PENYAJIAN GRAFIK DAN LUAS FRAKTAL KOCH SNOWFLAKE PADA LEVEL KE - *n*

SKRIPSI

Diajukan untuk Memenuhi Persyaratan Penyelesaian Program Sarjana Sains
Jurusan Matematika Fakultas Matematika dan Ilmu Pengetahuan Alam
Universitas Jember

Oleh :

Agung Kriswantoro
NIM. 201810101041

JURUSAN MATEMATIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS JEMBER
Oktober 2004

MOTTO

Wujudkanlah pikiran yang bebas
(Master Zhongfeng)

Teruslah berjalan walau hanya selangkah, karena itu lebih baik daripada hanya berpikir dan berdiam diri
(Agung Kriswantoro)

Lihatlah segala sesuatu dengan pikiran dan hati yang jernih, baca, baca dan bacalah apa yang engkau lihat maka engkau akan menemukan kebenaran dan kebahagiaan yang kau cari
(Agung Kriswantoro)

Tumbuhkan rasa suka dan cinta untuk segala hal yang kau kerjakan, dengan demikian segalanya akan terasa menyenangkan dan tanpa beban
(Agung Kriswantoro)

Jadikanlah kekurangan yang ada pada diri kita sebagai kelebihan yang tidak dimiliki orang lain, karena Allah telah menciptakan apa yang terbaik bagi kita
(Agung Kriswantoro)

Jadikanlah kebosanan sebagai sumber kekuatan dan kesabaran
(Agung Kriswantoro)

PERSEMBAHAN

Dengan menyebut asma Allah yang Maha Pengasih lagi Maha Penyayang kupersembahkan Karya sederhana ini kepada:

- *Ibuku Sumarni dan Bapakku Suwadi yang do'a, perhatian, kasih sayang dan perjuangannya selalu menyertai setiap jengkal langkahku serta segala bimbingannya yang kutahu semua itu untuk kebaikan, keberhasilan dan kebahagiaanku.*
- *Mas Prapto, Mas Mamik dan Mbak Lilik sekeluarga yang telah banyak membantu mengantarku menuju kesuksesan.*
- *Keluarga Besar Ngadimin (Bapak dan Ibu Angkatku) yang telah merawat dan memberikan kasih sayangnya semasa kecilku.*
- *Agama yang menuntunku menuju kebaikan dan kebenaran serta sebagai pedoman dalam berpijak dan melangkah.*
- *Matahari, Bulan, Bintang, Bumi dan Langitku yang membuatku tetap bertahan dan terus berjuang dalam mewujudkan inspirasi, imajinasi dan obsesiku.*
- *Ilmu Pengetahuan dan Almamaterku Universitas Jember sebagai tempat berproses dan pembelajaran diri yang aku banggakan.*

DEKLARASI

Karya Tulis Ilmiah (skripsi) ini berisi hasil kerja penelitian mulai bulan September 2003 sampai dengan bulan September 2004, bersama ini saya menyatakan bahwa isi skripsi adalah hasil pekerjaan saya sendiri kecuali disebutkan sumbernya dan skripsi ini belum pernah diajukan pada institusi lain.

Jember, Oktober 2004

Agung Kriswantoro

ABSTRAK

Panyajian Grafik dan Luas Fraktal Koch Snowflake pada Level ke-n, Agung Kriswantoro, 201810101041, Skripsi, Oktober 2004, Jurusan Matematika, Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Jember.

Koch Snowflake adalah suatu fraktal yang pada mulanya dibangun dari kurva *Koch* dibangkitkan pada sisi-sisi segitiga samasisi. Kurva *Koch* dibangun dari sebuah garis lurus yang dibagi menjadi tiga bagian yang sama, kemudian bagian tengahnya dibentuk segitiga samasisi tanpa alas. Penulisan skripsi ini bertujuan untuk mendapatkan algoritma dan program untuk menggambarkan dan menghitung luas fraktal *Koch Snowflake* dengan generator segitiga samasisi, samakaki, dan bujursangkar pada inisiator berupa poligon reguler bersisi-*n*. Dari hasil penelitian ini diperoleh grafik fraktal *Koch Snowflake* dengan generator segitiga samasisi, samakaki dan bujursangkar sampai dengan level ke-9 pada inisiator poligon reguler bersisi 100 beserta perhitungan luasnya.

Kata Kunci: *Kurva Koch, Koch Snowflake, inisiator, generator, algoritma, program.*

PENGESAHAN

Skripsi ini diterima oleh Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Jember, pada :

Hari : KAMIS
Tangggal : 21 OCT 200
Tempat : Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Jember

Tim Penguji,

Ketua
(Dosen Pembimbing Utama),

Drs. Rusli Hidayat, M.Sc.
NIP. 132 048 321

Sekretaris
(Dosen Pembimbing Anggota),

Kosala Dwidja Purnomo, S.Si
NIP. 132 206 019

Anggota I,

Prof. Drs. Kusno, DEA, Ph.D.
NIP.131 592 357

Anggota II,

M. Fatekurohman, S.Si.M.Si.
NIP. 132 210 538

Mengesahkan,
Dekan Fakultas MIPA
Universitas Jember

Ir. Sumadi, M.S.
NIP. 130 368 784

KATA PENGANTAR

Dengan memanjatkan Alhamdulillah, puji syukur kehadirat Allah SWT yang telah melimpahkan rahmat dan hidayahnya sehingga skripsi ini dapat terselesaikan, sebagai syarat untuk memperoleh gelar Sarjana Sains pada Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Jember. Sholawat dan salam kepada junjungan Nabi Muhammad SAW, Rasul Illahi pembawa risalah kebenaran akhir zaman.

Dalam penulisan skripsi ini, penulis telah banyak mendapatkan bantuan dan dorongan secara langsung maupun tidak langsung dari berbagai pihak. Untuk itu penulis ingin menyampaikan terima kasih yang sedalam-dalamnya kepada:

1. Bapak Drs. Rusli Hidayat, M.Sc. selaku Dosen Pembimbing Utama dan Bapak Kosala Dwidja, S.Si. Selaku dosen Pembimbing Anggota yang telah memberikan bimbingan dan arahan sehingga skripsi ini dapat terselesaikan dengan baik.
2. Bapak Prof. Drs. Kusno, DEA. Ph.D. dan Bapak M. Fatekurohman, S.Si. M.Si. serta Bapak Drs. Budi Lestari, Dip. Sc. M.Si. selaku dosen penguji yang telah banyak memberikan kritik, saran, dan masukan sehingga skripsi ini dapat terselesaikan dengan baik.
3. Kedua orang tuaku yang telah membesarkan, mendidik, mencurahkan do'a dengan penuh kasih sayang serta perjuangannya untuk keberhasilanku selama ini.
4. Kakakku Mas Prapto, Mas Mamik dan Mbak Lilik yang memberikan semangat serta koponakanku yang nakal dan lucu-lucu yang menjadi sumber inspirasiku.
5. Elin yang selalu terpaksa menasehatiku selama ini dan telah membangkitkan semangatku untuk mencapai semua inspirasi dan obsesiku.
6. Semua teman Matematika angkatan 2000 terima kasih atas semua bantuannya, saudaraku (Yayan, Martin) yang setia menjadi sahabat perjuanganku dalam suka dan duka. Abdul Kamil yang membantu ide dan buku-bukunya dalam menulis skripsi ini.

7. Mas Jati, Mas Wirid sekeluarga, Mas Sandi dan Mas Yoyok yang telah banyak mengajariku ilmu komputer.
8. Semua dosen di Jurusan Matematika Fakultas MIPA Unej dan Bapak Ibu guru yang telah membantuku dalam beproses dengan Ilmu Pengetahuan.
9. Teman-teman Asrama Mahasiswa Unej (Bang Tatang, Prist, Yus, Mas Bas, Pak Eko, Mbah Dung, Juli, Edi, Guntur, Agung Ma, Arip, Indra, Ikrar, Ucop, Nadib, Doni, Andhi) terima kasih untuk kenangan, proses dan perjuangan yang kita jalani bersama.
10. Teman-teman Natural band (saat SMU) dan Me'Nyes band (saat Kuliah) yang telah menjadi semangat dan inspirasiku.
11. Semua pihak yang telah membantu kelancaran penulisan skripsi ini.

Penulis menyadari sepenuhnya bahwa dalam penyusunan skripsi ini masih jauh dari sempurna, oleh karena itu kritik dan saran yang membangun dari semua pihak sangat diharapkan.

Jember, Oktober 2004 Penulis

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR LAMPIRAN

BAB I
PENDAHULUAN

1.1 Latar Belakang

Ilmu pengetahuan dan teknologi saat ini berkembang sangat pesat terutama setelah ditemukannya komputer. Komputer akan memudahkan manusia untuk melakukan perhitungan matematika atau perhitungan lainnya. Selain itu paket-paket program komputer yang tersedia dapat digunakan untuk menampilkan grafik suatu kurva secara cepat dan mudah.

Salah satu bidang matematika yang berkembang cukup pesat dengan adanya komputer ini adalah geometri fraktal atau lebih dikenal dengan istilah fraktal. Fraktal pertama kali diperkenalkan oleh Benoit Mandelbrot pada tahun 1975. Benoit Mandelbrot merupakan seorang perintis dalam penyelidikan tentang *self-similarity* (kesebangunan diri), yaitu sifat yang sangat penting dalam fraktal. Mandelbrot menggunakan istilah *fractal* (yang berasal dari kata *fractional dimensional*), untuk menunjuk kurva-kurva yang mempunyai sifat *self-similarity*. Sifat *self-similarity* pada prinsipnya identik dengan kesebangunan. Suatu objek dikatakan *self-similarity* jika perubahan ukurannya tidak mengubah bentuknya.

Koch Snowflake yang merupakan salah satu contoh fraktal yang juga dikenal sebagai pulau Koch, pertama kali digambarkan oleh Helge Von Koch, seorang matematikawan Swedia pada tahun 1904. *Koch Snowflake* dibangun dari kurva *Koch* takhingga yang digandakan pada segitiga sama sisi.

Kurva *Koch* ini dibangun dari sebuah segmen garis yang dibagi menjadi tiga bagian yang sama dan bagian tegahnya diubah bentuknya menjadi segitiga sama sisi tanpa alas sehingga membentuk kurva Koch dengan empat segmen garis. Untuk orde berikutnya kurva *Koch* ini dibentuk dengan membagi setiap segmen garis dari kurva *Koch* orde sebelumnya menjadi tiga bagian, dan bagian tengahnya diubah menjadi segitiga sama sisi tanpa alas. Proses ini juga berlaku untuk kurva orde yang lebih tinggi, sehingga untuk menggambarkan kurva ini untuk orde tinggi sulit dilakukan dengan cara manual.

Kamil (2004), telah memformulasikan luas fraktal *Koch Snowflake* dengan generator segitiga samasisi, samakaki, dan bujursangkar pada inisiator segitiga samasisi dan poligon reguler bersisi-*n*. Dengan bantuan komputer, kurva *Koch* dan *Koch Snowflake* dapat dibangun dan dibangkitkan melalui proses iterasi. Sehingga dari yang diuraikan diatas menarik untuk dikaji bagimana cara membangun dan menampilkan gambar kurva *Koch* dan *Koch Snowflake* dengan menggunakan program komputer. Dalam hal ini program yang digunakan adalah Borland Delphi 6.0.

1.2 Perumusan Masalah

Masalah yang akan dibahas dalam skripsi ini adalah bagaimana algoritma dan program untuk menggambarkan kurva *Koch* dan fraktal *Koch Snowflake* serta variasinya yang dibangun pada inisiator berupa poligon reguler bersisi-*n* dengan generator segitiga sama sisi, sama kaki, dan bujur sangkar berserta perhitungan luasnya.

1.3 Tujuan

Tujuan dari penelitian ini adalah untuk mendapatkan algoritma dan program untuk menggambarkan kurva *Koch* beserta variasinya dan fraktal *Koch Snowflake* dengan generator segitiga samasisi, samakaki, dan bujursangkar pada inisiator berupa poligon reguler bersisi-*n* sehingga didapatkan bentuk fraktal *Koch Snowflake* yang lebih menarik. Selain itu juga untuk mendapatkan algoritma dan program menghitung luas fraktal *Koch Snowflake* beserta variasinya (*Koch snowflake* dengan luas bertambah dan luas berkurang).

1.4 Manfaat

Manfaat yang didapatkan dalam penelitian ini adalah memudahkan kita untuk dapat menggambarkan objek-objek di alam yang memiliki kemiripan sifat dengan *Koch Snowflake*. Lebih lanjut konsep fraktal dapat digunakan untuk mendesain suatu pola yang lebih artistik.

BAB II
TINJAUAN PUSTAKA

2.1 Fraktal

2.1.1 Pengertian Fraktal

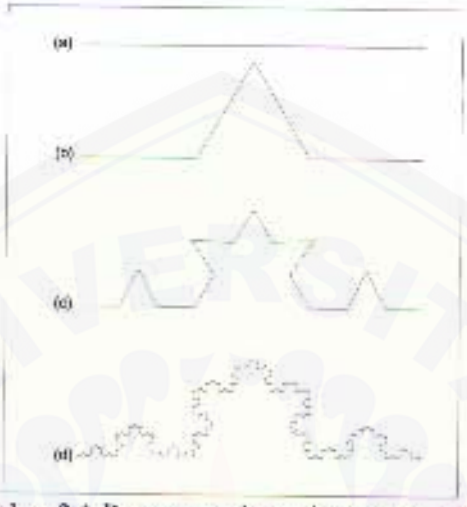
Self-similarity merupakan suatu keadaan objek yang dibangun secara berulang dengan mengganti suatu gambar dengan yang sebangun, tetapi berukuran lebih kecil dari asalnya. Ini berarti bahwa sekecil apapun gambar tersebut, apabila diperbesar tetap mempunyai bentuk yang sama.

Kata fraktal berasal dari bahasa latin *fractus* (memecahkan/menguraikan) dan kata kerja dalam bahasa latin *frangere*, yang berarti meretakkan atau membagi menjadi kepingan-kepingan. Menurut Benoit Mandelbrot fraktal merupakan bentuk geometri yang tampak kasar atau berupa sebuah pecahan yang dapat dibagi lagi pada bagian-bagiannya, masing-masing bagian merupakan (setidak-tidaknya) bentuk turunan yang diperbanyak dari ukuran keseluruhannya. Sedangkan definisi fraktal secara matematik sebagai suatu himpunan titik-titik yang memiliki dimensi fraktal melebihi dimensi topologinya (Bourke, 1991).

