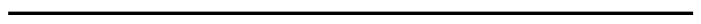
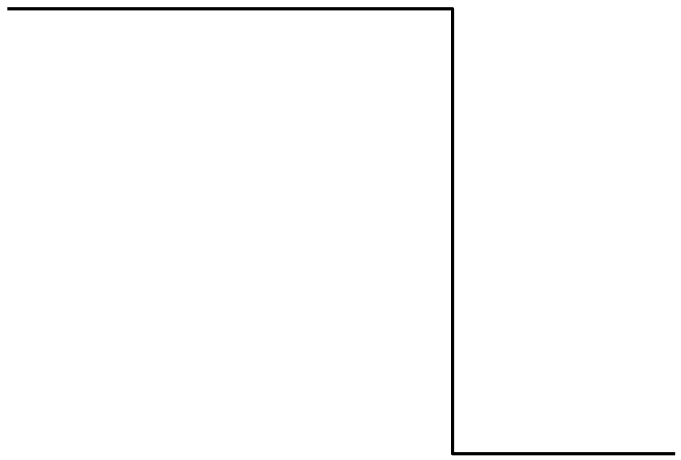
```
> ReadFromWeb :=proc(URL :: string, {printfile :: truefalse := false})
   local n, m, status, webfile, headers;
   status, webfile, headers := HTTP[Get](URL) :
   if ( HTTP[Code](status) ≠ "OK") then
      error(HTTP[Code](status), URL);
   fi;
   # now read the web page
   n := 0 :
   while (n < length(webfile)) do
     m := n;
     parse(webfile, statement, lastread='n', offset=n);
     if (printfile) then printf("%s", webfile[m + 1 ..n]); fi;
   od:
  end:
```

```
> ReadFromWeb("http://www.math.sunysb.edu/~scott/mat331.
  spr13/problems/turtle.txt");
> TurtleCmd('FFFFF');
```

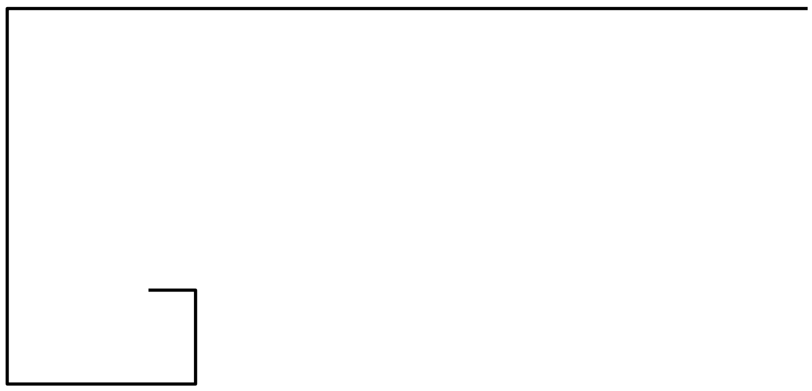


Language:
F   move forward one step
B move back one step
R turn right (don't move)
L turn left (don't move)

```
> TurtleCmd('FFRFFLF');
```

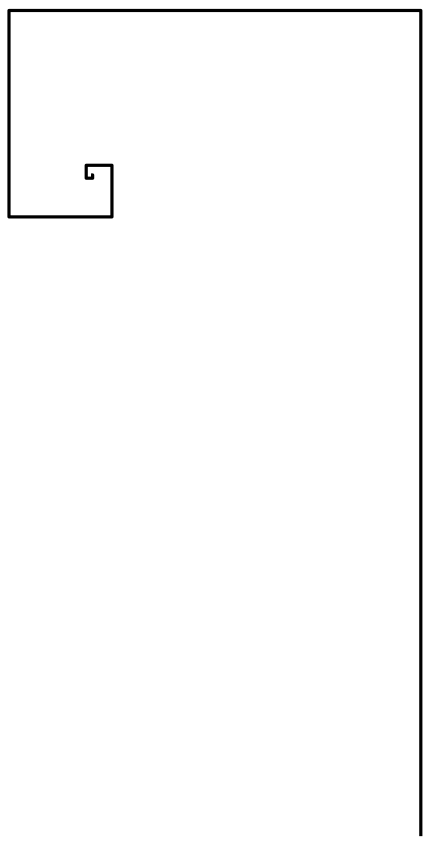


```
> TurtleCmd( "FRFFRFFFFRFFFFFFFFRFFFFFFFFFFFFFFFF");
```



