1703

Гимназија Јован Јовановић Змај

Colegiul Național „Constantin Diaconovici Loga" din Timișoara

Co-funded by the Erasmus+ Programme of the European Union

Project title: **Computer supported Math teaching,** COMPMATH
Ref. No.: KA210-SCH-ABDA6A90
Action Type: KA210-SCH - Small-scale partnerships in school education

Author:

DR. ING. ALEXANDRU IOVANOVICI(UNIVERSITATEA POLITEHNICA TIMISOARA)
DR. BOAR RENATA-GRAZIELA(COLEGIUL NATIONAL "CONSTANTIN DIACONOVICI LOGA")

Aplicatii ale recursivitatii pentru generarea de fractali

2023

Contents

1. Recursivitatea

Recursivitatea în programare se referă la abilitatea unei funcții de a se apela pe sine în timpul executării sale. Este ca și cum o funcție rezolvă o problemă prin împărțirea ei în cazuri mai mici ale aceleiași probleme, până când se ajunge la o situație de bază care poate fi rezolvată direct, fără alte apeluri recursive.

Există două componente importante ale unei funcții recursive:

1. **Cazul de bază:** Este condiția care determină oprirea recursivității și rezolvarea efectivă a problemei. Este esențial să existe un caz de bază bine definit pentru ca recursivitatea să nu continue la infinit.

2. **Cazul recursiv:** Acesta este pasul în care funcția se apelează pe sine pentru a rezolva o problemă mai mică, similară problemei inițiale. Această apelare recursivă trebuie să se apropie de cazul de bază pentru a asigura oprirea.

De exemplu, luând în considerare calculul factorialului unui număr, puteți avea o funcție factorial care se apelează pe sine în mod recursiv:

```python
def factorial(n):
    # Cazul de bază: factorialul lui 0 sau 1 este 1
    if n == 0 or n == 1:
        return 1
    else:
        # Cazul recursiv: factorialul lui n = n * factorialul(n - 1)
        return n * factorial(n - 1)
```

În acest caz, când se apelează `factorial(5)`, funcția va apela recursiv `factorial(4)`, `factorial(3)`, `factorial(2)`, `factorial(1)` și `factorial(0)`, rezultând într-o serie de apeluri până când se ajunge la cazurile de bază și se întorc valorile, rezolvând în cele din urmă calculul factorialului lui 5.

Recursivitatea poate fi extrem de puternică și elegantă pentru rezolvarea unor probleme complexe, dar necesită grijă pentru a evita situațiile în care poate duce la stivă de apeluri prea mare (depășind limitele de stivă) sau la o performanță scăzută în comparație cu soluții iterative.

2. Fractali

Un fractal este o structură geometrică complexă care poate fi divizată în părți mai mici, fiecare dintre ele fiind o copie la scară redusă a întregului. Aceste forme fractali au proprietăți matematice fascinante, fiind caracterizate prin autosimilaritate, adică aspectul lor rămâne neschimbat indiferent de nivelul la care sunt examinate.

Caracteristicile principale ale unui fractal sunt:

1. **Autosimilaritatea:** Partea individuală a unui fractal seamănă cu întregul fractal într-un mod auto-replicativ. Cu alte cuvinte, când te uiți la o porțiune mică a unui fractal, aceasta arată asemănător cu întregul obiect.

2. **Complexitatea infinită într-o scară finită:** Fractalile sunt detaliate la infinit, ceea ce înseamnă că, chiar și când sunt examinate la o scară mică, ele păstrează detalii și modele complexe.

3. **Auto-iterația:** Fractalile pot fi generate folosind procese iterative simple, aplicând aceleași reguli sau operații de bază repetate pentru a crea forme mai complexe.

Un exemplu clasic de fractal este setul lui Mandelbrot, care este generat prin aplicarea unei simple relații matematice asupra punctelor într-un plan complex. Acest set are proprietăți fractali impresionante, iar la o scală mare, are un aspect similar cu întregul său chiar și când te uiți la porțiuni mici ale acestuia.

Fractalile sunt întâlnite într-o varietate de domenii, inclusiv artă, știință, grafică computerizată și chiar în analiza datelor și modelarea fenomenelor naturale complexe, datorită naturii lor complexe și frumuseții lor estetice și matematice.

Recursivitatea și fractalii au o legătură strânsă, în mare parte din cauza modului în care fractalii sunt construiți și generați. Recursivitatea este adesea folosită pentru a crea și a genera structurile fractali.

