PRESEK
List za mlade matematike, fizike, astronome in računalnikarje

ISSN 0351-6652
Letnik **18** (1990/1991)
Številka 1
Strani 56–64, III, IV

Ciril Pezdir:

NENAVADNE KRIVULJE

Ključne besede: računalništvo, matematika, krivulje.

Elektronska verzija: http://www.presek.si/18/1023-Pezdir.pdf

NENAVADNE KRIVULJE

Najprej si bomo ogledali nekaj znanih krivulj in o vsaki povedali kaj zanimivega. Potem se bomo seznanili z enostavnim načinom generiranja vseh teh krivulj. Napisali pa bomo tudi program, s katerim bomo lahko vse te krivulje risali in si izmišljali svoje.

1. Kochova snežinka

Začnimo z enakostraničnim trikotnikom (slika 1a). V prvem koraku razdelimo vsako stranico trikotnika na tretjine, srednje odseke stranic pa nadomestimo z novimi, za tretjino manjšimi enakostraničnimi trikotniki (slika 1b). V drugem koraku naredimo enako z vsako od stranic tako dobljenega poligona, ki spominja na Davidovo zvezdo. Postopek lahko ponavljamo, dokler nam to dopušča natančnost risanja. Če po nekaj korakih postopka lahko približno vidimo, kakšen bo izgled krivulje, obris lika, če bi postopek izvedli neskončnokrat (slika 1e, glej 4. stran ovitka).

Krivulja, ki jo dobimo, če postopek izvedemo neskončnokrat, se imenuje *Kochova snežinka*, saj je podobna pravi snežinki. Prvi jo je raziskoval v začetku tega stoletja Helge von Koch. Zelo je zanimiva, saj ima neskončno velik obseg, pa vendar obsega končno površino. Komur so domača geometrijska zaporedja, mu obsega in površine ne bo težko izračunati.

Poglejmo, kakšen lik dobimo, če srednje dele stranic nadomeščamo s trikotniki obrnjenimi navznoter (slike 2a do 2e). Obseg lika, če postopek ponovimo neskončnokrat, je enak obsegu *Kochove snežinke*, površina lika pa je manjša od površine začetnega trikotnika za toliko, kot je površina *Kochove snežinke* večja od površine istega začetnega trikotnika. Imenujmo ta lik *obratna Kochova snežinka*. Ravnino lahko popolnoma prekrijemo z izmenjajočima se likoma *Kochove snežinke* in *obratne Kochove snežinke* (slika 3, glej 4. stran ovitka).

2. Peanova krivulja

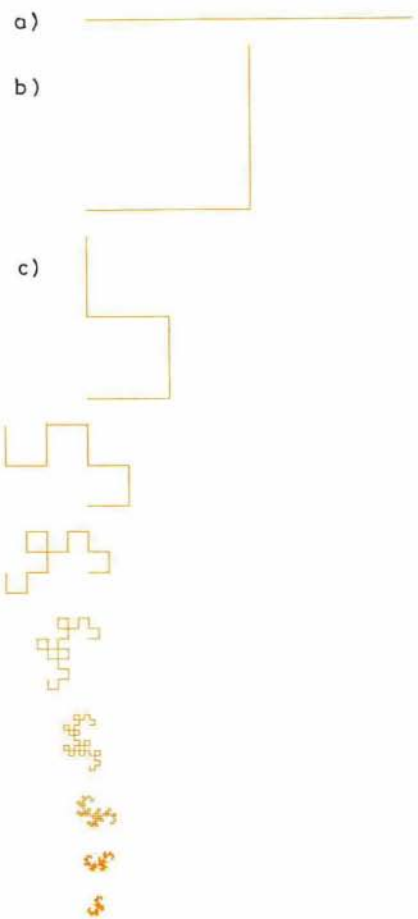
Peanovo krivuljo dobimo z naslednjim postopkom. V prvem koraku kvadratu narišemo diagonalo (slika 4a). V drugem koraku kvadrat razdelimo na devet enakih kvadratov in sedaj njim narišemo vse diagonale z eno potezo, tako da nikjer ne sekamo svoje poti (slika 4b). V tretjem koraku naredimo enako z vsakim od manjših kvadratov (slika 4c). Ker se na slikah 4 diagonale kvadratov v kotih stikajo, ni jasno, kako krivuljo narišemo. Jasno pa bo, če zavoje krivulje zaoblimo (slika 4cc). Krivulja, ki jo dobimo, če postopek ponovimo neskončnokrat, se imenuje *Peanova krivulja*. Opazimo, da se krivulja z vsakim naslednjim korakom vedno bolj gosto vije po površini, omejeni s kvadratom. Peanova krivulja je prostor *zapolnjujoča krivulja*, kar