Two new commands:
  G - grow (double step size)   /   S - shrink (halves stepsize)

```
> TurtleCmd("FRGFRGFRGFRGFRGFRGFRGFRGFRGFR");
```



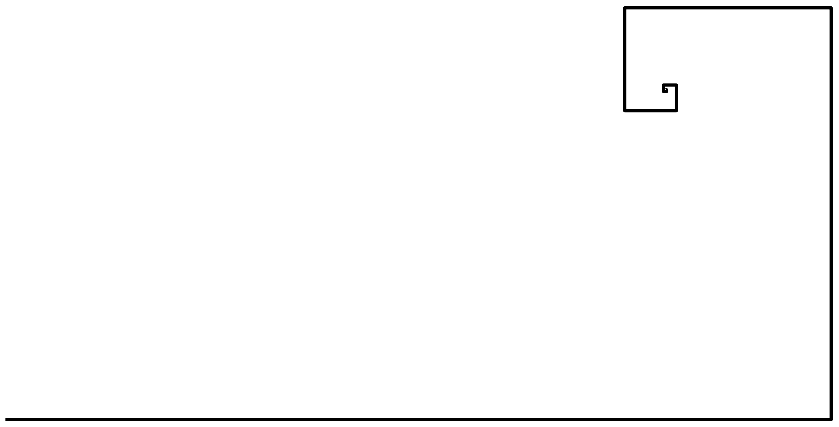
```
> Spiral:= n -> cat( seq("GFR", i=1..n));
```

$$Spiral := n \rightarrow cat(seq(\text{"GFR"}, i = 1 ..n)) \tag{1}$$

```
> Spiral(5);
```

"GFRGFRGFRGFRGFR" **(2)**

```
> TurtleCmd(Spiral(15));
```



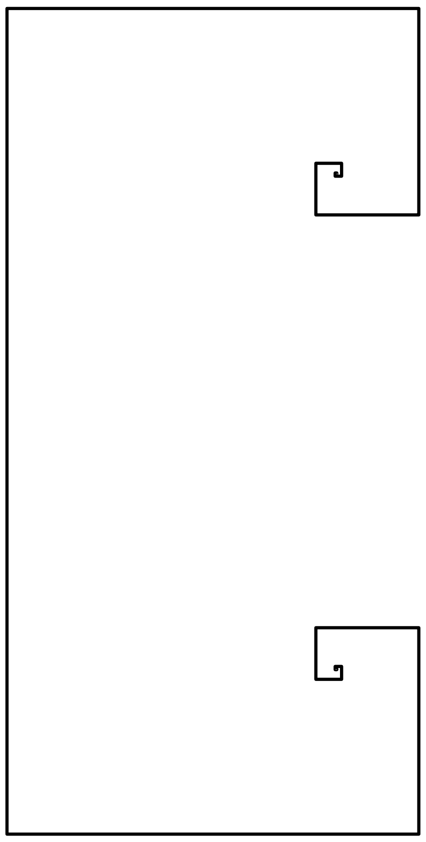
```
> Spiral:= (n,g) -> cat( seq( cat("FR",g), i=1..n));
```

$$Spiral := (n, g) \rightarrow cat(seq(cat(\text{"FR"}, g), i = 1 .. n)) \tag{3}$$

```
> Spiral(5,S);
```

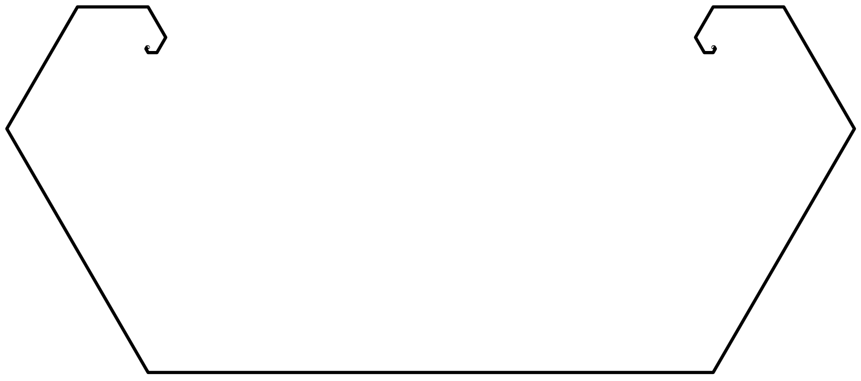
"FRSFRSFRSFRSFRS" **(4)**

```
> TurtleCmd( cat(Spiral(15,"G"), Spiral(15,"S")));
```



