```
>
> 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
    end if;
    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]) end if
    od:
   end proc:
> ReadFromWeb("http://www.math.sunysb.edu/~scott/mat331.
  spr13/problems/turtle.txt");
> ResetTurtleCmd;
```

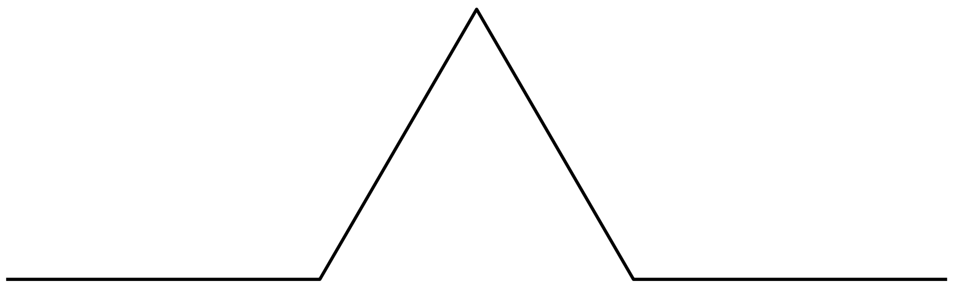
ResetTurtleCmd **(1)**

```
> 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(2);Koch(3);
```

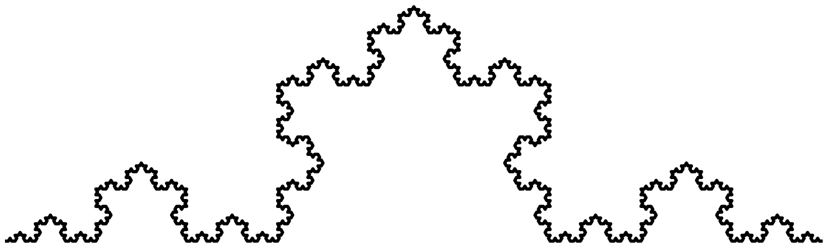
" FLFRRFLF "

" FLFRRFLF L FLFRRFLF RR FLFRRFLF L FLFRRFLF " **(2)**

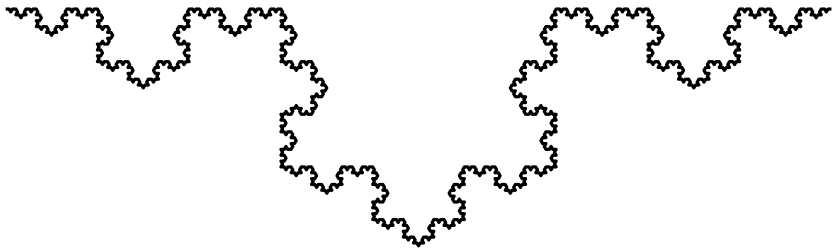
```
> SetTurtleAngle(60):TurtleCmd(Koch(2));
```