pomeni, da gre skozi vsako točko površine, omejene s kvadratom. To krivuljo je v drugi polovici prejšnjega stoletja odkril italijanski matematik in logik Giuseppe Peano (glej 3. stran ovitka).

3. Zmajeva krivulja

Zmajevo krivuljo lahko le za nekaj prvih korakov razvoja dobimo s prepogibanjem papirja. Vzemimo dolg papirnati trak, ki predstavlja začetek razvoja *zmajeve krivulje* (slika 5a). Trak prepognimo po polovici in odprimo do pravega kota (slika 5b). Trak potem dvakrat zaporedoma prepognimo po polovici in ga odprimo do pravih kotov (slika 5c), trak trikrat zaporedoma prepognimo po polovici... Kdor ima izkušnje s prepogibanjem papirja ve, da se papir ne da prepogniti več kot sedemkrat, pa naj bo še tako tanek in dolg.

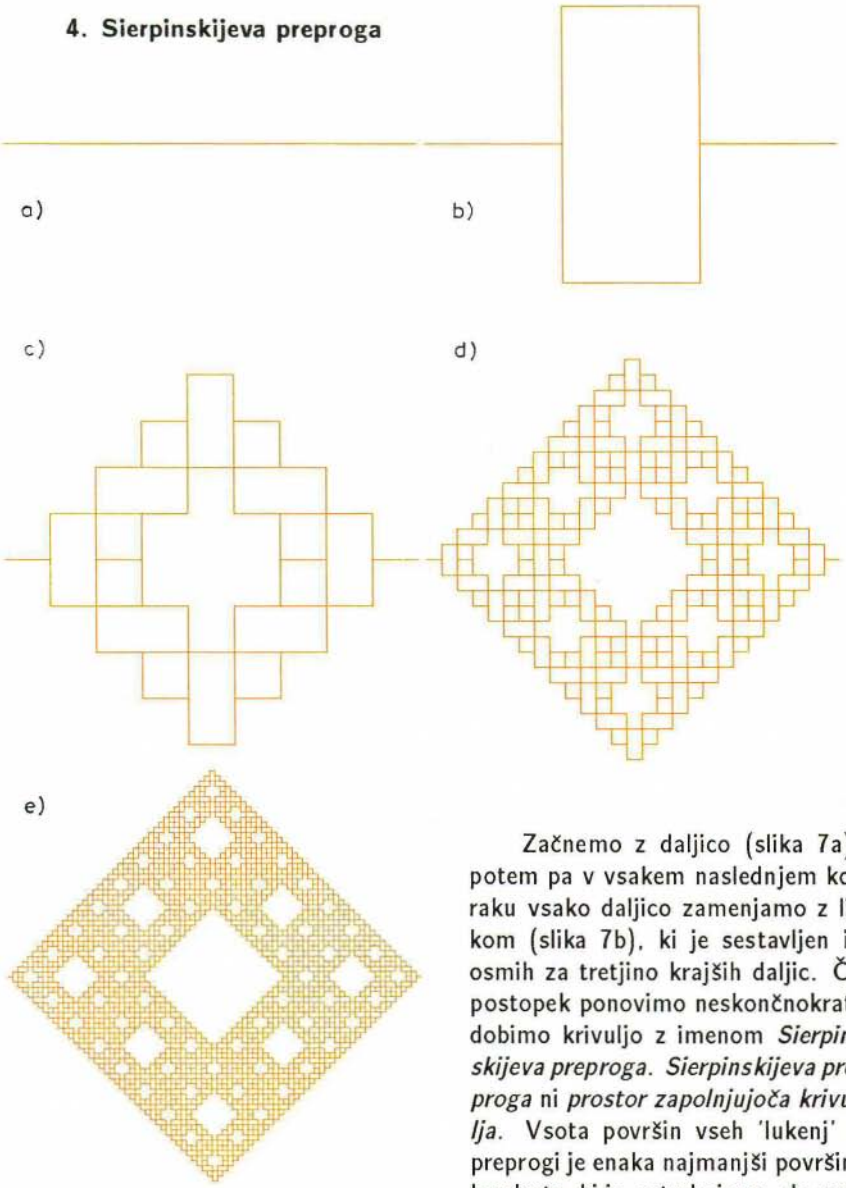
Naj bo papir na začetku še tako dolg, je prostor, ki ga zaseda zmajeva krivulja, po vsakem koraku manjši. Vendar pa je *zmajeva krivulja*, dobljena z neskončno mnogo prepogibanj, *prostor zapolnjujoča krivulja*, če na vsakem koraku dolžino papirja ustrezno podaljšamo. Če želimo, da je razdalja med začetkom in koncem *zmajeve krivulje* na vsakem koraku enaka, moramo krivuljo na vsakem koraku podaljšati za $\sqrt{r}$-krat. Štiri zmajeve krivulje lahko lepo zložimo skupaj, kot prikazuje slika 6 na 3. strani ovitka.