Fractalile sunt adesea definite și generate folosind procese recursive sau iterații repetitive. De exemplu, să luăm în considerare construcția unui fractal clasic, cum ar fi setul lui Mandelbrot. Acesta se bazează pe aplicarea unei relații matematice recursive pentru fiecare punct dintr-un plan complex.

În general, generarea unui fractal implică urmărirea unui proces iterativ sau recursive care se repetă la o scară mai mică și mai mică, adăugând detalii și structură pe măsură ce iterează. Această caracteristică de a genera detalii la o scară mică și de a avea autosimilaritate la diferite niveluri de magnificație este ceea ce face ca fractalii să fie atât de legați de conceptul de recursivitate.

Recursivitatea oferă un cadru conceptual și procedural puternic pentru a genera fractali, deoarece permite definirea și construcția acestora prin subdiviziuni repetitive și detaliate, adesea într-un mod care adaugă complexitate și frumusețe estetică la fiecare iterație sau recursiune. Astfel, capacitatea de a apela o funcție sau de a executa un proces în mod repetat și detaliat, adesea pe baza unor reguli simple, este esențială în crearea și înțelegerea structurilor fractali.

3. Dimensiunea fractala

Dimensiunea fractală este o măsură a complexității geometrice a unui obiect fractal. În timp ce dimensiunea obiectelor tradiționale (cum ar fi dreptunghiurile, cercurile etc.) este o valoare întreagă (de exemplu, o linie are dimensiunea 1, un plan are dimensiunea 2), fractalii prezintă dimensiuni non-integer sau fractale.

Această diferență în dimensiune se datorează naturii lor fractale, care sunt caracterizate prin detalii infinit de fine și autosimilaritate la diferite niveluri de magnificație. Pentru a măsura această complexitate, s-au dezvoltat mai multe concepte de dimensiune fractală, cele mai comune fiind:

1. **Dimensiunea Hausdorff-Besicovitch:** Este o măsură a gradului de umplere a unui set fractal într-un spațiu metric. Această dimensiune poate fi fracționară și este calculată folosind conceptul de acoperire a setului cu mulțimi mai mici.

2. **Dimensiunea box-counting:** Este o metodă comună pentru a estima dimensiunea fractală a unui set. Se bazează pe împărțirea setului fractal în pătrate mai mici (sau cuburi în spații tridimensionale) și determinarea numărului minim de astfel de pătrate necesare pentru a acoperi întregul set la diferite niveluri de magnificație.

Dimensiunea fractală este importantă în studiul fractalilor și în înțelegerea formelor complexe din natură și din diverse domenii științifice. Ea oferă o modalitate de a cuantifica și de a caracteriza aceste structuri intricate și autosimilare care nu pot fi complet descrise folosind dimensiunile tradiționale întregi.

Fractalul Koch

Fractalul Koch este unul dintre cele mai cunoscute fractali și poate fi generat prin aplicarea unor reguli simple pe segmente de linie. Voi folosi Python pentru a crea un program recursiv care să genereze fractalul Koch prin divizarea și modificarea laturilor unui triunghi.

```
import turtle
def fractal_koch(t, order, size):
    if order == 0:
        t.forward(size)
    else:
        for angle in [60, -120, 60, 0]:
            fractal_koch(t, order - 1, size / 3)
            t.left(angle)
def draw_koch_snowflake(t, order, size):
    for _ in range(3):
        fractal_koch(t, order, size)
        t.right(120)
# Set up the screen and turtle
screen = turtle.Screen()
screen.title("Fractal Koch")
screen.bgcolor("white")
pen = turtle.Turtle()
pen.color("blue")
pen.speed(0)
pen.penup()
pen.goto(-150, 90)
pen.pendown()
# Draw the Koch snowflake (equilateral triangle)
draw_koch_snowflake(pen, 4, 300)

# Hide the turtle and display the result
pen.hideturtle()
turtle.done()
```

Acest program folosește modulul turtle din Python pentru a desena fractalul Koch. Funcția fractal_koch generează recursiv liniile fractalului Koch în funcție de nivelul de adâncime (order). Funcția draw_koch_snowflake desenează un triunghi echilateral folosind linii fractal Koch.

Poți ajusta nivelul de adâncime order pentru a obține un fractal Koch cu mai multe sau mai puține detalii. Mai multe nivele vor adăuga mai multe iterații și detalii la fractal.

Aplicatii:

- modificati secvența încât sa permită schimbarea numărului de “niveluri” de adâncime a fractalului;
- Adaptați secvența încât sa permită generarea fractalului Sierpinski