2.1.2 Fraktal *Koch Snowflake*

Salah satu contoh kurva yang memiliki sifat *self-similarity* adalah kurva *Koch* yang merupakan generator dari pembentukan fractal *Koch Snowflake*. Kurva *Koch* didasarkan pada garis-garis yang mempunyai arah tertentu dan dihubungkan satu sama lain, sehingga terbentuk suatu garis yang sangat panjang pada suatu daerah yang terbatas. Langkah-langkah pembentukan kurva *Koch* dimulai dengan sebuah garis lurus (Gambar 2.1.a). Untuk membentuk kurva *Koch* orde satu, yaitu K_1, garis tersebut dibagi menjadi tiga bagian, dan bagian tengah diubah menjadi segitiga samasisi tanpa alas, sehingga membentuk bangun dengan empat buah segmen garis (Gambar 2.1.b). Untuk membentuk kurva *Koch* orde dua K_2, dibentuk dengan membagi setiap segmen garis dari kurva *Koch* orde satu menjadi tiga bagian, dan bagian tengahnya diubah menjadi segitiga samasisi (gambar 2.1.c). Dengan cara yang sama, kurva *Koch* untuk orde yang lebih tinggi bisa

diperoleh dari kurva *Koch* sebelumnya. Dengan kata lain, untuk memperoleh kurva *Koch* orde-i, setiap segmen yang ada pada kurva *Koch* orde $i-1$ dibagi menjadi tiga bagian sama panjang, dan bagian tengahnya diubah menjadi bangun dengan sisi yang sama tanpa alas.



Gambar 2.1 Proses pembentukan Kurva *Koch*

Fraktal *Koch Snowflake* (Gambar 2.2) dibangun dari kurva *Koch* yang dibangkitkan pada sisi-sisi segitiga samasisi, didasarkan pada garis-garis yang mempunyai arah tertentu dan dihubungkan satu sama lain. Variasi *Koch Snowflake* dapat dilakukan dengan cara mengkombinasikan antara inisiator dan generatornya (Kamil, 2004).

Gambar 2.2 *Koch Snowflake*

2.1.3 Variasi Bentuk *Koch Snowflake*

a. Variasi *Koch Snowflake* dengan luas bertambah

Pada dasarnya variasi dari *Koch Snowflake* ini dapat dilakukan dengan cara membangkitkan berbagai generator pada sebarang segmen garis. Akan tetapi, yang akan dibahas adalah yang menghasilkan segmen garis yang kongruen. Pengembangan variasi *Koch Snowflake* dengan luas bertambah dilakukan dengan cara membangkitkan generatornya dengan arah yang melebar keluar pada sisi-sisi inisiatornya.

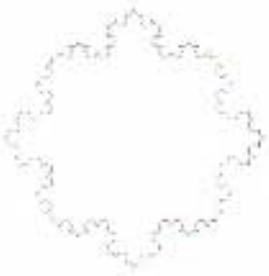
Beberapa gambar di bawah ini mengilustrasikan berbagai variasi *Koch Snowflake* dengan generator segitiga dan bujursangkar yang dibangkitkan pada inisiator segitiga samasisi (Gambar 2.3), bujursangkar (Gambar 2.4) dan poligon beraturan (Gambar 2.5).

a. Generator Segitiga samasisi

b. Generator segitiga samakaki

c. Generator bujursangkar

Gambar 2.3 *Koch Snowflake* dengan inisiator segitiga samasisi



a. Generator segitiga samasisi

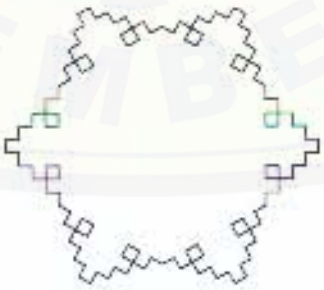
b. Generator segitiga samakaki

c. Generator bujursangkar

Gambar 2.4 *Koch Snowflake* dengan inisiator bujursangkar

a. Generator segitiga samasisi

b. Generator segitiga samakaki



c. Generator bujursangkar

Gambar 2.5 *Koch Snowflake* dengan inisiator poligon beraturan bersisi 6

b. Variasi *Koch Snowflake* dengan luas berkurang

Variasi *Koch Snowflake* ini dilakukan dengan cara mengubah arah melebarnya kurva *Koch* ke dalam atau pusat inisiatornya. Variasi yang dihasilkan ini dikenal sebagai *Anti Koch Snowflake*. Berikut adalah beberapa *Anti Koch Snowflake* yang dibangkitkan dari generator segitiga samasisi, samakaki dan bujursangkar pada beberapa poligon kongruen bersisi-*n*.

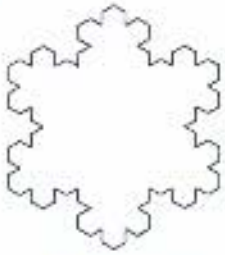
a. Generator segitiga samasisi b. Generator segitiga samakaki

Gambar 2.6 *Koch Snowflake* dengan inisiator segitiga samasisi

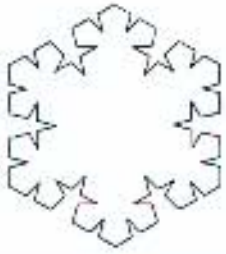
a. Generator segitiga samasisi b. Generator segitiga samakaki

c. Generator bujursangkar

Gambar 2.7 *Koch Snowflake* dengan inisiator bujursangkar



a. Generator segitiga samasisi



b. Generator segitiga samakaki

c. Generator bujursangkar

Gambar 2.8 *Koch Snowflake* dengan inisiator poligon bersisi 6

c. Variasi *Koch Snowflake* dengan mempertahankan luas

Variasi *Koch Snowflake* ini merupakan perpaduan antara *Koch Snowflake* dengan luas bertambah dan *Koch Snowflake* dengan luas berkurang. Dengan adanya kombinasi pada generatornya akan dihasilkan bentuk *Koch Snowflake* dengan tampilan lebih menarik dan tetap mempertahankan luas, yaitu sebesar luas inisiatornya. Generator yang digunakan merupakan variasi bentuk bujursangkar dengan arah kurva yang melebar keluar dan kedalam sisi-sisi inisiator secara kongruen, dikenal sebagai kurva *Koch* kuadratik (Paul Addison, 1997) (Gambar 2.9) dan segitiga (Gambar 2.10).

a. Generator kuadratik

b. *Koch Snowflake* pada inisiator bujursangkar

Gambar 2.9 *Koch Snowflake* Kuadratik

(a) Generator segitiga

(b) inisiator bujursangkar

(c) inisiator segitiga

Gambar 2.10 *Koch Snowflake* dengan generator segitiga yang mempertahankan luas

2.2 Luas *Koch Snowflake*

Kamil (2004) telah memformulasikan luas fraktal *Koch Snowflake*. Metode yang digunakan untuk menentukan luas *Koch Snowflake* adalah dengan membangun deret geometri yang merupakan penjumlahan luas semua bidang yang dibangkitkan oleh generator pada masing-masing tingkatan iterasi kurva *Koch*. Langkah pertama, menentukan luas inisiator yang berbentuk poligon beraturan bersisi-n dengan panjang sisi sama dengan L, misal A. Kedua, ditentukan penambahan atau pengurangan luas akibat dibangkitkannya suatu generator. Misalkan penambahan atau pengurangan luas pada setiap sisi adalah T. Maka, luas Koch Snowflake dengan inisiator poligon bersisi-n adalah $A \pm nT$.

2.2.1 *Koch Snowflake* dengan Luas Bertambah

Luas *Koch Snowflake* dengan panjang sisi inisiatornya L, untuk iterasi $i = 1,2,3\ldots$ pada:

1. Generator segitiga samasisi yang dibangkitkan pada inisiator poligon beraturan bersisi-n adalah:

$$\text{Luas} = \frac{nL^2}{4}\frac{\cos\left(\frac{\pi}{n}\right)}{\sin\left(\frac{\pi}{n}\right)} + n\frac{\sqrt{3}L^2}{36}\sum_{i=1}^{\infty}\left(\frac{4}{9}\right)^{i-1} \qquad (2.1)$$

2. Generator segitiga samakaki yang dibangkitkan pada inisiator Poligon beraturan bersisi-n adalah:

$$\text{Luas} = \frac{nL^2}{4}\left(\frac{\cos\left(\frac{\pi}{n}\right)}{\sin\left(\frac{\pi}{n}\right)} + (1-2p)(4p-1)^{1/2}\sum_{i=1}^{\infty}\left(4p^2\right)^{i(i-1)}\right) \qquad (2.2)$$

atau

$$\text{Luas} = \frac{nL^2}{4}\left(\frac{\cos\left(\frac{\pi}{n}\right)}{\sin\left(\frac{\pi}{n}\right)} + q(1-2q)^{1/2}\sum_{i=1}^{\infty}(1-q)^{i(i-1)}\right) \qquad (2.3)$$

Pada generator segitiga samakaki, diambil beberapa asumsi, yaitu pada setiap iterasi berlaku :

a. Perbandingan p dan q adalah sama.

b. $2p > q$

dengan p dan q masing masing adalah ukuran sisi miring dan alas dari segitiga samakaki pada generator, dalam hal ini diasumsikan juga, untuk setiap iterasi berlaku $2p + q$ sama dengan panjang garis yang dipartisi (Gambar 2.11), dan berlaku juga bahwa $2p + q = 1$ untuk setiap iterasi.

Gambar 2.11 Ukuran sisi pada setiap iterasi

3. Generator bujursangkar yang dibangkitkan pada inisiator poligon beraturan bersisi-n adalah:

$$\text{Luas} = nL^2\left(\frac{1}{4}\frac{\cos\left(\frac{\pi}{n}\right)}{\sin\left(\frac{\pi}{n}\right)} + \frac{1}{9}\sum_{i=1}^{\infty}\left(\frac{5}{9}\right)^{i-1}\right) \qquad (2.4)$$

2.2.2 *Koch Snowflake* dengan Luas Berkurang (Anti *Koch Snowflake*)

Formulasi perhitungan luas Anti *Koch Snowflake* untuk iterasi $i = 1,2,3\ldots$ pada:

1. Generator segitiga samasisi yang dibangkitkan pada inisiator poligon beraturan bersisi-n adalah:

$$\text{Luas} = \frac{nL^2}{4}\frac{\cos\left(\frac{\pi}{n}\right)}{\sin\left(\frac{\pi}{n}\right)} - n\frac{\sqrt{3}L^2}{36}\sum_{i=1}^{\infty}\left(\frac{4}{9}\right)^{i-1} \qquad (2.5)$$

2. Generator segitiga samakaki yang dibangkitkan pada inisiator Poligon beraturan bersisi-n adalah:

$$\text{Luas} = \frac{nL^2}{4}\left(\frac{\cos\left(\frac{\pi}{n}\right)}{\sin\left(\frac{\pi}{n}\right)} - (1-2p)(4p-1)^{1/2}\sum_{i=1}^{\infty}(4p^2)^{(i-1)}\right) \qquad (2.6)$$

atau

$$\text{Luas} = \frac{nL^2}{4}\left(\frac{\cos\left(\frac{\pi}{n}\right)}{\sin\left(\frac{\pi}{n}\right)} - q(1-2q)^{1/2}\sum_{i=1}^{\infty}(1-q)^{2(i-1)}\right) \qquad (2.7)$$

3. Generator bujursangkar yang dibangkitkan pada inisiator poligon beraturan bersisi-n adalah:

$$\text{Luas} = nL^2\left(\frac{1}{4}\frac{\cos\left(\frac{\pi}{n}\right)}{\sin\left(\frac{\pi}{n}\right)} - \frac{1}{9}\sum_{i=1}^{\infty}\left(\frac{5}{9}\right)^{(i-1)}\right) \qquad (2.8)$$

2.3 Algoritma dan Pemrograman

2.3.1 Pengertian Algoritma

Algoritma adalah urutan langkah instruksi logis yang disusun secara sistematis untuk menyelesaikan suatu masalah (Rinaldi, 1999). Rancangan yang baik untuk suatu algoritma adalah menguraikan prosedur dalam beberapa subprosedur, dari subprosedur ini diuraikan lagi menjadi sub-sub prosedur dan seterusnya. Metode ini disebut rancangan algoritma terstruktur yang memberikan kemudahan pemahaman logika algoritma.

2.3.2 Pemrograman Komputer

Program komputer adalah suatu urutan instruksi untuk dilaksanakan komputer agar hasil tertentu dapat diperoleh. Jadi program komputer merupakan perwujudan atau implementasi dari algoritma yang ditulis dalam bahasa pemrograman tertentu sehingga dapat dilaksanakan oleh komputer (Rinaldi, 1999).

2.4 Delphi

Borland Delphi atau sering dikenal dengan Delphi adalah paket bahasa pemrograman yang bekerja dalam sistem operasi windows. Delphi dapat juga digunakan untuk berbagai aplikasi seperti aplikasi untuk mengolah teks, grafik, angka, database dan aplikasi web.

Delphi menyediakan dua kelompok fasilitas pemrograman, yaitu object dan bahasa pemrograman. Object adalah suatu komponen yang mempunyai bentuk fisik dan dapat dilihat (visual), sedangkan bahasa pemrograman yang digunakan dalam delphi adalah struktur bahasa pemrograman Pascal. Gabungan dari object dan bahasa pemrograman ini sering disebut dengan bahasa pemrograman berorientasi object atau *Object Oriented Programing* (OOP) (Alam, 2001).

2.5 Kontrol Program, Function dan Procedure

2.5.1 Kontrol Program

Delphi mempunyai kontrol program yang digunakan untuk mengatur jalannya program satu per satu, karena dalam satu program terdapat beberapa pernyataan yang dipisahkan dengan tanda titik koma. Berikut ini adalah beberapa kontrol program yang digunakan dalam Delphi:

1. Pengulangan

   Kontrol program pengulangan ini ada tiga macam yaitu:

   a. Pengulangan **`While ... Do`**

      Bentuk penulisannya adalah :

      ```
      While <kondisi> do
      <Blok pernyataan>;
      ```

      Struktur pengulangan ini digunakan untuk melakukan perulangan satu pernyataan atau satu blok pernyataan, selama kondisi bernilai benar.

   b. Pengulangan **`Repeat ... Until`**

      Bentuk penulisannya adalah :

      ```
      Repeat
      <Blok pernyataan>;
      until <Kondisi>
      ```

Struktur pengulangan ini menekankan pada kondisi berhenti, artinya pernyataan akan dilaksanakan berulangkali dan hanya akan berhenti jika kondisi terpenuhi (bernilai benar).

c. **Pengulangan For...do**

Struktur ini digunakan untuk menghasilkan perulangan sejumlah kali tanpa penggunaan kondisi apapun. Bentuk umumnya ada dua macam yaitu: menaik (*ascending*) dan menurun (*descending*).

Bentuk penulisannya untuk menaik adalah:

```
for nilai awal to nilai akhir do <pernyataan>;
```

sedangkan yang menurun adalah sebagai berikut

```
for nilai akhir downto nilai awal do <pernyataan>;
```

2. Pencabangan Bersyarat **If ... Then ... Else**

Pencabangan bersyarat ini digunakan untuk mencabang ke pilihan tertentu berdasar pengujian suatu nilai logika. Bentuk penulisannya seperti berikut ini,

```
If  <kondisi> Then <pernyataan>;
```

atau

```
If <kondisi> Then <pernyataan.1> Else <Pernyataan.2>;
```

Pernyataan **If** yang pertama akan menguji <kondisi> disebelah kanannya, jika <kondisi> bernilai benar maka <pernyataan> akan dilaksanakan, jika salah maka akan melanjutkan ke pernyataan berikutnya. Bentuk **If** yang kedua akan menguji <kondisi> disebelah kanannya, jika <kondisi> bernilai benar maka <pernyataan.1> akan dilaksanakan, jika <kondisi> bernilai salah maka <pernyataan.2> akan dilaksanakan .