Slika 5

Prvi je zmajevo krivuljo opisal fizik John E. Heighway leta 1960 in po njem krivuljo imenujemo tudi *Heighwayev zmaj*.

4. Sierpinskijeva preproga

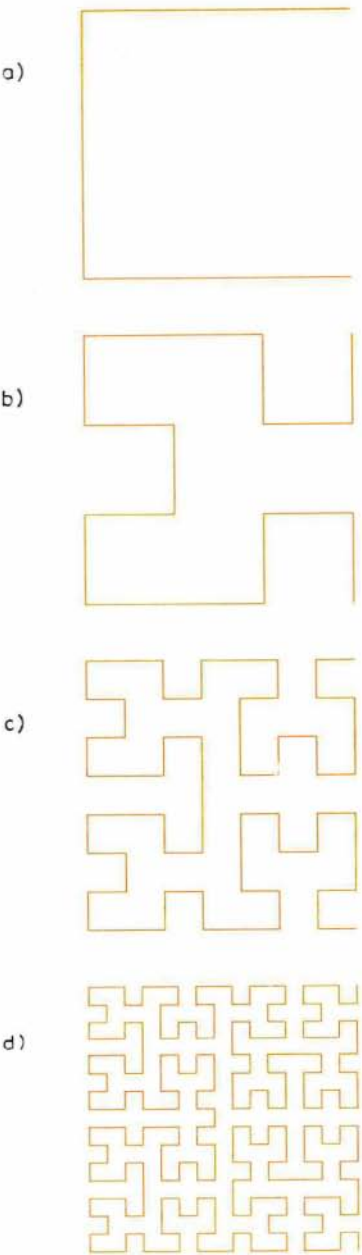


Slika 7

Začnemo z daljico (slika 7a), potem pa v vsakem naslednjem koraku vsako daljico zamenjamo z likom (slika 7b), ki je sestavljen iz osmih za tretjino krajših daljic. Če postopek ponovimo neskončnokrat, dobimo krivuljo z imenom *Sierpinskijeva preproga*. *Sierpinskijeva preproga* ni *prostor zapolnjujoča krivulja*. Vsota površin vseh 'lukenj' v preprogi je enaka najmanjši površini kvadrata, ki jo potrebujemo, da vanj vrišemo krivuljo. Torej sama krivulja ne prekrije nobene površine.

5. Hilbertova krivulja

Osnovni element te krivulje je lik na sliki 8a. V prvem koraku uporabimo štiri osnovne elemente in jih povežemo med seboj s tremi daljicami, kot prikazuje slika 8b. Za vsak naslednji korak uporabimo zadnjo dobljeno sliko kot osnovni element. Krivulja, ki jo dobimo po neskončno mnogo korakih, se imenuje *Hilbertova krivulja* in je *prostor zapolnjujoča krivulja*.



