130
" "
0.5362035736
" "
131
" C "
0.5358006062
71

" "
132
" "
0.5353578090
" "
133
" C "
0.5338742060
73
" "
134
" "
0.5337625318
" "
135
" C "
0.5313755988
74
" "
136
" "
0.5311493629
" "
137
" C "
0.5309947775
75
" "
138
" "
0.5296219847
" "
139
" "
0.5296002982
" "
140
" C "
0.5294347438
76
" "
141

" C "
0.528974208477
" "
142
" "
0.528928375243
" "
143
" C "
0.528697288078
" "
144
" "
0.527999839244
" "
145
" "
0.527892038846
" "
146
" C "
0.527800332079
" "
147
" "
0.527649942246
" "
148
" C "
0.527391092079
" "
149
" "
0.527080225243
" "
150
" "
0.526611126477

" "
151
" C "
0.5246357420
81
" "
152
" "
0.5243787476
" "
153
" "
0.5240869770
" "
154
" C "
0.5240259286
83
" "
155
" "
0.5236811032
" "
156
" C "
0.5235890308
84
" "
157
" "
0.5234955650
" "
158
" "
0.5228287426
" "
159
" "
0.5226764980
" "
160
" "

0.5222988687
" "
161
" C "
0.5222975053
85
" "
162
" "
0.5220291579
" "
163
" "
0.5219701513
" "
164
" "
0.5219426021
" "
165
" "
0.5216506791
" "
166
" "
0.5215460709
" "
167
" "
0.5213976578
" "
168
" C "
0.5202098534
86
" "
169
" C "
0.5197531190
87
" "
170

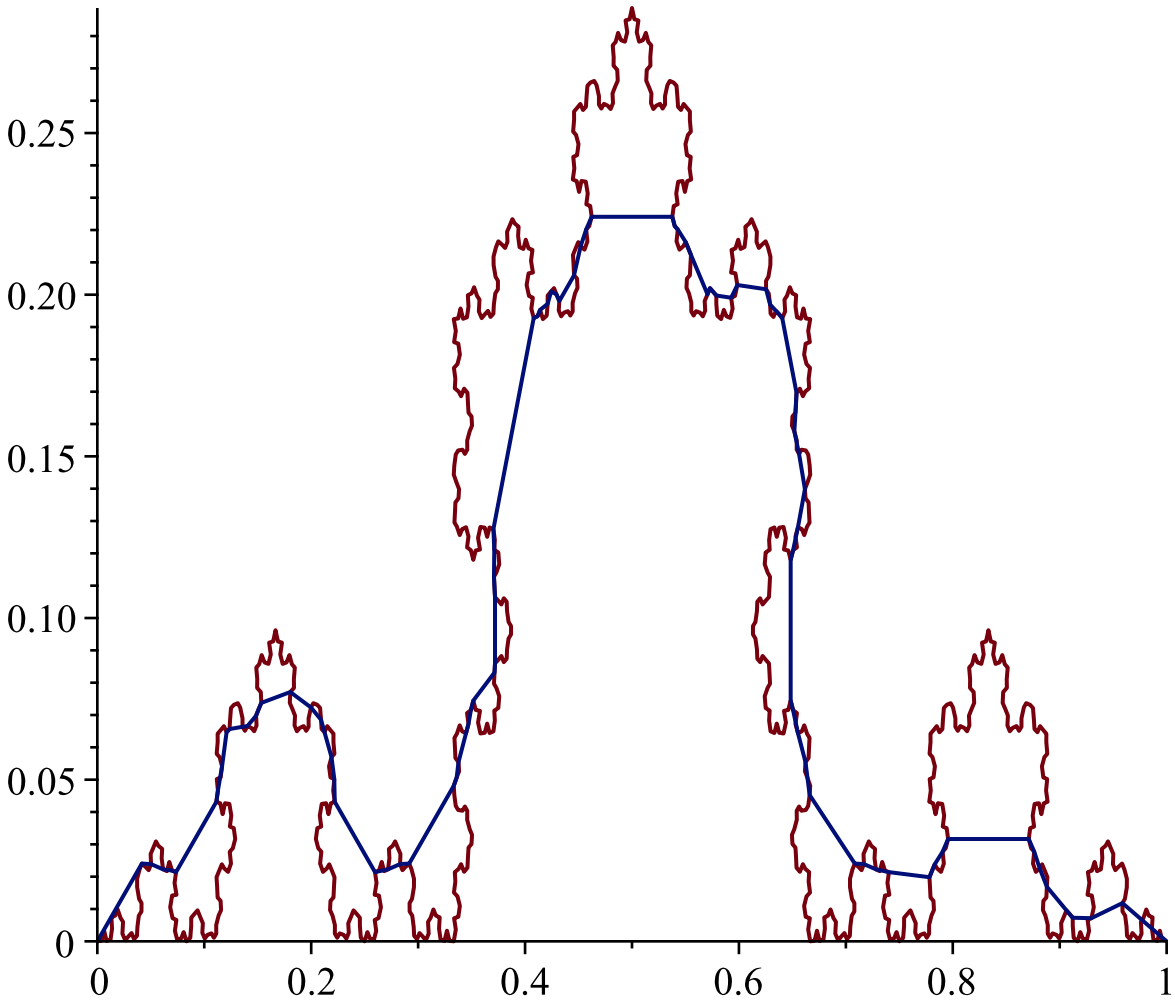
" "
0.5194846274
" "
171
" "
0.5194115548
" "
172
" "
0.5193384696
" "
173
" "
0.5193384215
" "
174
" "
0.5192543259
" "
175
" "
0.5189787199
" "
176
" "
0.5185866081
" "
177
" "
0.5180155453
" "
178
" C "
0.5171019271
88
" "
179
" "
0.5165100457
" "
180
" "

