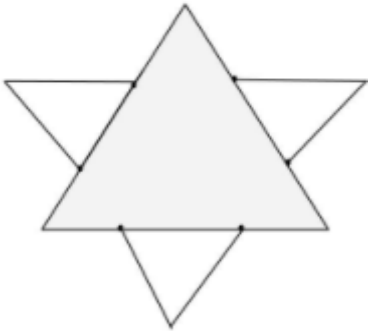
Aim: To implement Fractal (Koch Curve).

Objective:
A Koch curve is a fractal curve that can be constructed by taking a straight-line segment and replacing it with a pattern of multiple line segments. Then the line segments in that pattern are replaced by the same pattern.

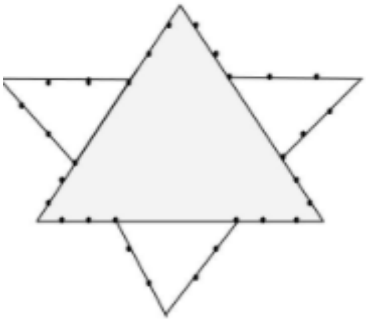
Theory:
1) Draw an equilateral triangle.

2) Divide each side in three equal parts.

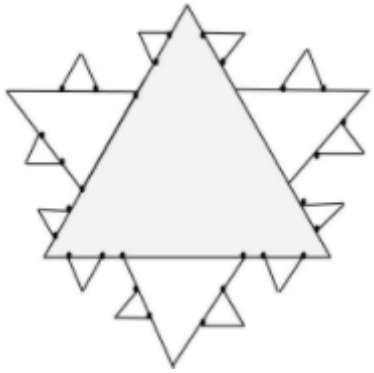
3) Draw an equilateral triangle on each middle part. Measure the length of the middle third to know the length of the sides of these new triangles.



4) Divide each outer side into thirds. You can see the 2nd generation of triangles covers a bit of the first. These three line-segments shouldn't be parted in three.



5) Draw an equilateral triangle on each middle part.

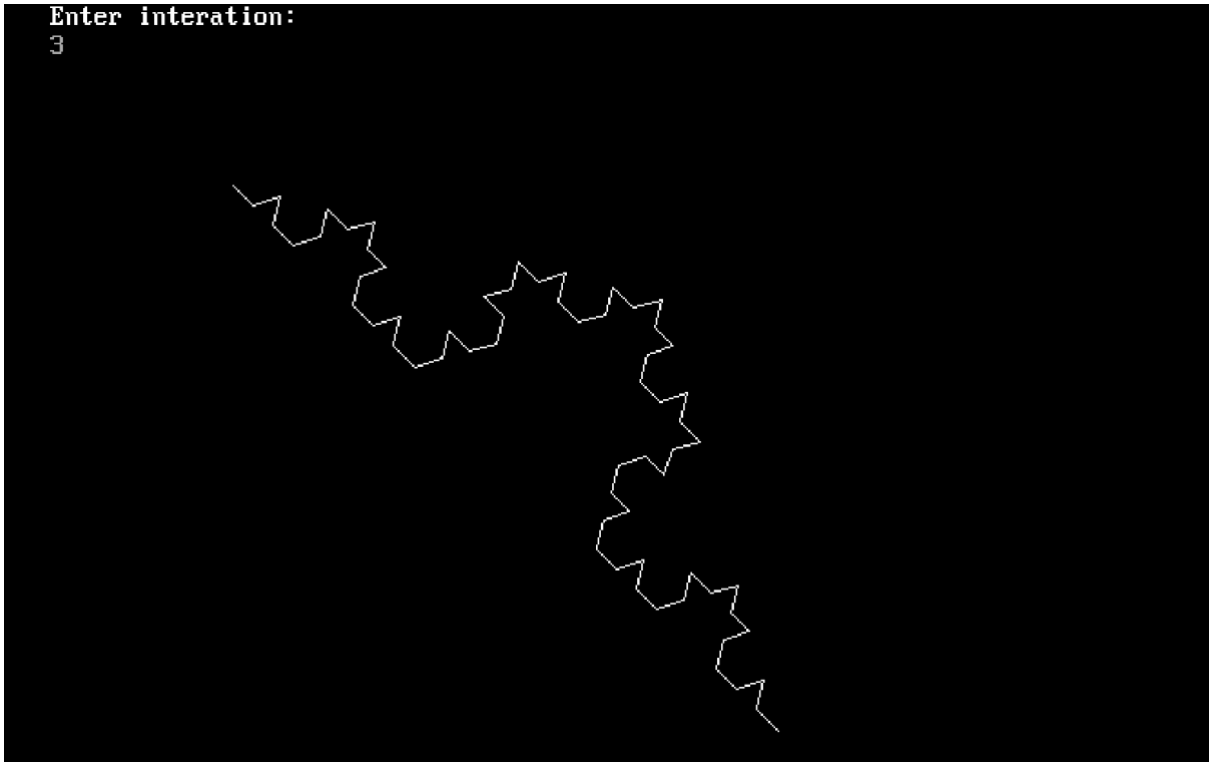


Program:

```
#include<stdio.h>
#include<conio.h>
#include<graphics.h>
#include<math.h>
void koch(int,int,int,int,int);
void main()
{
  int gd=DETECT,gm;
  int i,x1=100,y1=100,x2=400,y2=400,a;
  initgraph(&gd,&gm,"C:\\TURBOC3\\BGI");
  printf("Enter interation:\n");
  scanf("%d",&a);
  for(i=0;i<a;i++)
  {
  koch(x1,y1,x2,y2,a);
  getch();
  }
  closegraph();
}

void koch(int x1,int y1,int x2,int y2,int itr)
{
    float angle=60*3.14/180;
    int x3=(2*x1+x2)/3;
    int y3=(2*y1+y2)/3;

    int x4=(x1+2*x2)/3;
    int y4=(y1+2*y2)/3;

    int x=x3+(x4-x3)*cos(angle)+(y4-y3)*sin(angle);
    int y=y3-(x4-x3)*sin(angle)+(y4-y3)*cos(angle);
```

```
if(itr>0)
{
 koch(x1,y1,x3,y3,itr-1);
 koch(x3,y3,x,y,itr-1);
 koch(x,y,x4,y4,itr-1);
 koch(x4,y4,x2,y2,itr-1);
 }
 else
 {
 line(x1,y1,x2,y2);
 }
 }
```

Output:



Conclusion:-

1. Difference from Bezier Curve:-We cannot extend Bezier curve. The koch curve is a simple fractal that creates a pretty snowflake-line object.
2. Application:-Fractal behavior manifests itself in nature in everything from broccoli to coastlines.