Slika 8

RISANJE KRIVULJ

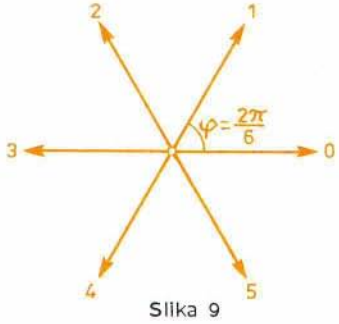
Z risanjem teh krivulj je podobno kot z risanjem premice, trikotnika, kroga,... Vse kar narišemo, pa naj bo še tako natančno, je le približek, pripomoček za nazorno predstavo ideje. Vsaka črta, ki jo narišemo, ima neko debelino, končno dolžino in tudi popolnoma ravna ni. Vse zgoraj opisane krivulje so dobljene s postopkom deljenja, ponovljenim neskončnokrat. Mi bomo risali le krivulje na začetnih stopnjah njihovega razvoja.

Generiranje krivulj si oglejmo kar na primeru Kochove snežinke, ki je zelo enostavna. Kateri so osnovni elementi potrebni za risanje Kochove snežinke? To je šest vektorjev določene dolžine, ki jih bomo oštevilčili od 0 do 5, kot prikazuje slika 9. Krivuljo, ki je sestavljena iz

teh vektorjev, lahko sedaj opišemo z nizom vektorjev. Npr.: niz 024 predstavlja enakostraničen trikotnik (slika 1a), niz 051021324354 pa *Kochovo snežinko* na prvi stopnji razvoja (slika 1b).

Videli smo, da se na vsakem naslednjem koraku vse stranice poligona nadomestijo z likom, sestavljenim iz štirih za tretjino krajših daljic. To lahko opišemo z naslednjimi pravili: vektor 0 se nadomesti z nizom vektorjev 0510, vektor 1 se nadomesti z nizom vektorjev 1021,... Podajmo ta pravila v tabeli:

celica	nadaljne delitve	predstavitev
0	0 5 1 0	0
1	1 0 2 1	1
2	2 1 3 2	2
3	3 2 4 3	3
4	4 3 5 4	4
5	5 4 0 5	5
6	0 2 4	-1



Slika 9

V prvem stolpcu so *celice*. Vsaka *celica* ima svoje ime, to je število. V zadnjem stolpcu je podana grafična *predstavitev*, interpretacija *celic*. *Celicam* 0 do 5 smo priredili vektorje s slike 9, celica 6 pa nima nobene grafične predstavitve in smo jo označili z -1. V srednjem delu tabele, imenovanem *nadaljnje delitve*, pa so podani nizi *celic*, ki nadomestijo celico v naslednjem koraku.

Celico, ki jo vzamemo kot začetno in iz nje potem v nadaljnih korakih razvijamo krivuljo v skladu s pravili v tabeli, imenujemo *rojstno celico*. Na sliki 10 je predstavljen razvoj *celice 6*. *Generacija* označuje, kolikokrat *celice* nadomestimo z njim ustreznimi nizi *celic*.

Oglejmo si sedaj program (str. 61) za risanje krivulj, ki so predstavljene s tabelo. Program je napisan v *turbo pascalu*, ne bo pa ga težko prilagoditi za katerikoli prevajalnik na drugem računalniku, ki omogoča risanje.

Ko program poženemo, vpišemo število smeri, t.j. vektorjev, ki so potrebni za risanje krivulje. Pri *Kochovi snežinki* je to število 6. Potem vpisujemo nadaljne delitve *celic* in za vsako *celico* niz končamo z -1. Za vsakim nizom *nadaljnih delitev* vnesemo še predstavitev celice. Vpisovanje v tabelo končamo z dvema -1, prvič ob prvi nadaljni delitvi celice, ki je ne potrebujemo več, in drugič ob predstavitvi te celice.