0.5163831467
" "
181
" "
0.5163433923
" "
182
" C "
0.5159955347
89
" "
183
" "
0.5158413395
" "
184
" "
0.5158329981
" "
185
" C "
0.5152702633
90
" "
186
" "
0.5152107083
" "
187
" "
0.5150147393
" "
188
" "
0.5146093117
" "
189
" C "
0.5144443307
91
" "
190

" C "
0.5139591597
92
" "
191
" "
0.5136442731
" "
192
" "
0.5135802003
" "
193
" "
0.5134109323
" "
194
" "
0.5133243763
" "
195
" "
0.5132413658
" "
196
" C "
0.5126869470
94
" "
197
" C "
0.5123555932
95
" "
198
" "
0.5120494388
" "
199
" "
0.5118931484
" "

```
                                  200
                    "       "
                                  0.5115645575
                          "   "
                                  201
                    "       "
                                  0.5112289254
                          "   "
                                  202
                    "       "
                                  0.5112205528
                          "   "
                                  203
                    "       "
                                  0.5104499564
                          "   "
                                  204
                    "       C "
                                  0.5101199942
                                   96
                          "   "
                                  205
                    "       C "
                                  0.5092893958
                                   97
                          "   "
                                  206
                    "       C "
                                  0.5086432581
                                   98
                          "   "
                                  207
                    "       "
                                  0.5083178250
                          "   "
                                  208
Warning,  computation interrupted
> for i from 1 to N do
  GGGG[i]:=KoxaFunction(SubdivizionTrue[i]):
  od:
  seq(SubdivizionTrue[i],i=1..N);
  plot([[seq([SaveFractalNotWar[i,1],SaveFractalNotWar[i,2]],i=1.
  .10^ChZnPoZap)],[seq([GGGG[i][1],GGGG[i][2]],i=1..N)]]);
```

0, 0.025, 0.028000, 0.035000, 0.038000, 0.039000, 0.086000, 0.087000, 0.090000, 0.097000, 0.098000, 0.10500, 0.10700, 0.11100, 0.11200, 0.11300, 0.13600, 0.14700, 0.15100, 0.15300, 0.15900, 0.16200, 0.16400, 0.21100, 0.21500, 0.22200, 0.22500, 0.27500, 0.27600, 0.27800, 0.28500, 0.28600, 0.28800, 0.28900, 0.30400, 0.30500, 0.30500, 0.32000, 0.32200, 0.32600, 0.32800, 0.42200, 0.42300, 0.42400, 0.42700, 0.42900, 0.43100, 0.43200, 0.44300, 0.44800, 0.45100, 0.45300, 0.54700, 0.54800, 0.54900, 0.55300, 0.55500, 0.56900, 0.57000, 0.57200, 0.58000, 0.58200, 0.60600, 0.60800, 0.61100, 0.61200, 0.61300, 0.63700, 0.63900, 0.64200, 0.64500, 0.65200, 0.66000, 0.66300, 0.66400, 0.71100, 0.71200, 0.71500, 0.72200, 0.72600, 0.77500, 0.77800, 0.78500, 0.78900, 0.82000, 0.82100, 0.82200, 0.82600, 0.82800, 0.92200, 0.92400, 0.92700, 0.93100, 0.94500, 0.94700, 0.95600, 0.98, 1.0




```
> with(Statistics):

  #
```

```
FromA:={0, 0.12e-1, 0.16000e-1, 0.44000e-1, 0.44000e-1,
0.45000e-1, 0.53000e-1, 0.55000e-1, 0.70000e-1, 0.71000e-1,
0.73000e-1, 0.76000e-1, 0.80000e-1, 0.82000e-1, .10600, .10800,
.11000, .14000, .14100, .16900, .16900, .17000, .17800, .17900,
.18000, .19500, .19700, .20500, .20707, .23100, .23200, .24000,
.24100, .24300, .24400, .25600, .25800, .27300, .27600, .27800,
.28500, .28800, .28900, .33600, .33700, .33700, .33900, .34000,
.34700, .34800, .35500, .35700, .36100, .36200, .36300, .38700,
.38700, .38800, .38900, .39300, .39400, .41800, .41900, .42000,
.42800, .42900, .43100, .43500, .43600, .43900, .44000, .44200,
.44800, .45500, .45700, .48100, .48200, .48300, .48300, .48600,
.48700, .48800, .51200, .51200, .51400, .51700, .51800, .52000,
.52700, .52800, .53500, .53600, .53800, .53900, .58600, .58700,
.59000, .59700, .59800, .60500, .60700, .60800, .61100, .61300,
.63700, .63800, .63900, .64300, .64400, .66800, .67000, .67800,
.68000, .71100, .71500, .72200, .72400, .72500, .72600, .77500,
.77600, .77700, .77800, .78500, .78600, .78900, .83600, .83700,
.83800, .84100, .84700, .84700, .84900, .85300, .86600, .87100,
.88700, .88800, .88900, .89300, .89400, .91800, .92000, .92800,
.93000, .94500, .94700, .95500, .95600, .95700, .98100, .99000,
.99, 1.0  }:
                    "      "
                                    0.5082750569
                                "   "
                                        1
                    "      "
                                    0.5080219322
                                "   "
                                        2
                    "      "
                                    0.5080135596
                                "   "
                                        3
Warning,  computation interrupted
> evalm(T[1]&*v0)[1,1]; evalf(%)

                                    1
                                    -
                                    6
Warning, inserted missing semicolon at end of statement
                                0.1666666667                    (3)
```