3. Pencabangan **Try ... Finally**

Pencabangan **Try ... Finally** dipakai untuk pencabangan yang memiliki kemungkinan kesalahan operasi atau pelaksanaan program. Bentuk penulisannya adalah seperti berikut ini:

```
Try
    <daftar pernyataan.1>;
Finally
```

```
    <daftar pernyataan.2>;
end;
```

Pertama kali program akan menjalankan pernyataan yang ada dibawah pernyataan **Try**. Jika tidak ada kesalahan program akan melaksanakan sampai pernyataan sesudah pernyataan **Finally**. Jika terdapat kesalahan maka hanya akan melaksanakan pernyataan yang ada setelah pernyataan **Finally**.

2.5.2 Function dan Procedure

Function (fungsi) dan procedure adalah suatu rutin (sub program) yang biasanya dipakai sebagai alat untuk melakukan tugas tertentu dan atau mendapatkan nilai tertentu. Function sebenarnya hampir sama dengan procedure, yang membedakan adalah function mempunyai nilai kembalian (*return value*), sedangkan procedure tidak mempunyai nilai kembalian. Pada penulisannya procedure ditulis sebagai suatu penyataan yang berdiri sendiri, berikut ini adalah contoh penulisan procedure dalam program.

```
Procedure TForm1. FormCreate(Sender: TObject);
Var
  <variabel yang digunakan>;
Begin
  <blok pernyataan>;
End;
```

Dan berikut ini adalah beberapa function yang disediakan dalam Delphi:

1. **StrToFloat**

   Function ini digunakan untuk mendapatkan nilai bertipe extended (salah satu tipe real) dari suatu data string. Bentuk penulisanya seperti berikut,

   ```
   StrToFloat (S:String);
   ```

   S adalah data tipe string yang akan diambil nilai extended-nya.

2. **FloatToStr**

   Function ini digunakan untuk mendapatkan nilai bertipe string dari suatu data bertipe extended. Bentuk penulisanya seperti berikut,

   ```
   FloatToStr (S:Extended);
   ```

   S adalah data tipe string yang akan diambil nilai Integer-nya

3. **StrToInt**

Function ini digunakan untuk mendapatkan nilai bertipe integer (bilangan bulat) dari suatu data string. Bentuk penulisanya seperti berikut,

```
StrToInt (S:String);
```

S adalah data tipe string yang akan diambil nilai Integer-nya.

4. **IntToStr**

Function ini digunakan untuk mendapatkan nilai bertipe string dari suatu data bernilai real. Bentuk penulisanya seperti berikut,

```
IntToStr (S:integer);
```

S adalah data tipe integer yang akan diambil nilai string-nya.

KESIMPULAN DAN SARAN

4.1 Kesimpulan

Dari pembahasan pada bab III dapat diambil kesimpulan bahwa dengan menggunakan bantuan program Borland Delphi 6.0 dapat digambarkan :

1. Grafik fraktal *Koch Snowflake* dan *Anti Koch Snowflake* dengan generator segitiga samasisi sampai level 9 pada generator poligon reguler bersisi sampai dengan 100.
2. Grafik fraktal *Koch Snowflake* dan *Anti Koch Snowflake* dengan generator segitiga samakaki sampai level 9 pada generator poligon reguler bersisi sampai dengan 100.
3. Grafik fraktal *Koch Snowflake* dan *Anti Koch Snowflake* dengan generator bujursangkar sampai level 9 pada generator poligon reguler bersisi sampai dengan 100.

Untuk penyajian grafik dengan level yang lebih tinggi dari 9 memerlukan waktu yang cukup lama dan juga perbedaan tampilan level ke 9 dengan level 10 dan seterusnya sangat kecil sekali disebabkan segmen garis yang dibentuk sangat kecil. Selain itu juga didapatkan perhitungan luas untuk tiap-tiap fraktal yang disajikan.

4.2 Saran

Dalam menyajikan grafik fraktal *Koch Snowflake* dengan menggunakan Borland Delphi 6.0, masih terdapat kekurangsempurnaan diantaranya adalah untuk menampilkan grafik fraktal *Koch Snowflake* pada level yang tinggi sehingga diperlukan adanya batasan untuk input levelnya. Tentu saja tidak mungkin mendapatkan hasil yang sempurna. Namun dengan demikian masih dapat digunakan untuk menyajikan grafik fraktal koch snowflake dengan generator segitiga samasisi, samakaki dan bujursangkar sampai level ke-9. Selain itu juga masih terbuka untuk mengembangkan program dengan beberapa modifikasi yang nantinya dapat menyajikan grafik fraktal dengan generator yang berbeda dan level yang lebih tinggi.

DAFTAR PUSTAKA

Addison P. S., 1997, ***Fractal and caos an Illustrated Course***, Institute of Publishing, London.

Agus M., Alam J., 2001. ***Belajar Sendiri Borland Delphi 6.0***. PT Elex Media Komputindo Kelompok Gramedia, Jakarta.

Bourke P., 1991, ***An Introduction to Fractals***. http://www.astronomy.swin.edu.au/~pbourke/fractals/francito.htm

Falconer K., 1990, ***Fractal Geometry Mathematical Fondations and Applications***. Jhon Willey and Sons, Chicester, New York, Brisbane, Toronto, Singapore.

Jogiyanto, 1989, ***Teori dan Aplikasi Program Komputer Bahasa Pascal***, Andi Offset, Yogyakarta.

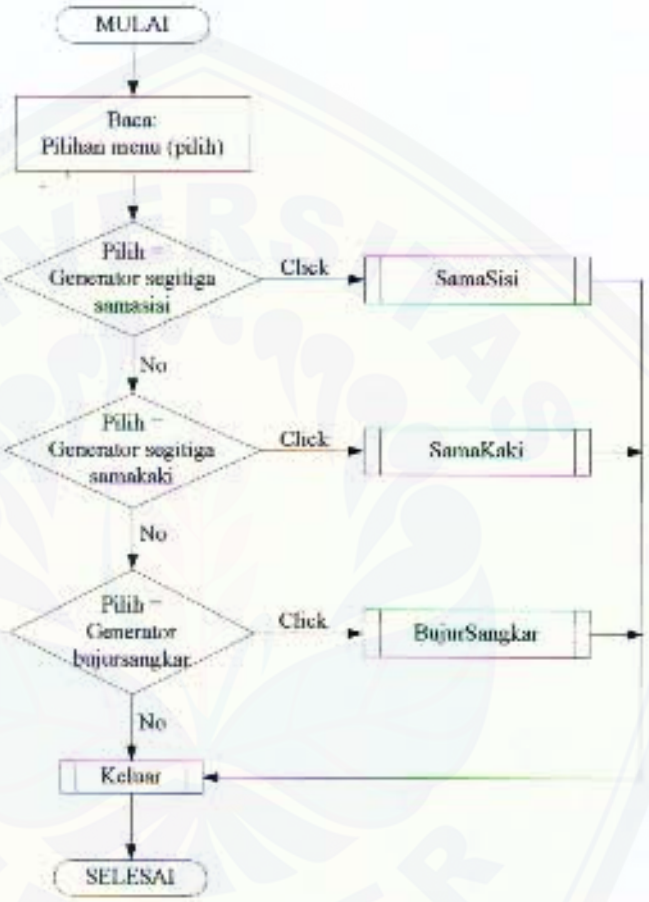
Kamil A., 2004, ***Penentuan Luas Fraktal Koch Snowflake***, Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Jember, Jember.

Munir R., 1999, ***Algoritma dan Pemrograman Bahasa Pascal dan C***, CV. Informatika, Bandung.

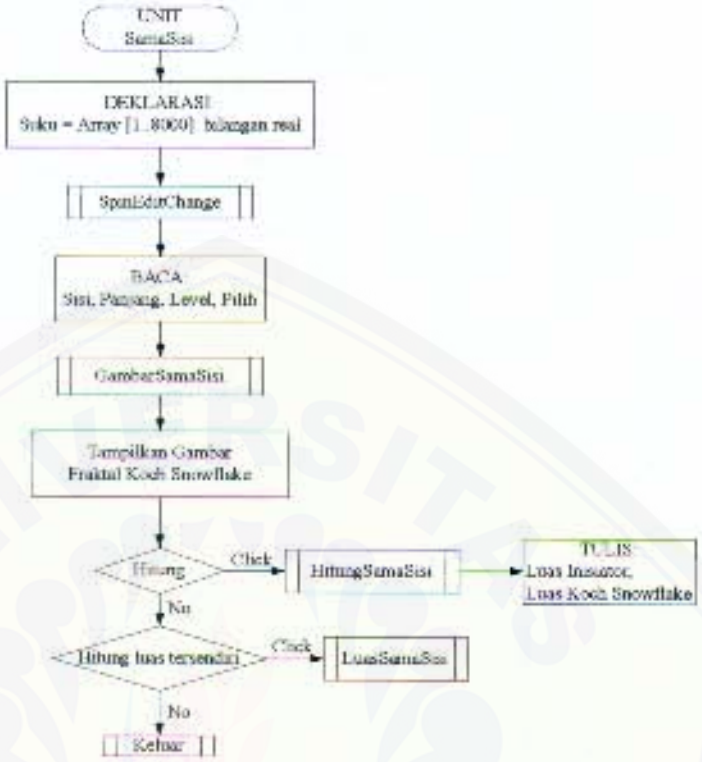
Lampiran 1:

Bagan Alir Program Penyajian Grafik dan Luas Fraktal *Koch Snowflake* pada Level ke-*n* dengan menggunakan Borland Delphi 6.0.