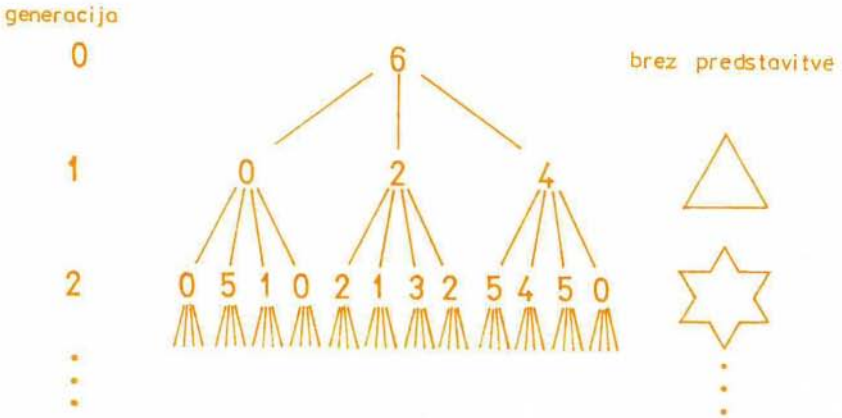
Po vnosu osnovnih podatkov krivulje v zanki spreminjamo naslednje parametre: dolžino celice, rojstno celico, generacijo in koordinati začetka

```
program Generiranje_Fraktalov;
uses Crt,Graph;
var nadaljneDelitve:array[0..50,0..50]of integer;
    predstavitev:array[0..50]of integer;
    stSmeri,dolzinaCelice,generacija,rojstnaCelica,zacetekX,zacetekY,i,j:integer;
    grDriver,grMode:integer;
    ch:char;


procedure narisi(celica:integer);
var dx,dy:integer;
begin
  if predstavitev[celica]>=0 then begin
    dx:=round(dolzinaCelice*cos(predstavitev[celica]mod
stSmeri*2*Pi/stSmeri));
    dy:=-round(dolzinaCelice*sin(predstavitev[celica]mod
stSmeri*2*Pi/stSmeri));
    if (predstavitev[celica]>=0)and(predstavitev[celica]<stSmeri)
      then LineRel(dx,dy)
      else MoveRel(dx,dy);
  end;
end; { narisi }


procedure krivulja(generacija,celica:integer);
var i:integer;
begin
  if generacija=0
    then narisi(celica)
    else begin
      i:=0;
      while nadaljneDelitve[celica,i]>=0 do begin
        krivulja(generacija-1,nadaljneDelitve[celica,i]);
        i:=i+1;
      end; { while }
    end; { else }
end; { krivulja }


begin { glavni }
  ClrScr;
  write('stevilo smeri = ');readln(stSmeri);
  i:=0;
  repeat
    j:=0;
    repeat
      write(j+1:3,'. nadaljna delitev celica ',i:3,' = ');readln(nadaljneDelitve[i,j]);
      j:=j+1;
    until nadaljneDelitve[i,j-1]<0;
    write('predstavitev celice ',i:3,' = ');readln(predstavitev[i]);
    i:=i+1;
  until nadaljneDelitve[i-1,0]<0;
  ch:=#0;
  repeat { spreminjanje parametrov in risanje }
    write('dolzina celice = ');readln(dolzinaCelice);
    write('rojstna celica = ');readln(rojstnaCelica);
    write('generacija = ');readln(generacija);
    write('zacetek risanja X koordinata = ');readln(zacetekX);
    write('zacetek risanja Y koordinata = ');readln(zacetekY);
    grDriver:=detect; { vstop v graficni nacin }
    InitGraph(grDriver,grMode,'');
    MoveTo(zacetekX,zacetekY);
    krivulja(generacija,rojstnaCelica);
    repeat until KeyPressed; { cakaj na pritisk katerekoli tipke }
    ch:=ReadKey; { preberi katera tipka je bila pritisnjena }
    CloseGraph; { nazaj v tekstovni nacin }
  until ch=#27; { koncaj ob pritisku na tipko ESC }
end. { glavni }
```



Slika 10

risanja. Po vsakem vnosu parametrov se krivulja nariše, in če želimo končati, pritisnemo tipko ESC, drugače pa katerokoli drugo tipko.

Celice imajo lahko naslednje grafične predstavitve p, če je število vseh smeri enako n: če je $0<p<n-1$, potem se nariše vektor pod kotom $p\frac{2\varphi}{n}$, če je $n<p<2n-1$, potem se le premaknemo v smeri $p\frac{2\varphi}{n}$ za dolžino vektorja, če pa je p negativno število, celica nima predstavitve.