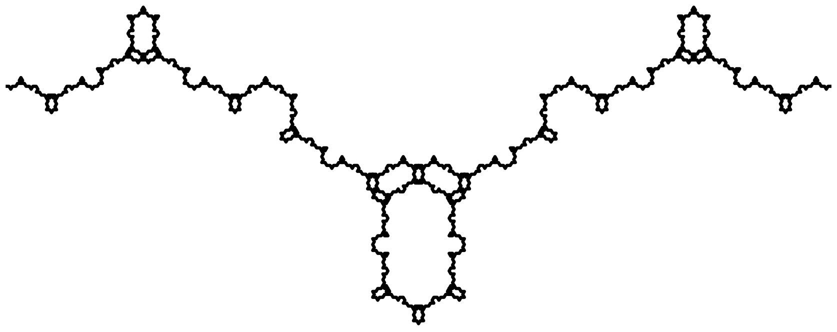
```
> TurtleCmd(Koch(6));
```



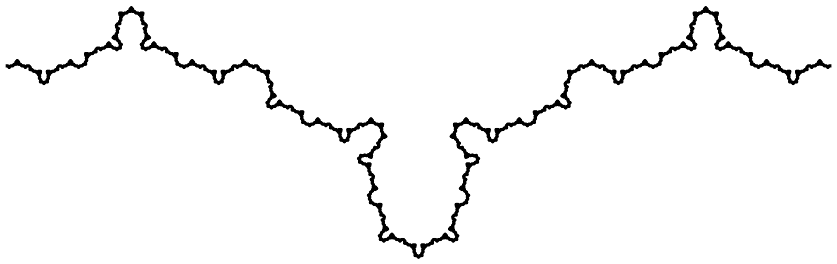
```
> DKoch:= proc(n::posint)
    if (n=1) then return("F"); fi;
    ## now n>1
    return(  cat( " ",DKoch(n-1), "R", DKoch(n-1), "LL",
                 DKoch(n-1), "R", DKoch(n-1), " "));
  end:
> TurtleCmd(DKoch(6));
```



```
> UDKoch:= proc(n::posint)
    if (n=1) then return("F"); fi;
    ## now n>1
    if (n mod 2 = 0) then
       return(  cat( " ",UDKoch(n-1), "R", UDKoch(n-1), "LL",
                         UDKoch(n-1), "R", UDKoch(n-1), " "));
    else
      return(  cat( " ",UDKoch(n-1), "L", UDKoch(n-1), "RR",
                         UDKoch(n-1), "L", UDKoch(n-1), " "));
     fi;
  end:
> SetTurtleAngle(60);TurtleCmd(UDKoch(6));
```

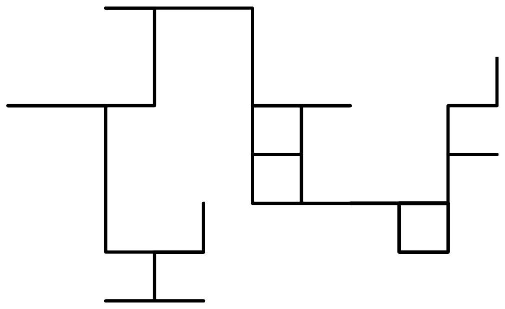


```
> SetTurtleAngle(50);TurtleCmd(UDKoch(6));
```

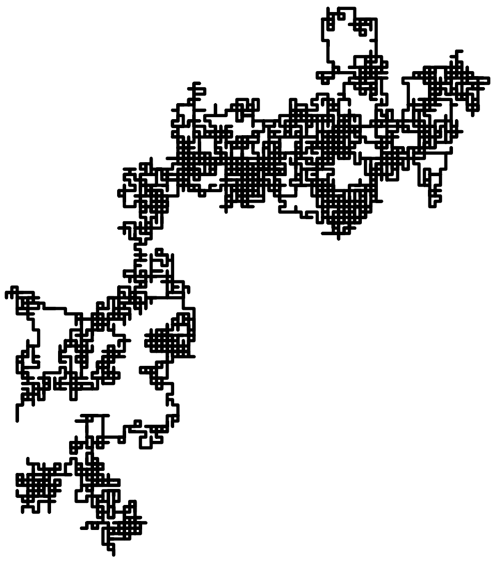


```
> d4:= rand(0..3): randomize():
> d4();
                                   3                                   (3)
> Brownian:=proc(n)
    local i,dir,path:="";
    for i from 1 to n do
       dir:=d4();
      if (dir=0) then path:=cat(path,"F"); fi;
      if (dir=1) then path:=cat(path,"RFL"); fi;
      if (dir=2) then path:=cat(path,"B"); fi;
      if (dir=3) then path:=cat(path,"LFR"); fi;
    od;
    return(path);
  end:

> ResetTurtle();
  TurtleCmd(Brownian(60));
```