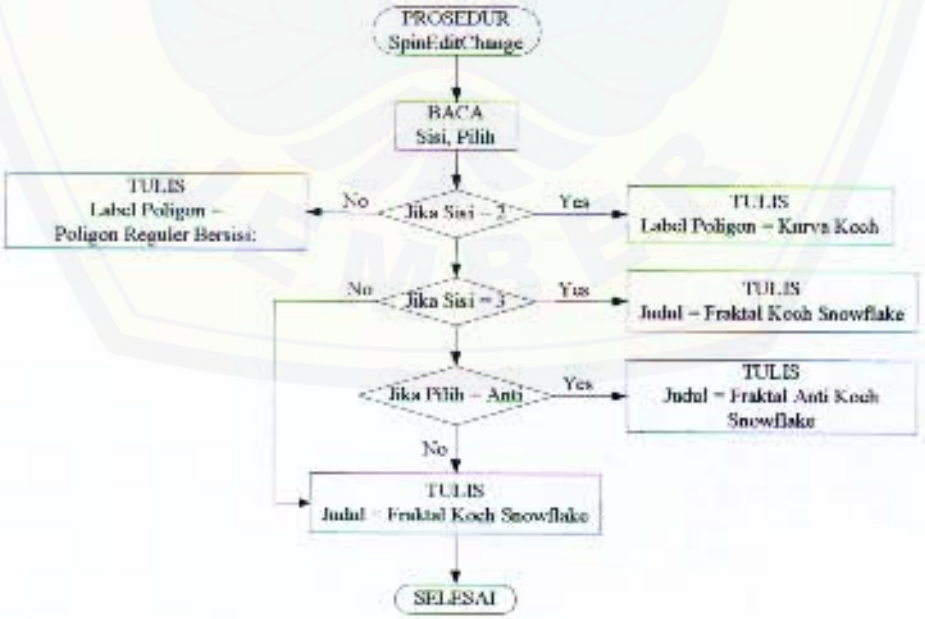
Lampiran 1.1 Bagan Alir Program Menu



Lampiran 1.2 Bagan Alir Unit SamaSisi

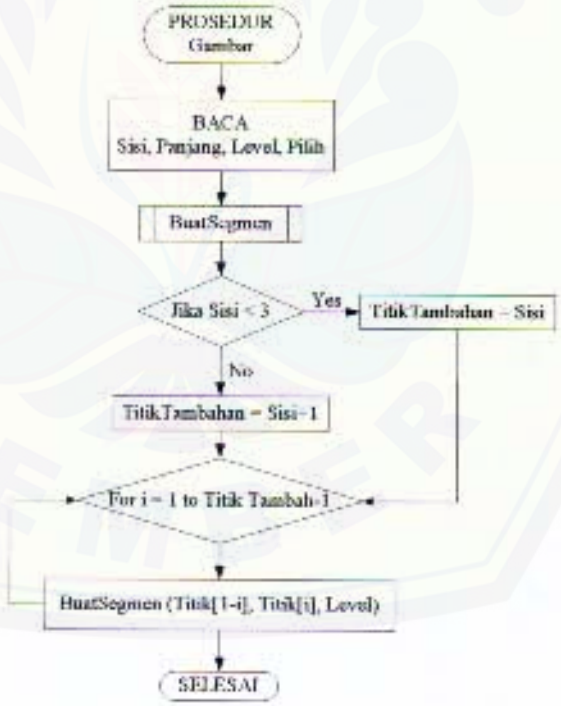


Lampiran 1.3 Bagan Alir Prosedur SpinEditChange

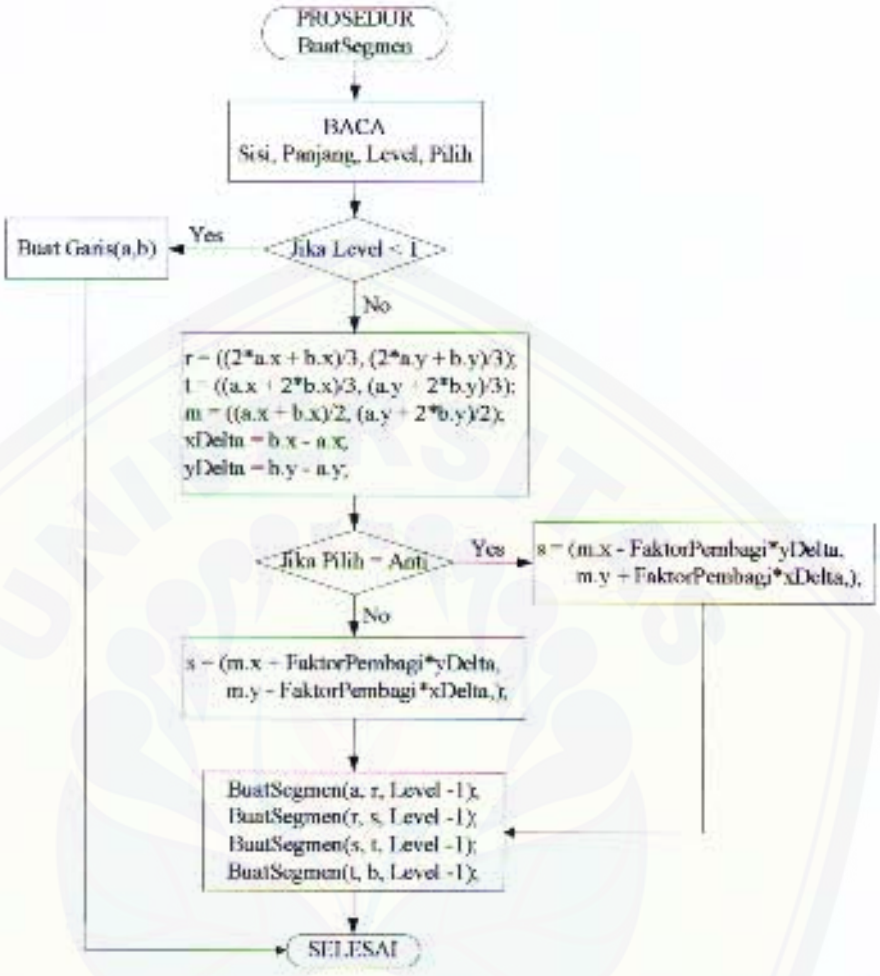


Lampiran 1.4 Bagan Alir Prosedur GambarSamaSisi

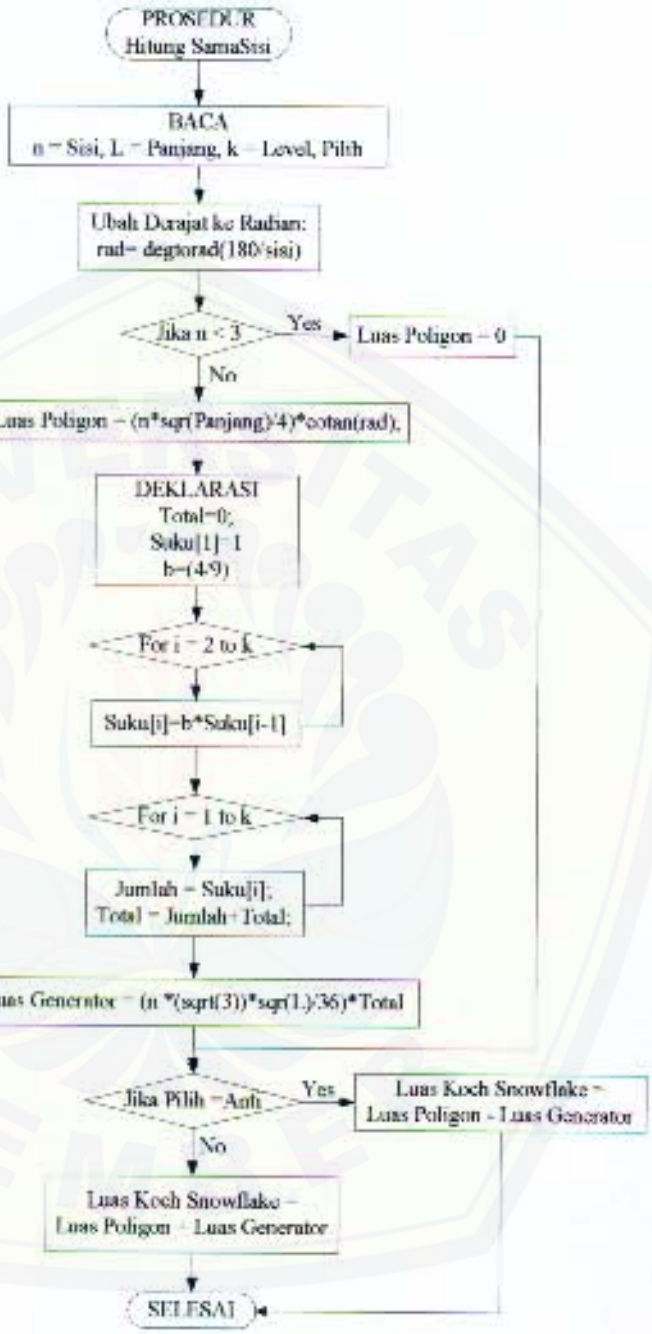
Lampiran 1.5 Bagan Alir Prosedur Gambar



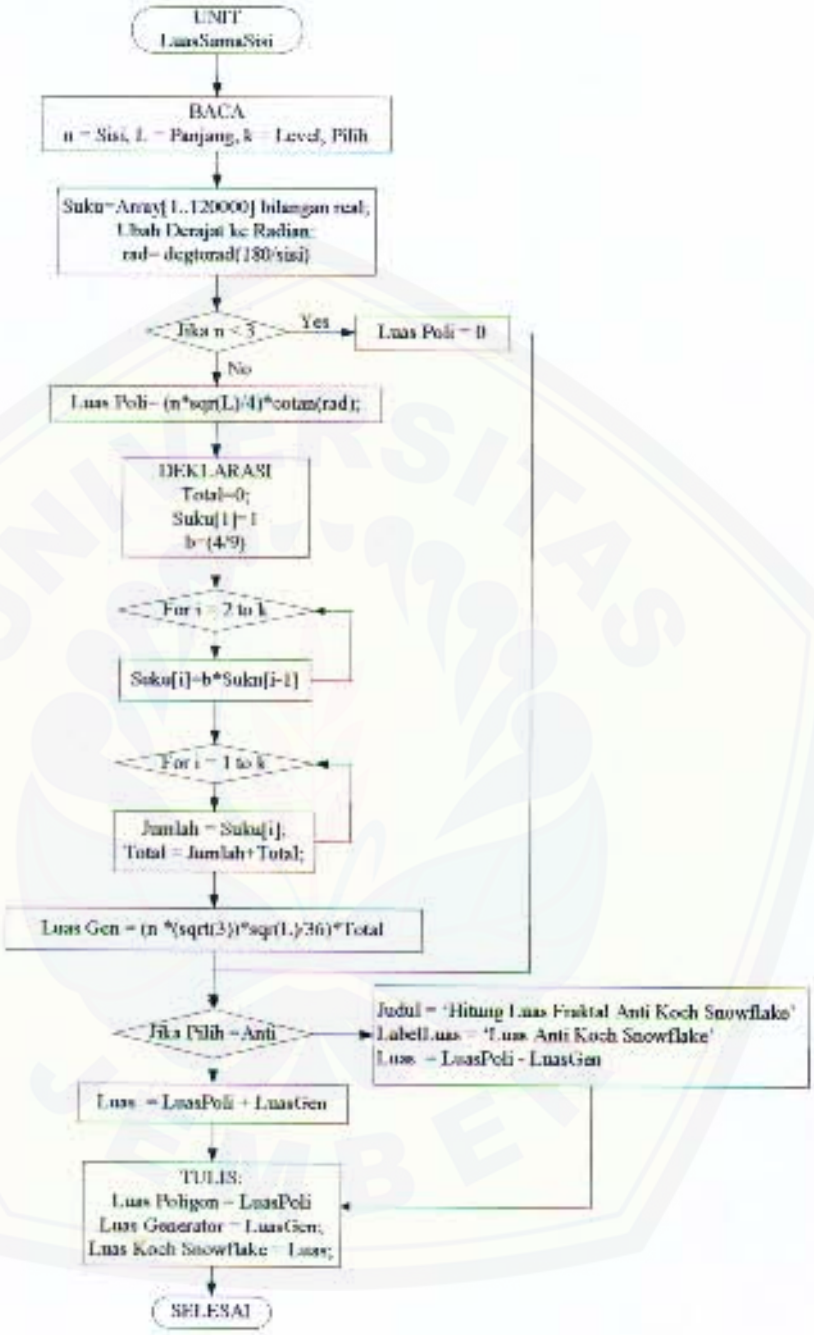
Lampiran 1.6 Bagan Alir Prosedur BuatSegmen



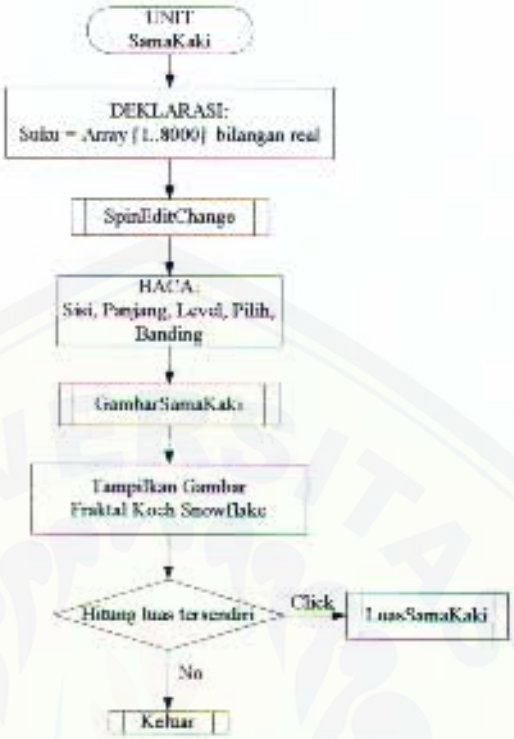
Lampiran 1.7 Bagan Alir Prosedur HitungSamaSisi