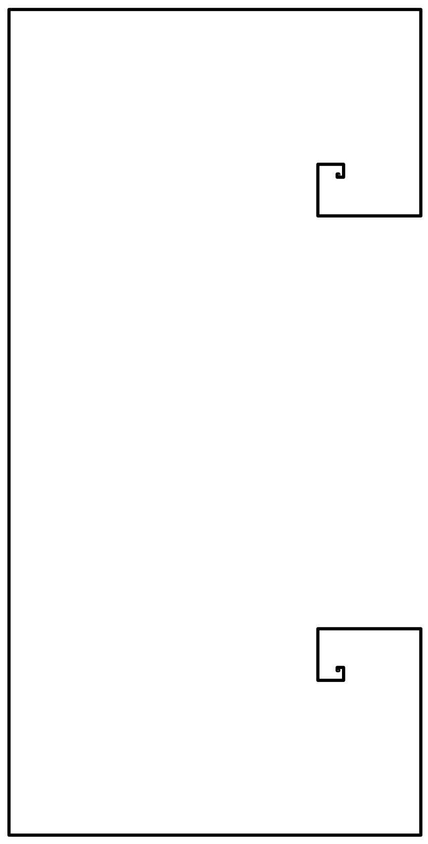
Change the angle
   SetTurtleAngle(theta) --- turtle turns by theta degrees

```
> SetTurtleAngle(60);
  TurtleCmd( cat(Spiral(15,"G"), Spiral(15,"S")));
```



ResetTurtle puts stuff back

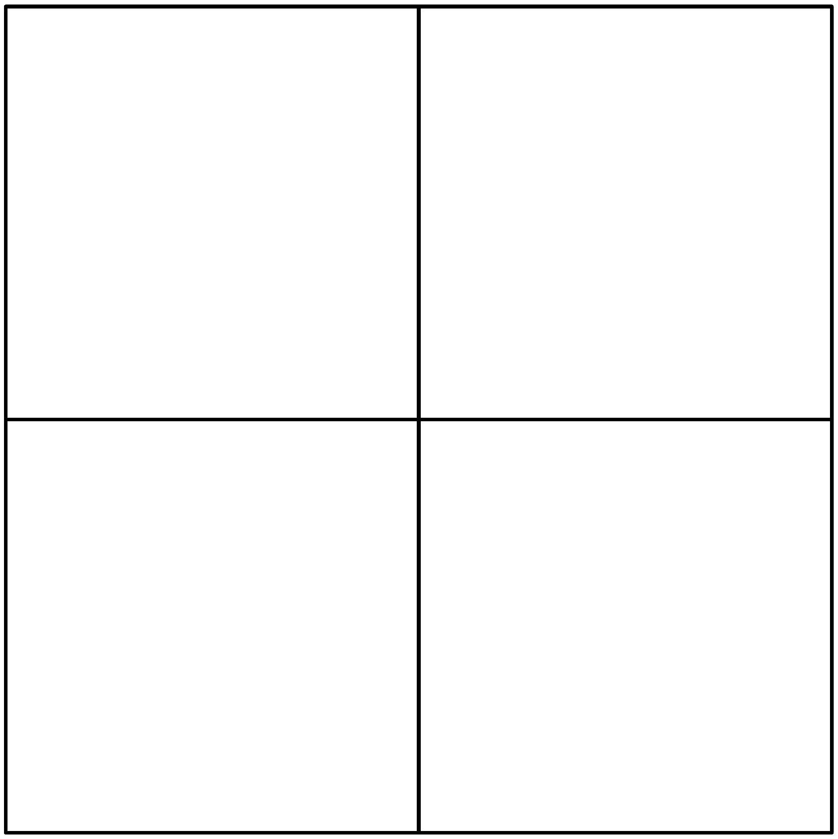
```
> ResetTurtle();
  TurtleCmd( cat(Spiral(15,"G"), Spiral(15,"S")));
```