Podajmo še tabele za ostale krivulje, ki smo jih spoznali:

Peanova krivulja
število smeri = 8
rojstna celica = katerakoli

celica	nadaljne delitve	predstavitev
0	010323010	1
1	121030121	3
2	232101232	5
3	303212303	7

Zmajeva krivulja
število smeri = 4
rojstna celica = katerakoli

celica	nadaljne delitve	predstavitev
0	01	0
1	21	1
2	23	2
3	03	3

Sierpinskijeva preproga
število smeri = 4
rojstna celica =
= katerakoli od 0 do 3

celica	nadaljne delitve	predstavitev
0	010363010	0
1	121070121	1
2	232141232	2
3	303252303	3
4	444	4
5	555	5
6	666	6
7	777	7

Hilbertova krivulja
število smeri = 4
rojstna celica = 8 ali 9

celica	nadaljne delitve	predstavitev
0	1034	0
1	0125	1
2	0126	2
3	1037	3
4	6740	0
5	7651	1
6	7652	2
7	6743	3
8	1039	-1
9	6748	-1

Napotki za konstrukcijo lastnih krivulj

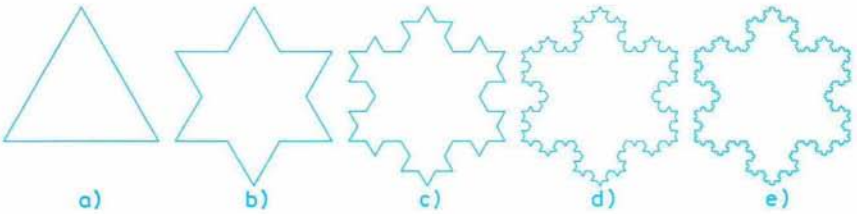
Najenostavnejši način za konstrukcijo lastne krivulje je seveda, da si nadaljne delitve v tabeli kar izmislimo. Če nam je krivulja potem všeč, jo lahko poskušamo še izboljšati. Drugemu načinu bomo rekli metoda iniciatorja in generatorja, pri kateri je iniciator nekakšno seme, iz katerega se razvija krivulja po pravilih, ki jih podaja generator. Po tej metodi je dobljena Kochova snežinka, Peanova krivulja in Sierpinskijeva preproga. Pri Kochovi snežinki je iniciator enakostraničen trikotnik (slika 1a), generator pa Davidova zvezda (slika 1b). Pri Peanovi krivulji je iniciator diagonala kvadrata (slika 4a), generator pa povezane diagonale devetih kvadratov (slika 4b). Pri Sierpinskijevi preprogi je iniciator daljica (slika 7a), generator pa lik

na sliki 7b. Enostavno je če vzamemo za iniciator daljico, za generator pa si izmislimo lik. Vsakemu vektorju potem priredimo celico, ki se v naslednjem koraku nadomesti z ustrezno orientiranim likom. Konstrukcija krivulj bo šla seveda tembolj od rok, čim bolj se bomo poglobili v delovanja samega postopka risanja.

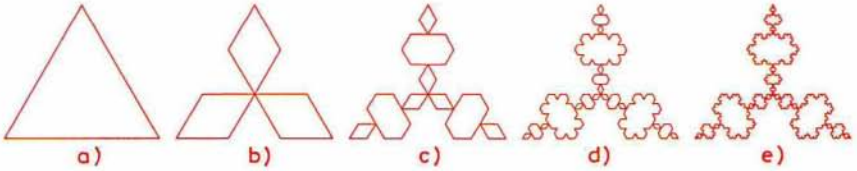
Ideje za razširitev programa

Vsako krivuljo lahko shranimo na disk v obliki datoteke. V datoteko shranimo vse parametre potrebne za risanje krivulje, število smeri, celotno tabelo, rojstno celico, dolžino celice, koordinate začetka risanja krivulje. Tako si prihranimo veliko dela s tipkanjem tabel v program. Ni nujno, da so celice predstavljene z različno usmerjenimi, enako dolgimi daljicami. Predstavitev celic bi lahko bila tudi tridimenzionalna.