Lampiran 1.8 Bagan Alir LuasSamaSisi

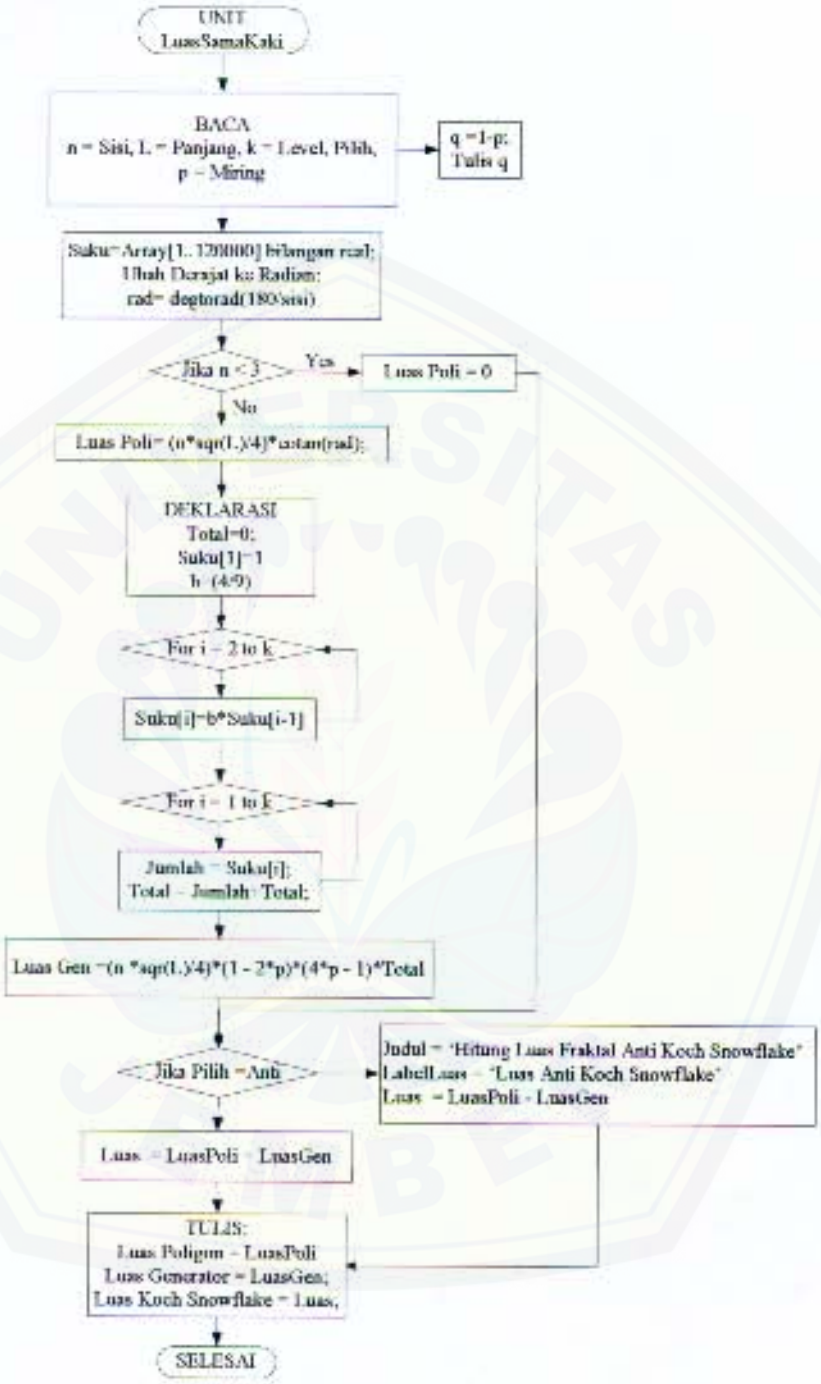


Lampiran 1.9 Bagan Alir Unit SamaKaki

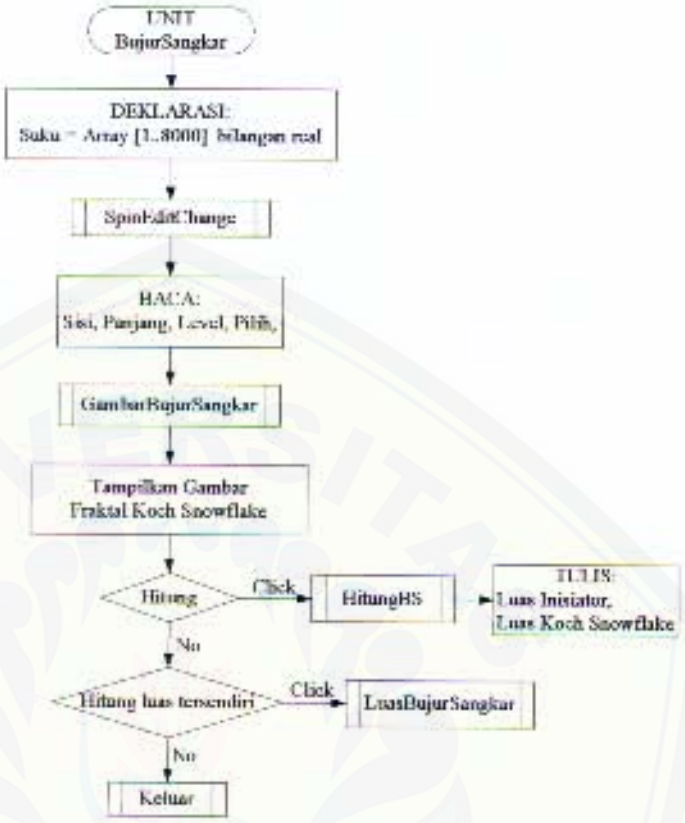


Lampiran 1.10 Bagan Alir GambarSamaKaki

Lampiran 1.11 Bagan Alir Unit LuasSamaKaki

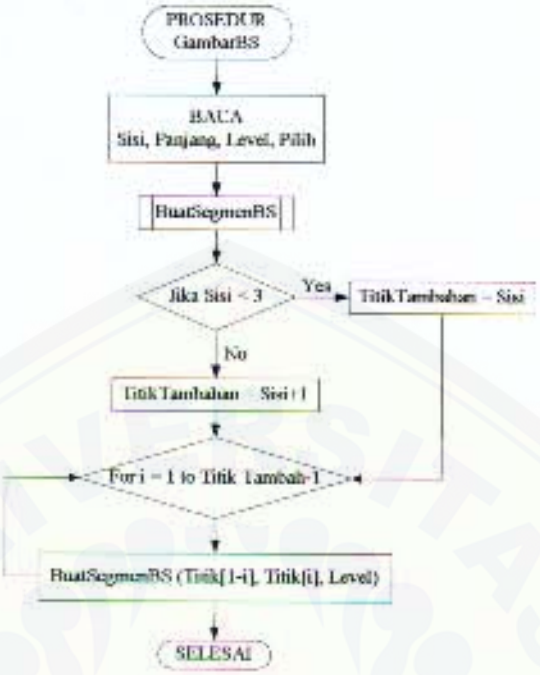


Lampiran 1.12 Bagan Alir Unit BujurSangkar

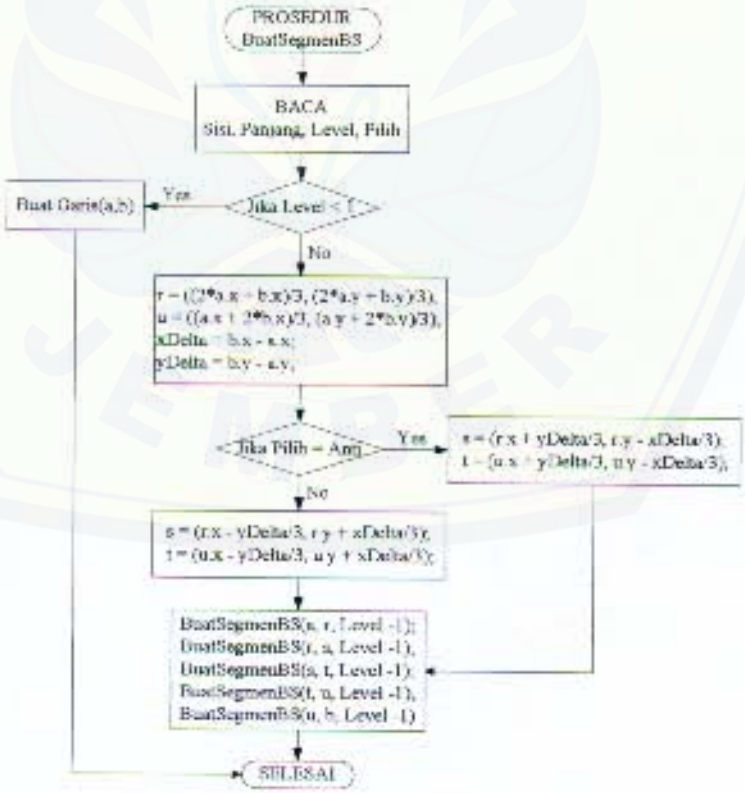


Lampiran 1.13 Bagan Alir Prosedur GambarBujurSangkar

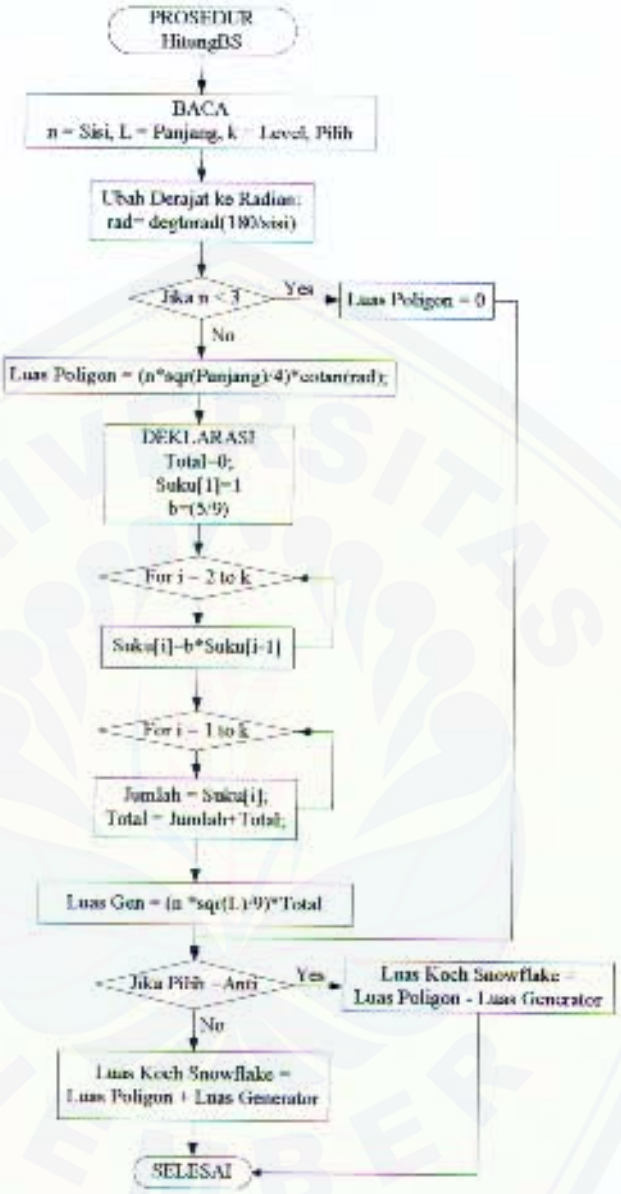
Lampiran 1.14 Bagan Alir Prosedur GambarBS



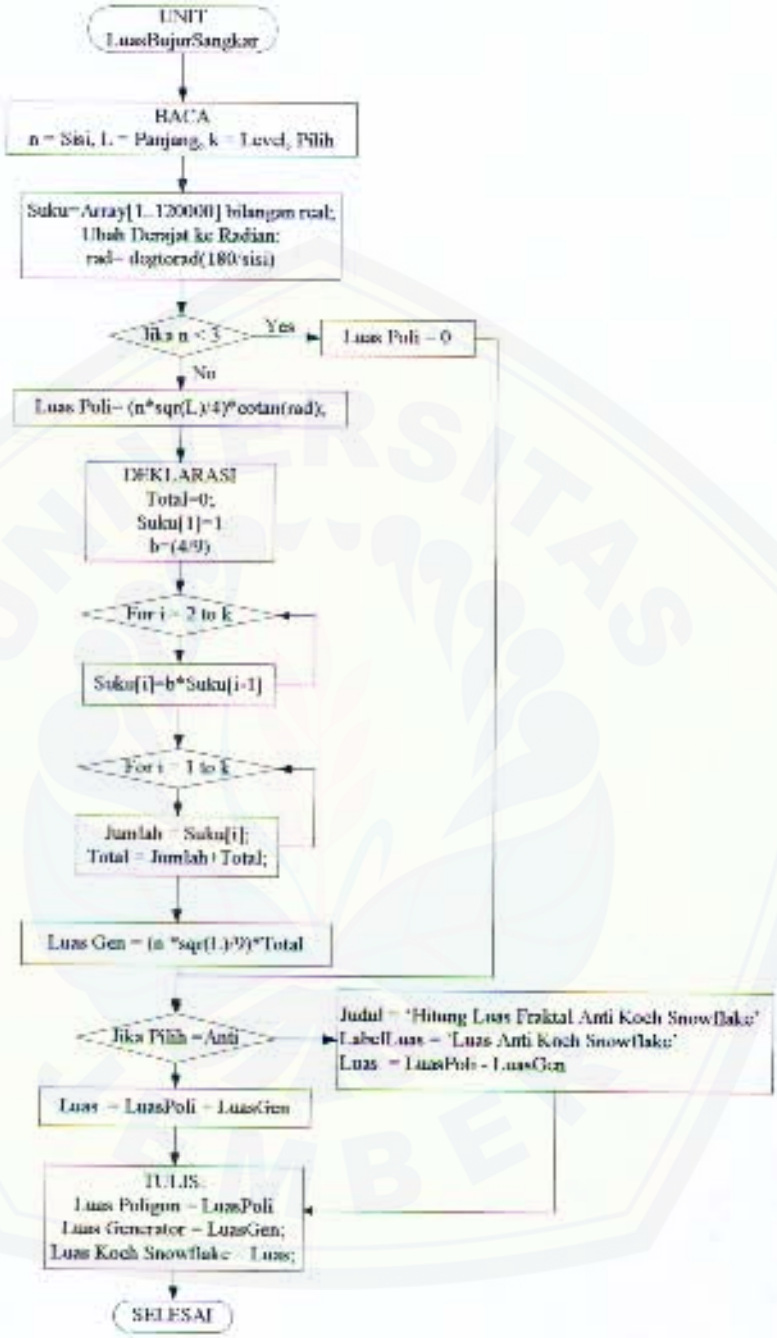
Lampiran 1.15 Bagan Alir Prosedur BuatSegmenBS



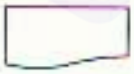
Lampiran 1.16 Bagan Alir Prosedur HitungBS



Lampiran 1.17 Bagan Alir Unit LuasBujurSangkar



Lampiran 1.18 Tabel Keterangan Simbol Bagan Alir

Simbol	Keterangan
	Simbol untuk menandakan awal dan akhir dari suatu program
	Simbol untuk menuliskan data/input maupun hasil/output
	Simbol memproses perintah/pernyataan pada program
	Simbol untuk eksekusi dengan menggunakan unit subprogram
	Simbol untuk melakukan tindakan pilihan berdasarkan terpenuhi atau tidaknya suatu syarat yang ditetapkan
	Simbol untuk menuliskan hasil proses eksekusi bagi dokumentasi
	Simbol penghubung, untuk menandai garis alir ke atau dari bagian tertentu bagan alir yang ditulis terpisah karena tidak cukupnya tempat penulisan pada satu bagian yang ditulis di halaman yang berbeda
	Simbol menunjukkan arah alir proses

Lampiran 2:

List Program Meyajikan Grafik Fraktal Koch Snowflake dengan Generator Segitiga Samasisi, Samakaki dan Bujursangkar Beserta Perhitungan Luasnya.

Lampiran 2.1 List Program Menu

```
// Program Menu untuk menampilkan fraktal Koch Snowflake dengan generator
// Segitiga Samasisi, Segitiga Samakaki, dan Bujursangkar

unit Menu;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls,
  Forms, Dialogs, StdCtrls, Buttons, Menus;

type
  TFormMenu = class(TForm)
    BitBtnSS: TBitBtn;
    BitBtnSK: TBitBtn;
    BitBtnBS: TBitBtn;
    Label1: TLabel;
    BitBtnEXIT: TBitBtn;
    MainMenu1: TMainMenu;
    File1: TMenuItem;
    Exit1: TMenuItem;
    About1: TMenuItem;
    Generator1: TMenuItem;
    SegitigaSamasisi1: TMenuItem;
    Bujursangkar1: TMenuItem;
    SegitigaSamakaki1: TMenuItem;
    Help1: TMenuItem;
    Procedure BitBtnSSClick(Sender: TObject);
    Procedure BitBtnBSClick(Sender: TObject);
    Procedure BitBtnEXITClick(Sender: TObject);
    Procedure BitBtnSKClick(Sender: TObject);
    Procedure About1Click(Sender: TObject);
    Procedure SegitigaSamakaki1Click(Sender: TObject);
    Procedure SegitigaSamasisi1Click(Sender: TObject);
    Procedure Bujursangkar1Click(Sender: TObject);
    Procedure Exit1Click(Sender: TObject);

  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  FormMenu: TFormMenu;

implementation
{$R *.dfm}

uses SamaSisi,SamaKaki,
```

```
    BujurSangkar, TentangKU;
Procedure TFormMenu.BitBtnSSClick(Sender: TObject);
Begin
  FormSamaSisi.ShowModal;
End;

Procedure TFormMenu.BitBtnSKClick(Sender: TObject);
Begin
  FormSamaKaki.ShowModal;
End;

Procedure TFormMenu.BitBtnBSClick(Sender: TObject);
Begin
  FormBujurSangkar.ShowModal;
End;

Procedure TFormMenu.About1Click(Sender: TObject);
Begin
  AboutBoxKu.ShowModal;
End;

Procedure TFormMenu.BitBtnEXITClick(Sender: TObject);
Begin
  Application.Terminate
End;

Procedure TFormMenu.SegitigaSamasisi1Click(Sender: TObject);
Begin
  FormSamaSisi.ShowModal;
End;

Procedure TFormMenu.SegitigaSamakaki1Click(Sender: TObject);
Begin
  FormSamaKaki.ShowModal;
End;

Procedure TFormMenu.Bujursangkar1Click(Sender: TObject);
Begin
  FormBujurSangkar.ShowModal;
End;

Procedure TFormMenu.Exit1Click(Sender: TObject);
Begin
  Application.Terminate
End;

End.
```