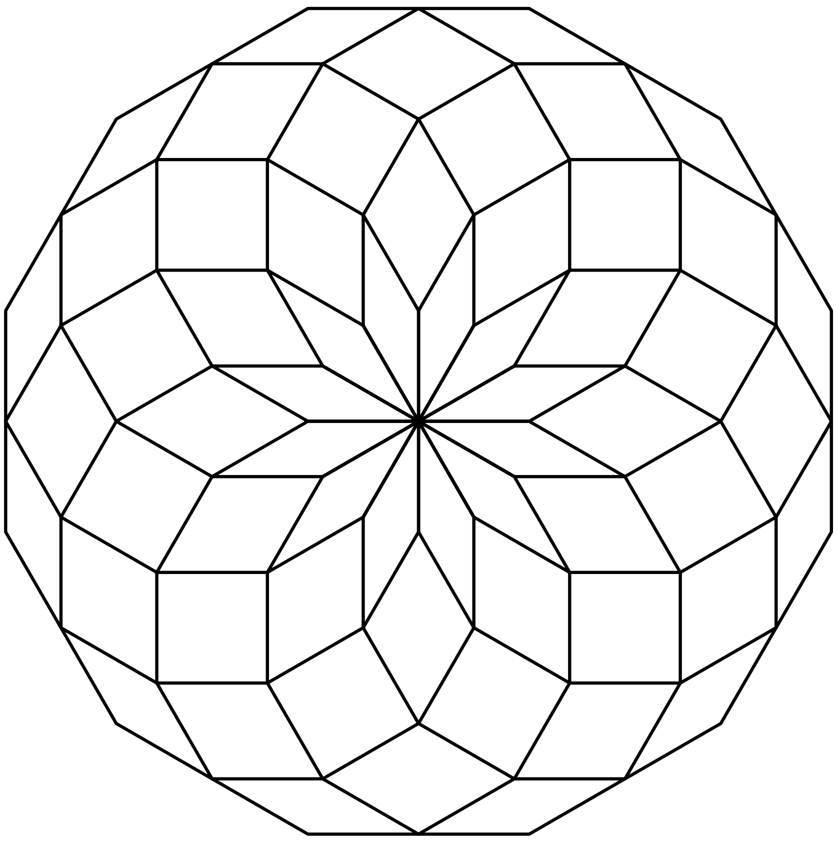
```
> cat("L",seq("FL",i=1..12));
```

```
                    "LFLFLFLFLFLFLFLFLFLFLFLFL"                    (5)
```

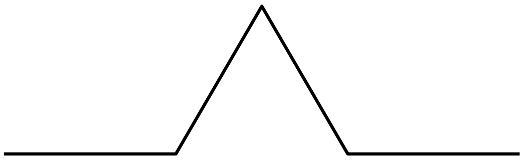
```
> TurtleCmd(cat(seq(cat("L",seq("FL",i=1..12)),j=1..12)));
```



```
> SetTurtleAngle(30); TurtleCmd(cat(seq(cat("L",seq("FL",i=1..12)),
  j=1..12)));
```

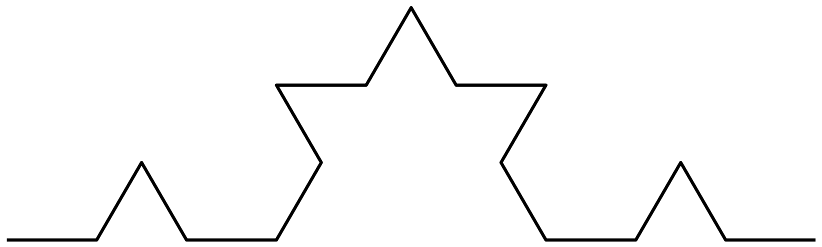


```
> ResetTurtle();
> SetTurtleAngle(60);
> TurtleCmd("FLFRRFLF");
```



Idea: replace each F with FLFRRFLF

```
> TurtleCmd("FLFRRFLFLFLFRRFLFRRFLFRRFLFLFLFRRFLF" ) ;
```



Write a program to do this.

Side trip: recursion

   Familiar thing: n! is   n*(n-1)!, with 1! = 1.

```
> 5!;
```

                                        120   (6)

```
> Fact:=proc(n::posint)
      if (n>1) then
        return(n*Fact(n-1));
      else
        return(1);
      fi;
  end:
> Fact(5);
```

                                        120   (7)