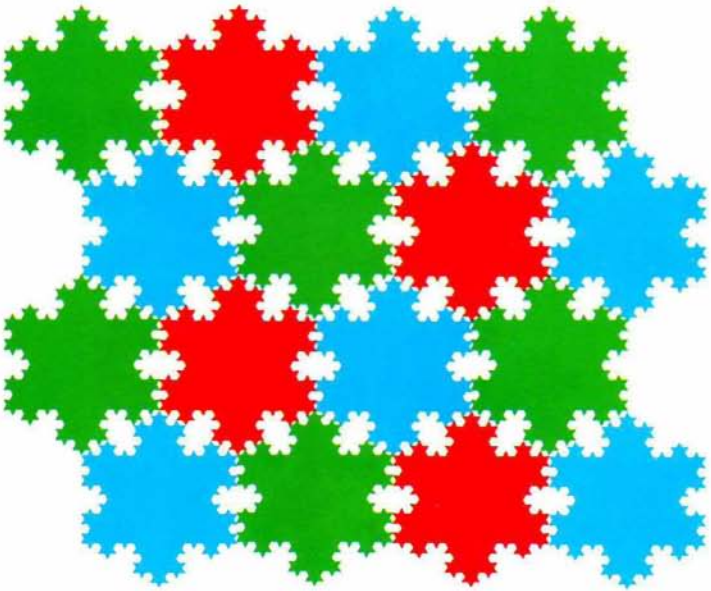
Ciril Pezdir



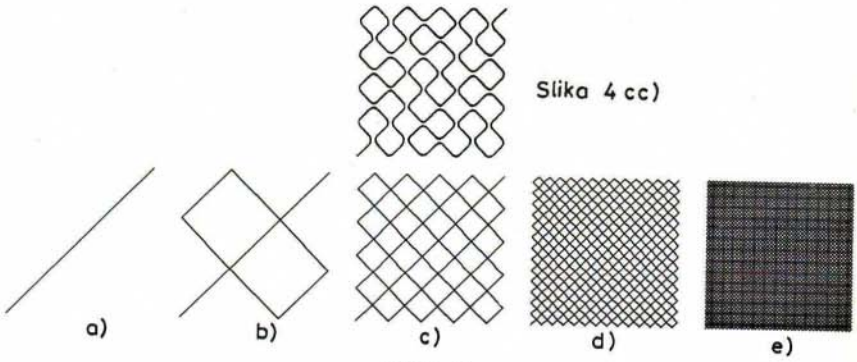
Slika 1



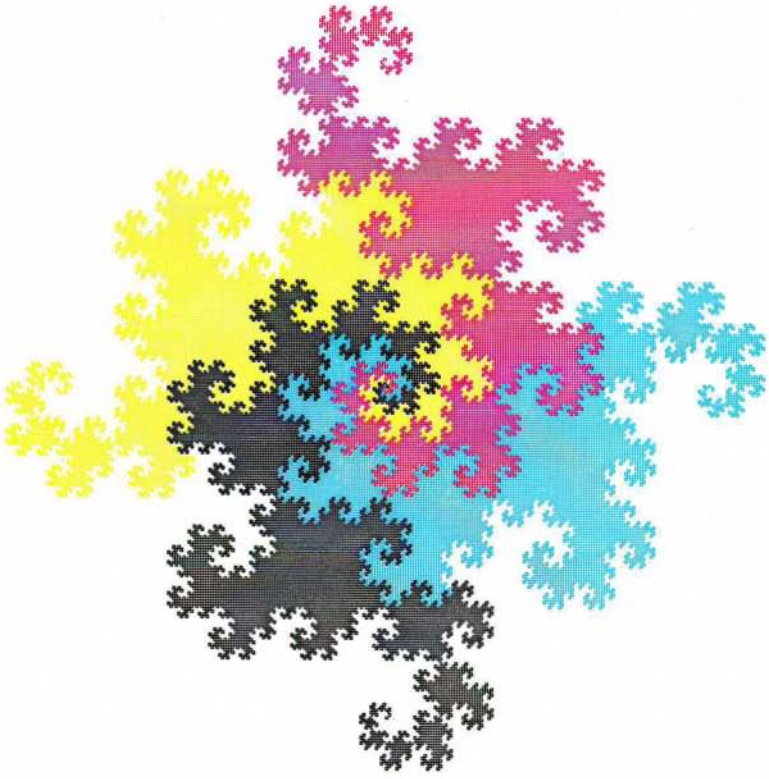
Slika 2



Slika 3



Slika 4



Slika 6