Lampiran 2.2 List Program Fraktal Koch Snowflake Generator Samasisi

```
// Program Fraktal Koch Snowflake dengan Generator Segitiga Samasisi

unit SamaSisi;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms,
  Dialogs, ComCtrls, StdCtrls, ExtCtrls, Spin, ExtDlgs, Menus, Buttons;

type
  TFormSamaSisi = class(TForm)
```

```
    ImageKoch: TImage;
    LabelPoligon: TLabel;
    SpinEditSisi: TSpinEdit;
    LabelLevel: TLabel;
    SpinEditLevel: TSpinEdit;
    LabelRadius: TLabel;
    SpinEditPanjang: TSpinEdit;
    LabelPutar: TLabel;
    SpinEditPutar: TSpinEdit;
    CheckBoxAnti: TCheckBox;
    Label3: TLabel;
    LPol: TEdit;
    Label1: TLabel;
    Hitung: TButton;
    Sendiri: TButton;
    BitBtn1: TBitBtn;
    LabelJudul: TLabel;
    Label2: TLabel;
    LuasKS: TEdit;
    BitBtn2: TBitBtn;
    Procedure FormCreate(Sender: TObject);
    Procedure SpinEditChange(Sender: TObject);
    Procedure HitungClick(Sender: TObject);
    Procedure BitBtn1Click(Sender: TObject);
    Procedure SendiriClick(Sender: TObject);

  private
    Procedure GambarKoch ;
  public
    { Public declarations }
  End;

var
  FormSamaSisi: TFormSamaSisi;

implementation

{$R *.DFM}

  Uses
    Math,                              // DegToRad
    ShellAPI,                          // ShellExecute
    GambarSamaSisi, LuasSamaSisi;  // TKurvaKoch

  Procedure TFormSamaSisi.GambarKoch;
  Var
      Bitmap:  TBitmap;
      Kurva :  TKurvaKoch;
  Begin
    Bitmap := TBitmap.Create;
    TRY
      Bitmap.Width  := ImageKoch.Width;
      Bitmap.Height := ImageKoch.Height;
      Bitmap.PixelFormat := pf24bit;

      Bitmap.Canvas.Pen.Color := clBlue;
      Bitmap.Canvas.Pen.Width := 1;

      Kurva := TKurvaKoch.Create(Bitmap.Canvas, Bitmap.Canvas.ClipRect);
      TRY
        Kurva.Gambar(SpinEditLevel.Value,
                     SpinEditSisi.Value,
                     SpinEditPanjang.Value,
```

```
                    SpinEditPutar.Value,
                    CheckBoxAnti.Checked);
          IF   SpinEditSisi.Value > 2 THEN
          Begin
            // Floodfill Gambar Bagian Luar Koch Snowflake Dengan Warna
            // form background.
            // Floodfill Gambar Bagian Dalam Koch Snowflake Dengan Warna
            // Putih.
            Bitmap.Canvas.Brush.Color := FormSamaSisi.Color;
            Bitmap.Canvas.FloodFill(0, 0, clWhite, fsSurface);
            Bitmap.Canvas.FloodFill(0, Bitmap.Height-1, clWhite,fsSurface);
            Bitmap.Canvas.FloodFill(Bitmap.Width-1, 0, clWhite, fsSurface);
            Bitmap.Canvas.FloodFill(Bitmap.Width-1,Bitmap.Height-1,clWhite,
            fsSurface)
          End;

          ImageKoch.Picture.Graphic := Bitmap
        FINALLY
          Kurva.Free
        End

      FINALLY
        Bitmap.Free
      End
    End {Gambar Koch};

Procedure TFormSamaSisi.FormCreate(Sender: TObject);
Begin
  GambarKoch
End;

Procedure TFormSamaSisi.SpinEditChange(Sender: TObject);
Begin
  TRY
    IF   SpinEditSisi.Value = 2
    THEN LabelPoligon.Caption := 'Kurva Koch'
    ELSE LabelPoligon.Caption := 'Poligon Regular Bersisi:';

    IF   SpinEditSisi.Value = 3
    THEN LabelJudul.Caption := 'Fraktal Koch Snowflake'
    ELSE LabelJudul.Caption := 'Fraktal Koch Snowflake';

    IF   CheckBoxAnti.Checked
    THEN LabelJudul.Caption := 'Fraktal Anti Koch Snowflake';

    GambarKoch
  FINALLY
    Screen.Cursor := crDefault
  End
End;

Procedure TFormSamaSisi.HitungClick(Sender: TObject);
var L,LuasPoli,Luas,Rad :real;
    LuasGen :real;
    Jumlah,b,Total :real;
    i,k,n,a :integer;
    Suku :array[1..8000]of real;
Begin
   L:= StrToFloat(SpinEditPanjang.Text);
   n:= StrToInt(SpinEditSisi.Text);
   k:= StrToInt(SpinEditLevel.Text);
   Rad:=DegToRad(180/n);
```

```
    LuasPoli:=(n*sqr LuasPoli:=(n*sqr(L)/4)*cotan(rad);
    if n < 3 then
       LuasPoli:=0;
       LPol.Text:=FloatToStr(LuasPoli);

    Total:=0;
    Suku[1]:=1;
    a:=1;
    b:=(4/9);
    for i:= 2 to k do
      Begin
        Suku[i]:= b*Suku[i-1];
      End;
    for i:= 1 to k do
      Begin
        Jumlah :=Suku[i];
        Total:=Jumlah + Total;
      End;
    LuasGen:=(n*(sqrt(3)/4)*sqr(L)/36)*Total;
    Luas:=LuasPoli+LuasGen;
    LuasKS.Text:=FloatToStr(Luas);
    Label1.Caption := 'Luas Koch Snowflake';

    if   CheckBoxAnti.Checked then
    Luas:=LuasPoli-LuasGen;
    LuasKS.Text:=FloatToStr(Luas);

    if   CheckBoxAnti.Checked then
    Label1.Caption := 'Luas Anti Koch Snowflake';
End;

Procedure TFormSamaSisi.BitBtn1Click(Sender: TObject);
Begin
Application.Terminate
End;

Procedure TFormSamaSisi.SendiriClick(Sender: TObject);
Begin
FormLKS.ShowModal;
End;

End.
```

Lampiran 2.3 List Program Menggambar Fraktal Koch Snowflake Generator Segitiga Samasisi

```
// " Program untuk menggambarkan fraktal Koch Snowflake dengan generator
//   Segitiga samasisi pada Image Koch "

unit GambarSamaSisi;

INTERFACE

  USES
    Windows,           // TRect
    Graphics,          // TCanvas
    MapWorldToPixel;   // TSimplePantograph

  TYPE
    TKurvaKoch =
```

```
  CLASS(TObject)
    PRIVATE
      FCanvas     :  TCanvas;
      FPixelRect  :  TRect;
      FWorldRect  :  TRealRect;
      FPantograph:  TDigitalPantograph;

    PUBLIC
      CONSTRUCTOR Create(CONST Canvas:  TCanvas;
                         CONST PixelRect:  TRect);
      DESTRUCTOR  Destroy;  Override;

      PROCEDURE Gambar(CONST Level    :  INTEGER;
                       CONST Sisi     :  INTEGER;
                       CONST Panjang  :  INTEGER;
                       CONST Putar    :  INTEGER;
                       CONST AntiKoch :  BOOLEAN);

  End;

IMPLEMENTATION

  USES
    Math;              // DegToRad {Derajat ke radian}

  VAR
    FaktorPembagi:  TReal;   // Mendefinisikan variabel FaktorUkur


  CONSTRUCTOR TKurvaKoch.Create(CONST Canvas:  TCanvas;
                                CONST PixelRect:  TRect);
  Begin
    Inherited Create;

    FCanvas := Canvas;
    FPixelRect := PixelRect;

    // Membatasi Panjang dalam Menggambar
    FWorldRect := RealRect(-150,-150, +150,+150);

    Fpantograph := TDigitalPantograph.Create(FCanvas,PixelRect,
                   FWorldRect)
  End ;


  DESTRUCTOR TKurvaKoch.Destroy;
  Begin
    FPantograph.Free;
    Inherited Destroy
  End {Destroy};


  Procedure TKurvaKoch.Gambar(CONST Level    :  INTEGER;
                              CONST Sisi     :  INTEGER;
                              CONST Panjang  :  INTEGER;
                              CONST Putar    :  INTEGER;
                              CONST AntiKoch :  BOOLEAN);
    VAR
      i             :  INTEGER;
      TitikTambahan :  INTEGER;
      Titik         :  ARRAY OF TRealPoint;
      Radian        :  TReal;
```

```
Procedure BuatSegmen(CONST a,b:  TRealPoint; CONST level:  INTEGER);
  VAR
    m     :  TRealPoint;
    r     :  TRealPoint;
    s     :  TRealPoint;
    t     :  TRealPoint;
    xDelta:  TReal;
    yDelta:  TReal;

Begin
  If   level < 1 Then
  Begin
    // Plot segmen hanya untuk level 0
    fPantograph.MoveTo(a);
    fPantograph.LineTo(b)
  End
  Else
  Begin
    If   AntiKoch Then
    Begin
      //  "Modifikasi Kurva Koch  dengan generator segitiga samasisi
      //   Untuk Anti Koch Snowflake.
      //   Mengganti segmen ab dengan 4 segmen, ar, rs, st, tb
      //
      //  a........r    m    t........b
      //                .  .
      //                 .    .
      //                  .  .
      //                   . .
      //                    s
      //
      //  ar = st = ub
      //  sr = tu
      //  (3/4)ar = sr
      //
      // Membagi 1 segmen menjadi 3 bagian
      r := RealPoint( (2.0*a.x +     b.x) / 3.0,
                      (2.0*a.y +     b.y) / 3.0 );
      t := RealPoint( (    a.x + 2.0*b.x) / 3.0,
                      (    a.y + 2.0*b.y) / 3.0 );
      m := RealPoint( (    a.x +     b.x) / 2.0,
                      (    a.y +     b.y) / 2.0 );

      xDelta := b.x - a.x;
      yDelta := b.y - a.y;

      // Untuk menentukan Titik s.
      s := RealPoint( m.x - FaktorPembagi*yDelta,
                      m.y + FaktorPembagi*xDelta );

      // membagi segmen menjadi empat bagian.
      BuatSegmen(a,r, level-1);
      BuatSegmen(r,s, level-1);
      BuatSegmen(s,t, level-1);
      BuatSegmen(t,b, level-1)
    End
    Else
    Begin
      //  Kurva Koch dengan generator segitiga samasisi untuk Koch
      //  Snowflake.
      //  Mengganti segmen ab dengan 4 segmen, ar, rs, st, tb
      //
```

```
            //                s
            //               . .
            //              .   .
            //             .     .
            //            .       .
            //  a........r    m    t........b
            //
            //  ar = rs = st = tb = 1  {ukuran relatif}
            //  rm = mt = 1/2
            //  ms = SQRT(3)/2
            //
            // Membagi 1 segmen menjadi 3 bagian
            r := RealPoint( (2.0*a.x +     b.x) / 3.0,
                            (2.0*a.y +     b.y) / 3.0 );
            t := RealPoint( (   a.x + 2.0*b.x) / 3.0,
                            (   a.y + 2.0*b.y) / 3.0 );
            m := RealPoint( (   a.x +     b.x) / 2.0,
                            (   a.y +     b.y) / 2.0 );

            xDelta := b.x - a.x;
            yDelta := b.y - a.y;

            // Untuk menentukan Titik s.
            s := RealPoint( m.x + FaktorPembagi*yDelta,
                            m.y - FaktorPembagi*xDelta );

            // membagi segmen menjadi empat bagian.
            BuatSegmen(a,r, level-1);
            BuatSegmen(r,s, level-1);
            BuatSegmen(s,t, level-1);
            BuatSegmen(t,b, level-1)
          End
        End
      End {BuatSegmen};

    Begin
      Assert (Panjang <= 110, 'Terlalu Panjang, gambar mungkin terpotong');
      If   Sisi < 3 Then
      TitikTambahan := Sisi    // Kasus untuk Sisi = 2 -- bukan poligon
      Else TitikTambahan := Sisi+1; // tambah 1 titik untuk membuat poligon

      SetLength(Titik, TitikTambahan);

      // mengambar poligon reguler bersisi n
      // Points[0] = Points[nilai point] untuk Sisi > 2
      For i := 0 TO TitikTambahan-1 DO
      Begin
        Radian := DegToRad(Putar + 360*i/Sisi);
        Titik[i] := RealPoint( Panjang*COS(Radian), Panjang*SIN(Radian) );
      End;

      // Gambar segmen untuk setiap segmen poligon
      For i := 1 TO TitikTambahan-1 DO
      Begin
        BuatSegmen(Titik[i-1], Titik[i], Level);
      End
    End {Gambar};


INITIALIZATION
  FaktorPembagi := SQRT(3) / 6;
End.Interface
```

```
implementation

End.
```

Lampiran 2.4 List Program Menghitung Luas Koch Snowflake Generator Samasisi

```
// Program Menghitung Luas Koch Snowflake Generator Segitiga Samasisi

unit LuasSamaSisi;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls,
Forms,
  Dialogs, Buttons, StdCtrls;

type
  TFormLSS = class(TForm)
    Panjang: TEdit;
    Judul: TLabel;
    Judul2: TLabel;
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    Sisi: TEdit;
    Iterasi: TEdit;
    LuasPol: TEdit;
    Label4: TLabel;
    Label5: TLabel;
    LabelKS: TLabel;
    Anti: TCheckBox;
    Hitung: TButton;
    BitBtn2: TBitBtn;
    Koch: TEdit;
    Gen: TEdit;
    Label6: TLabel;
    Procedure HitungClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  FormLSS: TFormLSS;

implementation
uses Math;
{$R *.dfm}

Procedure TFormLSS.HitungClick(Sender: TObject);
var L,LuasPol1,Luas,Rad :real;
    LuasGen :real;
    Jumlah,b,Total:real;
    i,k,n,a :integer;
    Suku :array[1..120000]of real;

begin
   L:= StrToFloat(Panjang.Text);
```

```
n:= StrToInt(Sisi.Text);
k:= StrToInt(Iterasi.Text);
if n < 2then
 begin
   n:=2;
   Sisi.Text:=IntToStr(n);
   Application.MessageBox('Minimal Sisinya adalah 2','Sisi
   Minimal',MB_OK);
 End;

if k >=120000 then
 begin
   k:=119999;
   Iterasi.Text:=IntToStr(k);
   Application.MessageBox('Level Terlalu Besar','Level Maksimal
   199999',MB_OK);
 End;

Rad:=DegToRad(180/n);

LuasPoli:=(n*sqr(L)/4)*cotan(rad);
LuasPol.Text:=FloatToStr(LuasPoli);

if n < 3 then
LuasPoli:=0;
LuasPol.Text:=FloatToStr(LuasPol);

Total:=0;
a:=1;
Suku[1]:=a;

b:=(4/9);

for i:= 2 to k do
  begin
    Suku[i]:= b*Suku[i-1];
  End;
for i:= 1 to k do
  begin
    Jumlah :=Suku[i];
    Total:=Jumlah + Total;
  End;
LuasGen:=(n*(sqrt(3)*sqr(L))/36)*Total;
Gen.Text:=FloatToStr(LuasGen);

Luas:=LuasPoli+LuasGen;
Judul.Caption := ' HITUNG LUAS FRAKTAL KOCH SNOWFLAKE';
LabelKS.Caption := 'Luas Koch Snowflake';
Koch.Text:=FloatToStr(Luas);

if Anti.Checked then
begin
Judul.Caption := ' HITUNG LUAS FRAKTAL ANTI KOCH SNOWFLAKE';
LabelKS.Caption := 'Luas Anti Koch Snowflake';
Luas:=LuasPoli-LuasGen;
Koch.Text:=FloatToStr(Luas)
End;
End;

End.
```