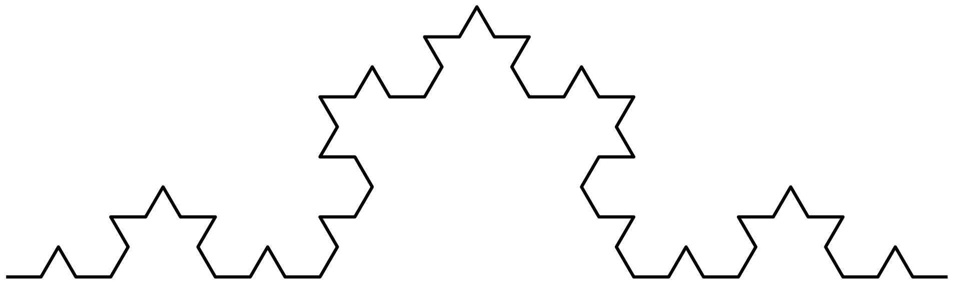
```
> debug(Fact):
  Fact(5);
{--> enter Fact, args = 5
{--> enter Fact, args = 4
{--> enter Fact, args = 3
{--> enter Fact, args = 2
{--> enter Fact, args = 1
<-- exit Fact (now in Fact) = 1}
<-- exit Fact (now in Fact) = 2}
<-- exit Fact (now in Fact) = 6}
<-- exit Fact (now in Fact) = 24}
<-- exit Fact (now at top level) = 120}
```

                                        120   (8)

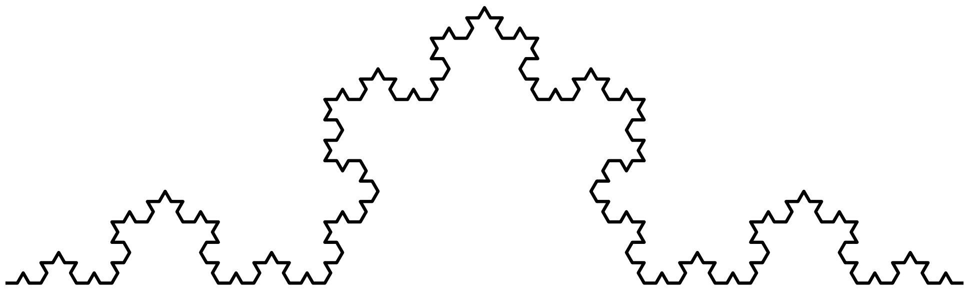
Idea is: [thing]L[thing]RR[thing]L[thing]
where [thing] is either F or [thing]L[thing]RR[thing]L[thing]

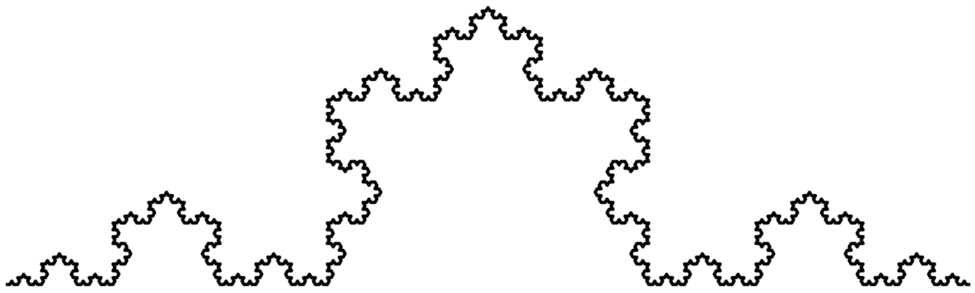
```
> Koch:= proc(n::posint)
     if (n=1) then return("F"); fi;
```

```
    ## now n>1
    return(  cat( Koch(n-1), "L", Koch(n-1), "RR",
                  Koch(n-1), "L", Koch(n-1)));
  end:
> Koch(1);
                              "F"                                    (9)
> Koch(3);
          "FLFRRFLFLFLFLFRRFLFRRFLFRRFLFLFLFLFRRFLF"                (10)
> TurtleCmd(Koch(4));
```

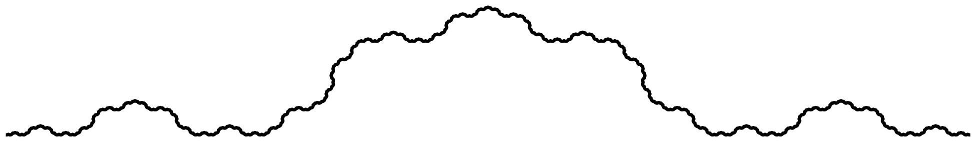


```
> TurtleCmd(Koch(5));TurtleCmd(Koch(6));
```