Lampiran 2.5 List Program Fraktal Koch Snowflake Generator Samakaki

```
// Program Fraktal Koch Snowflake dengan Generator Segitiga Samakaki

unit SamaKaki;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms,
Dialogs,
  ComCtrls, StdCtrls, ExtCtrls, Spin, ExtDlgs, Menus, Buttons;

type
  TFormSamaKaki = class(TForm)
    ImageKoch: TImage;
    LabelPolygon: TLabel;
    SpinEditSisi: TSpinEdit;
    LabelLevel: TLabel;
    SpinEditLevel: TSpinEdit;
    LabelRadius: TLabel;
    SpinEditPanjang: TSpinEdit;
    LabelPutar: TLabel;
    SpinEditPutar: TSpinEdit;
    CheckBoxAnti: TCheckBox;
    SendiriBtn: TButton;
    BitBtn1: TBitBtn;
    LabelJudul: TLabel;
    Label2: TLabel;
    BitBtn2: TBitBtn;
    SpinEditBanding: TSpinEdit;
    Procedure FormCreate(Sender: TObject);
    Procedure SpinEditChange(Sender: TObject);
    Procedure BitBtn1Click(Sender: TObject);
    Procedure SendiriClick(Sender: TObject);

  private
    Procedure GambarKoch ;
  public
    { Public declarations }
  End;

var
  FormSamaKaki: TFormSamaKaki;

implementation

{$R *.DFM}

  USES
    Math,                  // DegToRad
    ShellAPI,              // ShellExecute
    GambarSamaKaki,LuasSamaKaki;       // TKurvaKoch

  Procedure TFormSamaKaki.GambarKoch;
    VAR
      Bitmap:  TBitmap;
      Kurva :  TKurvaKoch;
  BEGIN
    Bitmap := TBitmap.Create;
    TRY
      Bitmap.Width  := ImageKoch.Width;
      Bitmap.Height := ImageKoch.Height;
```

```
        Bitmap.PixelFormat := pf24bit;
        Bitmap.Canvas.Pen.Color := clBlue;
        Bitmap.Canvas.Pen.Width := 1;

        Kurva := TKurvaKoch.Create(Bitmap.Canvas, Bitmap.Canvas.ClipRect);
        TRY
          Kurva.Gambar(SpinEditLevel.Value,
                       SpinEditSisi.Value,
                       SpinEditPanjang.Value,
                       SpinEditPutar.Value,
                       SpinEditBanding.Value,
                       CheckBoxAnti.Checked);

          IF   SpinEditSisi.Value > 2  THEN
          BEGIN
           // Floodfill Gambar Bagian Luar Koch Snowflake Dengan Warna form
           // background.
           // Floodfill Gambar Bagian Dalam Koch Snowflake Dengan Warna
           // Putih.
           Bitmap.Canvas.Brush.Color := FormSamaKaki.Color;
           Bitmap.Canvas.FloodFill(0, 0, clWhite, fsSurface);
           Bitmap.Canvas.FloodFill(0,Bitmap.Height-1, clWhite, fsSurface);
           Bitmap.Canvas.FloodFill(Bitmap.Width-1,0, clWhite, fsSurface);
           Bitmap.Canvas.FloodFill(Bitmap.Width-1,Bitmap.Height-1,clWhite,
           fsSurface)
          End;

          ImageKoch.Picture.Graphic := Bitmap
        FINALLY
          Kurva.Free
        End

      FINALLY
        Bitmap.Free
      End
    End {GambarVonKoch};

Procedure TFormSamaKaki.FormCreate(Sender: TObject);
begin
  GambarKoch
End;

Procedure TFormSamaKaki.SpinEditChange(Sender: TObject);
begin
  TRY
    IF   SpinEditSisi.Value = 2
    THEN LabelPoligon.Caption := 'Kurva Koch'
    ELSE LabelPoligon.Caption := 'Poligon Reguler Bersisi:';

    IF   SpinEditSisi.Value = 3
    THEN LabelJudul.Caption := 'Fraktal Koch Snowflake'
    ELSE LabelJudul.Caption := 'Fraktal Koch Snowflake';

    IF   CheckBoxAnti.Checked
    THEN LabelJudul.Caption := 'Fraktal Anti Koch Snowflake';

    GambarKoch
  FINALLY
    Screen.Cursor := crDefault
  End
End;

Procedure TFormSamaKaki.BitBtn1Click(Sender: TObject);
```

```
begin
  Application.Terminate
End;

Procedure TFormSamaKaki.SendirlClick(Sender: TObject);
begin
  FormLSK.ShowModal;
End;
End.
```

Lampiran 2.6 List Program Menggambar Fraktal Koch Snowflake Generator Samakaki

```
// " Program untuk menggambarkan fraktal Koch Snowflake dengan generator
//   segitiga samakaki pada image Koch "

unit GambarSamaKaki;

INTERFACE

  USES
    Windows,          // TRect
    Graphics,         // TCanvas
    MapWorldToPixel;  // TSimplePantograph

  TYPE
    TKurvaKoch =
    CLASS(TObject)
      PRIVATE
        FCanvas     :  TCanvas;
        FPixelRect  :  TRect;
        FWorldRect  :  TRealRect;
        FPantograph:   TDigitalPantograph;

      PUBLIC
        CONSTRUCTOR Create(CONST Canvas:  TCanvas;
                           CONST PixelRect:  TRect);
        DESTRUCTOR  Destroy;  Override;

        Procedure Gambar(CONST Level    :  INTEGER;
                         CONST Sisi     :  INTEGER;
                         CONST Panjang  :  INTEGER;
                         CONST Putar    :  INTEGER;
                         CONST Banding  :  INTEGER;
                         CONST AntiKoch :  BOOLEAN);

    End;

IMPLEMENTATION

  USES
    Math;              // DegToRad (Derajat ke radian)

  VAR
    FaktorPembagi:  TReal;   // Mendefinisikan variabel Faktorukur


  CONSTRUCTOR TKurvaKoch.Create(CONST Canvas:  TCanvas;
                                CONST PixelRect:  TRect);
  BEGIN
    Inherited Create;
```

```
  FCanvas := Canvas;
  FPixelRect := PixelRect;

  // Membatasi Panjang dalam Menggambar
  FWorldRect  := RealRect(-150,-150, +150,+150);
  Fpantograph := TDigitalPantograph.Create(FCanvas, PixelRect,
                 FWorldRect);
End ;

DESTRUCTOR TKurvaKoch.Destroy;
BEGIN
  FPantograph.Free;
  Inherited Destroy
End {Destroy};

Procedure TKurvaKoch.Gambar(CONST Level    : INTEGER;
                            CONST Sisi     : INTEGER;
                            CONST Panjang  : INTEGER;
                            CONST Putar    : INTEGER;
                            CONST Banding  : INTEGER;
                            CONST AntiKoch : BOOLEAN);

 VAR
   p             : REAL;
   i             : INTEGER;
   TitikTambahan : INTEGER;
   Titik         : ARRAY OF TRealPoint;
   radian        : TReal;

 Procedure BuatSegmen(CONST a,b: TRealPoint; CONST level: INTEGER);
   VAR
     m     : TRealPoint;
     r     : TRealPoint;
     s     : TRealPoint;
     t     : TRealPoint;
     xDelta: TReal;
     yDelta: TReal;

 BEGIN
   IF  level < 1 THEN
    BEGIN
    // Plot segmen hanya untuk level 0
      FPantograph.MoveTo(a);
      FPantograph.LineTo(b);
    End
   ELSE BEGIN
     IF  AntiKoch THEN
     BEGIN
       //  Modifikasi Kurva Koch  dengan generator segitiga samakaki
       //  untuk Anti Koch Snowflake.
       //  Mengganti segmen ab dengan 4 segmen, ar, rs, st, tb
       //
       //  a........r    m    t........b
       //             .       .
       //              .     .
       //               .   .
       //                . .
       //                 s
       //
       //  ar = rs = st = tb = 1  (ukuran relatif)
       //  rm = mt = 1/2
       //  ms = SQRT(3)/2
       //
```

```
        // Membagi 1 segmen menjadi 3 bagian
        r := RealPoint( (2.0*a.x +     b.x) / 3.0,
                        (2.0*a.y +     b.y) / 3.0 );
        t := RealPoint( (    a.x + 2.0*b.x) / 3.0,
                        (    a.y + 2.0*b.y) / 3.0 );
        m := RealPoint( (    a.x +     b.x) / 2.0,
                        (    a.y +     b.y) / 2.0 );

        xDelta := b.x - a.x;
        yDelta := b.y - a.y;

        // Untuk menentukan Titik s.
        s := RealPoint( m.x - FaktorPembagi*yDelta,
                        m.y + FaktorPembagi*xDelta );

        // membagi segmen menjadi empat bagian.
        BuatSegmen(a,r, level-1);
        BuatSegmen(r,s, level-1);
        BuatSegmen(s,t, level-1);
        BuatSegmen(t,b, level-1)
      End
      Else
      BEGIN
        //  Kurva Koch dengan generator segitiga samakaki untuk Koch
        //  Snowflake.
        //  Mengganti segmen ab dengan 4 segmen, ar, rs, st, tb
        //
        //                 s
        //                . .
        //               .   .
        //              .     .
        //             .       .
        //            .         .
        //  a.........r     m     t.........b
        //
        //  ar = rs = st = tb = 1  (ukuran relatif)
        //  rm = mt = 1/2
        //  ms = SQRT(3)/2
        //
        // Membagi 1 segmen menjadi 3 bagian
        r := RealPoint( (2.0*a.x +     b.x) / 3.0,
                        (2.0*a.y +     b.y) / 3.0 );
        t := RealPoint( (    a.x + 2.0*b.x) / 3.0,
                        (    a.y + 2.0*b.y) / 3.0 );
        m := RealPoint( (    a.x +     b.x) / 2.0,
                        (    a.y +     b.y) / 2.0 );

        xDelta := b.x - a.x;
        yDelta := b.y - a.y;

        // Untuk menentukan Titik s.
        s := RealPoint( m.x + FaktorPembagi*yDelta,
                        m.y - FaktorPembagi*xDelta );

        // membagi segmen menjadi empat bagian.
        BuatSegmen(a,r, level-1);
        BuatSegmen(r,s, level-1);
        BuatSegmen(s,t, level-1);
        BuatSegmen(t,b, level-1)
      End
    End
  End {BuatSegmen};

Begin
```

```
    ASSERT (Panjang <= 110, 'Terlalu Panjang, gambar mungkin terpotong');
    IF   Sisi < 3
    THEN TitikTambahan := Sisi // Kasus untuk Sisi = 2 -- bukan poligon
    ELSE TitikTambahan := Sisi+1; // tambah 1 titik untuk membuat poligon
    SetLength(Titik, TitikTambahan);

    // mengambar poligon reguler berisi n
    // Points[0] = Points[nilai point] untuk Sisi > 2
    FOR i := 0 TO TitikTambahan-1 DO
    BEGIN
      radian := DegToRad(Putar + 360*i/Sisi);
      Titik[i] := RealPoint( Panjang*COS(radian), Panjang*SIN(radian) )
    End;

    // Gambar segmen untuk setiap segmen poligon
    FOR i := 1 TO TitikTambahan-1 DO
    BEGIN
      BuatSegmen(Titik[i-1], Titik[i], Level)
    End;

    BEGIN
      p:=Banding/100;
      FaktorPembagi := (SQRT(3) / 2)/(2+p);
    End;
  End (Gambar);


INITIALIZATION

End.interface

implementation

End.
```

Lampiran 2.7 List Program Menghitung Luas Koch Snowflake Generator Segitiga Samakaki

```
// Program Menghitung Luas Koch Snowflake dengan Generator Segitiga
Samakaki
unit LuasSamaKaki;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls,
Forms,
  Dialogs, Buttons, StdCtrls;

type
  TFormLSK = class(TForm)
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    Label4: TLabel;
    Label5: TLabel;
    Label6: TLabel;
    Label7: TLabel;
    Label8: TLabel;
    Panjang: TEdit;
    Sisi: TEdit;
```

```
    Iterasi: TEdit;
    Hitung: TButton;
    LuasPol: TEdit;
    Gen: TEdit;
    Koch: TEdit;
    Miring: TEdit;
    Alas: TEdit;
    Asumsi: TGroupBox;
    Label9: TLabel;
    Label10: TLabel;
    Anti: TCheckBox;
    EXIT: TBitBtn;
    Label6: TLabel;
    procedure HitungClick(Sender: TObject);
    procedure AsumsiClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  FormLSK: TFormLSK;

implementation
Uses Math,Asumsi;
{$R *.dfm}

procedure TFormLSK.HitungClick(Sender: TObject);
var L,LuasPoli,Luas,Rad :real;
    LuasGen :real;
    Jumlah,b,Total,p,q :real;
    i,k,n,a :integer;
    Suku :array[1..120000]of real;
begin
   L:= StrToFloat(Panjang.Text);
   n:= StrToInt(Sisi.Text);
   k:= StrToInt(Iterasi.Text);
   p:= StrToFloat(Miring.Text);
   q:=1-2*p;

   if n < 2then
    begin
      n:=2;
      Sisi.Text:=IntToStr(n);
      Application.MessageBox('Minimal     Sisinya     adalah     2','Sisi
Minimal',MB_OK);
    end;

   if k >=120000 then
    begin
      k:=119999;
      Iterasi.Text:=IntToStr(k);
      Application.MessageBox('Level    Terlalu    Besar','Level    Maksimal
119999',MB_OK);
    end;

   Alas.Text:= FloatToStr(q);
   Rad:=DegToRad(180/n);

   LuasPoli:=(n*sqr(L)/4)*cotan(rad);
   LuasPol.Text:=FloatToStr(LuasPoli);
```

```
    if n < 3 then
    LuasPoli:=0;
    LuasPol.Text:=FloatToStr(LuasPoli);


    Total:=0;
    Suku[1]:=1;
    a:=1;
    b:=(4*sqr(p));

    for i:= 2 to k do
      begin
        Suku[i]:= b*Suku[i-1];
      end;
    for i:= 1 to k do
      begin
        Jumlah :=Suku[i];
        Total:=Jumlah + Total;
      end;
    LuasGen:= (n*sqr(L)/4)*(1-2*p)*(4*p-1)*Tota);
    Gen.Text:=FloatToStr(LuasGen);

    if ((p >=0.5 )OR (p <=0.2))then
      begin
        Application.MessageBox('Data        tidak        Sesuai        dengan
Asumsi','Peringatan',
            MB_OK);
        LuasGen:=0;
        Gen.Text:=FloatToStr(LuasGen);
      end;

    Luas:=LuasPoli+LuasGen;
    LabelKS.Caption := 'Luas Koch Snowflake';
    Koch.Text:=FloatToStr(Luas);

    if Anti.Checked then
    begin
    LabelKS.Caption := 'Luas Anti Koch Snowflake';
    Luas:=LuasPoli-LuasGen;
    Koch.Text:=FloatToStr(Luas)
    end;
end;

procedure TformLSK.AsumsiClick(Sender: TObject);
begin
AboutBoxAsumsi.ShowModal;
end;

end.
```

Lampiran 2.8 List Program Fraktal Koch Snowflake Generator Bujursangkar

```
// Program Fraktal Koch Snowflake dengan Generator Bujursangkar

unit BujurSangkar;

interface

uses
```

```
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms,
Dialogs,
  ComCtrls, StdCtrls, ExtCtrls, Spin, ExtDlgs, Menus, Buttons;

type
  TFormBujurSangkar = class(TForm)
    ImageKoch: TImage;
    LabelPoligon: TLabel;
    SpinEditSisi: TSpinEdit;
    LabelLevel: TLabel;
    SpinEditLevel: TSpinEdit;
    LabelRadius: TLabel;
    SpinEditPanjang: TSpinEdit;
    LabelPutar: TLabel;
    SpinEditPutar: TSpinEdit;
    CheckBoxAnti: TCheckBox;
    Label3: TLabel;
    LPul: TEdit;
    Label1: TLabel;
    LuasKS: TEdit;
    Hitung: TButton;
    Sendiri: TButton;
    BitBtn1: TBitBtn;
    LabelJudul: TLabel;
    Label2: TLabel;
    BitBtn2: TBitBtn;
    Procedure FormCreate(Sender: TObject);
    Procedure SpinEditChange(Sender: TObject);
    Procedure HitungClick(Sender: TObject);
    Procedure BitBtn1Click(Sender: TObject);
    Procedure SendiriClick(Sender: TObject);

  private
    Procedure GambarKoch ;
  public
    { Public declarations }
  End;

var
  FormBujurSangkar: TFormBujurSangkar;

implementation

{$R *.DFM}

  USES
    Math,                    // DegToRad
    ShellAPI,                // ShellExecute
    GambarBujurSangkar, LuasBujurSangkar;  // TVonKochCurve

  Procedure TFormBujurSangkar.GambarKoch;
    VAR
      Bitmap:  TBitmap;
      Kurva :  TKurvaKoch;
  BEGIN
    Bitmap := TBitmap.Create;
    TRY
      Bitmap.Width  := ImageKoch.Width;
      Bitmap.Height := ImageKoch.Height;
      Bitmap.PixelFormat := pf24bit;

      Bitmap.Canvas.Pen.Color := clBlue;
      Bitmap.Canvas.Pen.Width := 1;
```

```
      Kurva := TKurvaKoch.Create(Bitmap.Canvas, Bitmap.Canvas.ClipRect);
      TRY
        Kurva.Gambar(SpinEditLevel.Value,
                     SpinEditSisi.Value,
                     SpinEditPanjang.Value,
                     SpinEditPutar.Value,
                     CheckBoxAnti.Checked);

        IF   SpinEditSisi.Value > 2 THEN
        BEGIN
         // Floodfill Gambar Bagian Luar Koch Snowflake Dengan Warna form
         // background.
         // Floodfill Gambar Bagian Dalam Koch Snowflake Dengan Warna
         // Putih.
         Bitmap.Canvas.Brush.Color := FormBujurSangkar.Color;
         Bitmap.Canvas.FloodFill(0, 0, clWhite, fsSurface);
          Bitmap.Canvas.FloodFill(0, Bitmap.Height-1, clWhite,fsSurface);
          Bitmap.Canvas.FloodFill(Bitmap.Width-1, 0, clWhite, fsSurface);
          Bitmap.Canvas.FloodFill(Bitmap.Width-1,Bitmap.Height-1,clWhite,
          fsSurface)
        End;

        ImageKoch.Picture.Graphic := Bitmap
      FINALLY
        Kurva.Free
      End

    FINALLY
      Bitmap.Free
    End
  End {GambarVonKoch};


Procedure TFormBujurSangkar.FormCreate(Sender: TObject);
begin
  GambarKoch
End;

Procedure TFormBujurSangkar.SpinEditChange(Sender: TObject);
begin
  TRY
    IF   SpinEditSisi.Value = 2
    THEN LabelPoligon.Caption := 'Kurva Koch'
    ELSE LabelPoligon.Caption := 'Poligon Reguler Bersisi:';

    IF   SpinEditSisi.Value = 3
    THEN LabelJudul.Caption := 'Fraktal Koch Snowflake'
    ELSE LabelJudul.Caption := 'Fraktal Koch Snowflake';

    IF   CheckBoxAnti.Checked
    THEN LabelJudul.Caption := 'Fraktal Anti Koch Snowflake';

    GambarKoch
  FINALLY
    Screen.Cursor := crDefault
  End
End;

Procedure TFormBujurSangkar.HitungClick(Sender: TObject);
var L,LuasPoli,Luas,Rad :real;
    LuasGen :real;
    Jumlah,b,Total :real;
```

```
    i,k,n,a :integer;
    Suku :array[1..8000]of real;
begin
   L:= StrToFloat(SpinEditPanjang.Text);
   n:= StrToInt(SpinEditSisi.Text);
   k:= StrToInt(SpinEditLevel.Text);
   Rad:=DegToRad(180/n);

   LuasPoli:=(n*sqr(L)/4)*cotan(rad);
   if n < 3 then
   LuasPoli:=0;
   LPol.Text:=FloatToStr(LuasPoli);

   Total:=0;
   Suku[1]:=1;
   a:=1;
   b:=(5/9);
   for i:= 2 to k do
     begin
       Suku[i]:= b*Suku[i-1];
     End;
   for i:= 1 to k do
     begin
       Jumlah :=Suku[i];
       Total:=Jumlah + Total;
     End;
   LuasGen:=(n*sqr(L)/9)*Total;
   Luas:=LuasPoli+LuasGen;
   LuasKS.Text:=FloatToStr(Luas);
   Label1.Caption := 'Luas Koch Snowflake';

   if   CheckBoxAnti.Checked then
   Luas:=LuasPoli-LuasGen;
   LuasKS.Text:=FloatToStr(Luas);
   if   CheckBoxAnti.Checked then
   Label1.Caption := 'Luas Anti Koch Snowflake';
End;

Procedure TFormBujurSangkar.BitBtnClick(Sender: TObject);
begin
Application.Terminate
End;

Procedure TFormBujurSangkar.KendiriClick(Sender: TObject);
begin
  FormLBS.ShowModal;
End;

End.
```

Lampiran 2.9 List Program Menggambar Fraktal Koch Snowflake Generator Bujursangkar

```
// " Program untuk menggambarkan fraktal Koch Snowflake dengan generator
//   Bujursangkar pada Image Koch "

UNIT GambarBujurSangkar;

INTERFACE
```

```
USES
  Windows,          // TRect
  Graphics,         // TCanvas
  MapWorldToPixel;  // TSimplePantograph

TYPE
  TKurvaKoch =
  CLASS(TObject)
    PRIVATE
      FCanvas     :  TCanvas;
      FPixelRect  :  TRect;
      FWorldRect  :  TRealRect;
      FPantograph:  TDigitalPantograph;

    PUBLIC
      CONSTRUCTOR Create(CONST Canvas:  TCanvas;
                         CONST PixelRect:  TRect);
      DESTRUCTOR  Destroy;  Override;

      Procedure Gambar(CONST Level    :  INTEGER;
                       CONST Sisi     :  INTEGER;
                       CONST Panjang  :  INTEGER;
                       CONST Putar    :  INTEGER;
                       CONST AntiKoch :  BOOLEAN);

  End;

IMPLEMENTATION

 USES
   Math;              // DegToRad (Derajat ke radian)

 VAR
   FaktorUkur:  TReal;   // Mendefinisikan variabel Faktorukur

 CONSTRUCTOR TKurvaKoch.Create(CONST Canvas:  TCanvas;
                               CONST PixelRect:  TRect);
 BEGIN
   Inherited Create;

   FCanvas := Canvas;
   FPixelRect := PixelRect;

   // Membatasi Panjang dalam Menggambar
   FWorldRect  := RealRect(-160,-160, +160,+160);
   Ppantograph := TDigitalPantograph.Create(FCanvas,PixelRect,
                  FWorldRect)
 End ;

 DESTRUCTOR TKurvaKoch.Destroy;
 BEGIN
   FPantograph.Free;
   Inherited Destroy
 End {Destroy};

 Procedure TKurvaKoch.Gambar(CONST Level    :  INTEGER;
                             CONST Sisi     :  INTEGER;
                             CONST Panjang  :  INTEGER;
                             CONST Putar    :  INTEGER;
                             CONST AntiKoch :  BOOLEAN);
   VAR
     i             :  INTEGER;
     TitikTambahan :  INTEGER;
```

```
  Titik          :  ARRAY OF TRealPoint;
  Radian         :  TReal;

Procedure BuatSegmen(CONST a,b:  TRealPoint; CONST level:  INTEGER);
  VAR
    m      :  TRealPoint;
    r      :  TRealPoint;
    s      :  TRealPoint;
    t      :  TRealPoint;
    u      :  TRealPoint;
    xDelta:  TReal;
    yDelta:  TReal;
BEGIN
  IF   level < 1
  THEN BEGIN
    // Plot segment hanya untuk level 0
    FPantograph.MoveTo(a);
    FPantograph.LineTo(b)
  End
  ELSE BEGIN
    IF   AntiKoch
    THEN BEGIN
      //  Modifikasi Kurva Koch  dengan generator bujursangkar untuk
      //  Anti Koch Snowflake.
      //  Menggganti segmen ab dengan 4 segmen, ar, rs, st, tb
      //
      //
      //  a........r         u........b
      //           .         .
      //           .         .
      //           .         .
      //           s.........t
      //
      //  ar = st = ub
      //  sr = tu
      //  (3/4)ar = sr
      //
      // Membagi 1 segmen menjadi 4 bagian
      r := RealPoint( (2.0*a.x +     b.x) / 3.0,
                      (2.0*a.y +     b.y) / 3.0 );
      u := RealPoint( (    a.x + 2.0*b.x) / 3.0,
                      (    a.y + 2.0*b.y) / 3.0 );

      xDelta := b.x - a.x;
      yDelta := b.y - a.y;

      // Untuk menentukan Titik s dan t.
      s := RealPoint( r.x - yDelta/3,
                      r.y + xDelta/3 );
      t := RealPoint( u.x - yDelta/3,
                      u.y + xDelta/3 );

      // membagi segmen menjadi lima bagian.
      BuatSegmen(a,r, level-1);
      BuatSegmen(r,s, level-1);
      BuatSegmen(s,t, level-1);
      BuatSegmen(t,u, level-1);
      BuatSegmen(u,b, level-1)
    End
    ELSE BEGIN
      // Kurva Koch dengan generator bujursangkar untuk Koch
      // Snowflake.
      //  Mengganti segmen ab dengan 5 segmen, ar, rs, st, tu, ub
```

```
        //
        //          s.........t
        //          .         .
        //          .         .
        //          .         .
        //  a.........r       u.........b
        //
        //  ar = rs = st = tb = 1  (ukuran relatif)
        //  rm = mt = 1/2
        //  ms = SQRT(3)/2

        r := RealPoint( (2.0*a.x +     b.x) / 3.0,
                        (2.0*a.y +     b.y) / 3.0 );
        u := RealPoint( (    a.x + 2.0*b.x) / 3.0,
                        (    a.y + 2.0*b.y) / 3.0 );

        xDelta := b.x - a.x;
        yDelta := b.y - a.y;

        // Untuk menentukan Titik s dan t.
        s := RealPoint( r.x + yDelta/3,
                        r.y - xDelta/3 );
        t := RealPoint( u.x + yDelta/3,
                        u.y - xDelta/3 );

        // membagi segmen menjadi lima bagian.
        BuatSegmen(a,r, level-1);
        BuatSegmen(r,s, level-1);
        BuatSegmen(s,t, level-1);
        BuatSegmen(t,u, level-1);
        BuatSegmen(u,b, level-1)
      End
    End
  End {BuatSegmens};

BEGIN
  ASSERT (Panjang <= 110, 'Terlalu Panjang, gambar mungkin terpotong');
  IF   Sisi < 3 THEN
  TitikTambahan := Sisi // Kasus untuk Sisi = 2 -- bukan poligon
  ELSE TitikTambahan := Sisi+1; // tambah 1 titik untuk membuat poligon
  SetLength(Titik, TitikTambahan);
  // mengambar poligon reguler bersisi n
  // Points[0] = Points[nilai point] untuk Sisi > 2
  FOR i := 0 TO TitikTambahan-1 DO
  BEGIN
    Radian := DegToRad(Putar + 360*i/Sisi);
    Titik[i] := RealPoint( Panjang*COS(Radian), Panjang*SIN(Radian) )
  End;

  // Gambar segmen untuk setiap segmen poligon
  FOR i := 1 TO TitikTambahan-1 DO
  BEGIN
    BuatSegmen(Titik[i-1], Titik[i], Level)
  End
End {Gambar};

INITIALIZATION

End.
```

Lampiran 2.10 List Program Menghitung Luas Koch Snowflake Dengan Generator Bujursangkar

```
// Program Menghitung Luas Koch Snowflake dengan Generator Bujursangkar

unit LuasBujurSangkar;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls,
Forms,
  Dialogs, Buttons, StdCtrls;

type
  TFormLBS = class(TForm)
    Judul: TLabel;
    Judul2: TLabel;
    Label1: TLabel;
    Panjang: TEdit;
    Label2: TLabel;
    Sisi: TEdit;
    Label3: TLabel;
    Iterasi: TEdit;
    Label4: TLabel;
    LuasPoli: TEdit;
    Label5: TLabel;
    Gen: TEdit;
    LabelKS: TLabel;
    Koch: TEdit;
    Anti: TCheckBox;
    Hitung: TButton;
    BitBtn2: TBitBtn;
    Label6: TLabel;
    Procedure HitungClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  End;

var
  FormLBS: TFormLBS;

implementation
Uses Math;
{$R *.dfm}

Procedure TFormLBS.HitungClick(Sender: TObject);
var L,LuasPoli,Luas,Rad :real;
    LuasGen :real;
    Jumlah,b,Total:real;
    i,k,n,a :integer;
    Suku :array[1..120000]of real;
begin
   L:= StrToFloat(Panjang.Text);
   n:= StrToInt(Sisi.Text);
   k:= StrToInt(Iterasi.Text);
   if n < 2then
    begin
      n:=2;
      Sisi.Text:=IntToStr(n);
```

```
        Application.MessageBox('Minimal Sisinya adalah 2','Sisi Minimal',
        MB_OK);
      End;

     if k >=120000 then
      Begin
        k:=119999;
        Iterasi.Text:=IntToStr(k);
        Application.MessageBox('Level Terlalu Besar','Level Maksimal
        19999', MB_OK);
      End;

     Rad:=DegToRad(180/n);

     LuasPoli:=(n*sqr(L)/4)*cotan(rad);
     LuasPol.Text:=FloatToStr(LuasPoli);

     If n < 3 then
     LuasPoli:=0;
     LuasPol.Text:=FloatToStr(LuasPoli);

     Total:=0;
     Suku[1]:=1;
     a:=1;
     b:=(5/9);

     for i:= 2 to k do
       Begin
         Suku[i]:= b*Suku[i-1];
       End;
     for i:= 1 to k do
       Begin
         Jumlah :=Suku[i];
         Total:=Jumlah + Total;
       End;
     LuasGen:=(n*sqr(L)/9)*Total;
     Gen.Text:=FloatToStr(LuasGen);

     Luas:=LuasPoli+LuasGen;
     Judul.Caption := ' HITUNG LUAS FRAKTAL KOCH SNOWFLAKE';
     LabelKS.Caption := 'Luas Koch Snowflake';
     Koch.Text:=FloatToStr(Luas);

     if Anti.Checked then
     Begin
     Judul.Caption := ' HITUNG LUAS FRAKTAL ANTI KOCH SNOWFLAKE';
     LabelKS.Caption := 'Luas Anti Koch Snowflake';
     Luas:=LuasPoli-LuasGen;
     Koch.Text:=FloatToStr(Luas)
     End;
End;

End.
```