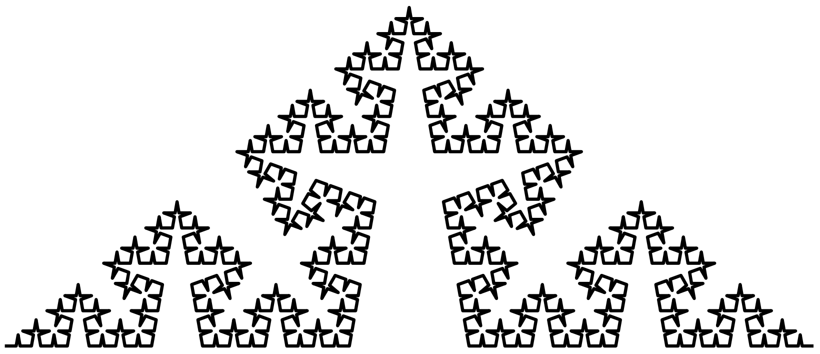




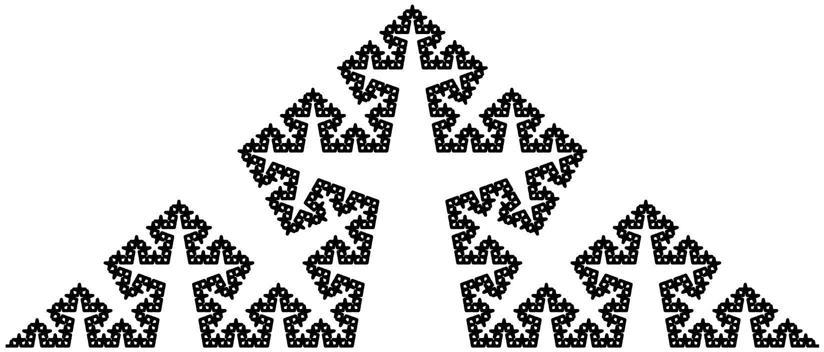
```
> SetTurtleAngle(30);TurtleCmd(Koch(6));
```



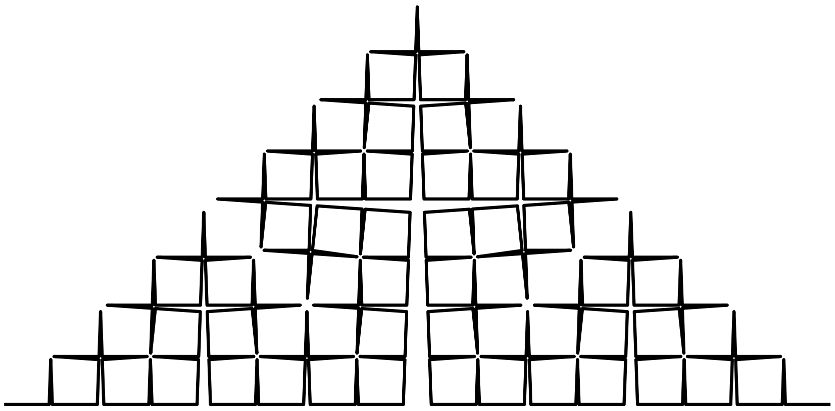
```
> SetTurtleAngle(80);TurtleCmd(Koch(6));
```



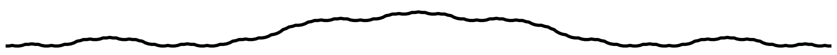
```
> SetTurtleAngle(80);TurtleCmd(Koch(7));
```



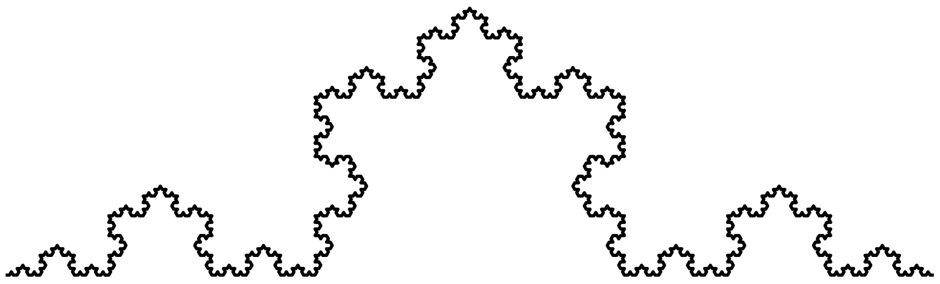
```
> SetTurtleAngle(88);TurtleCmd(Koch(5));
```



```
> SetTurtleAngle(10);TurtleCmd(Koch(5));
```



```
> SetTurtleAngle(60);TurtleCmd(Koch(6));
```



```
> SetTurtleAngle(90);TurtleCmd(Koch(7));
```