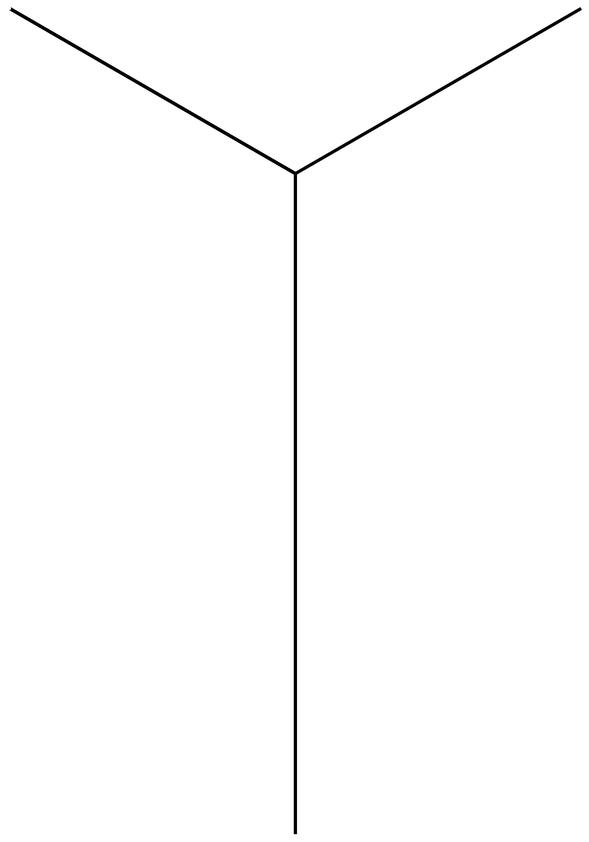
```
> TurtleCmd(Brownian(6000));
```



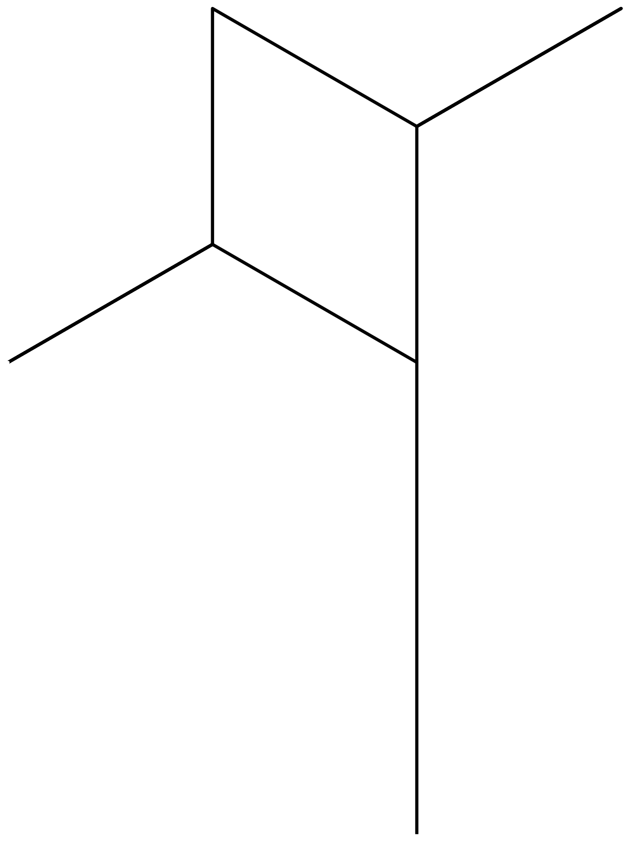
```
> 
  SetTurtleAngle(60);SetInitialTurtleHeading(90); # start going up.
```

Want to make a tree, which is a Y with each top replaced with a Y

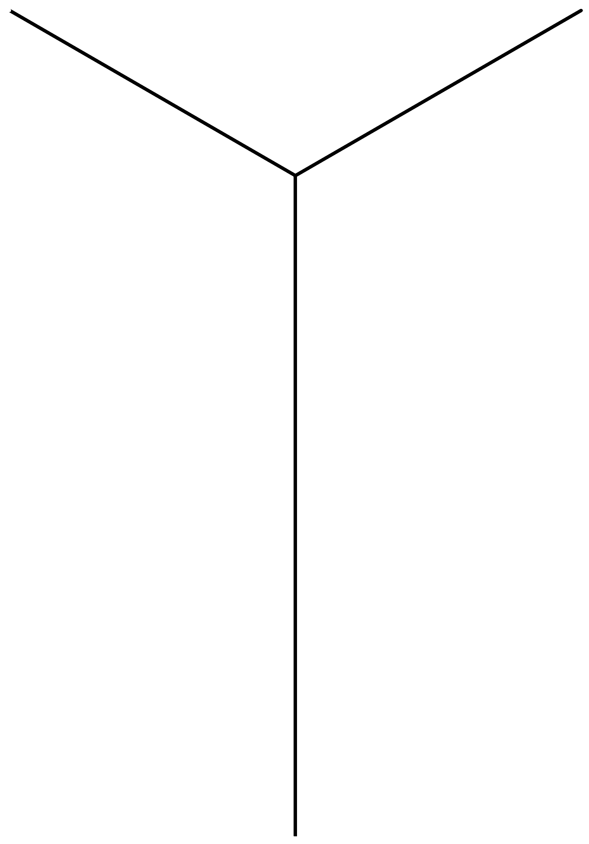
```
> TurtleCmd("FLSFBRRF");
```



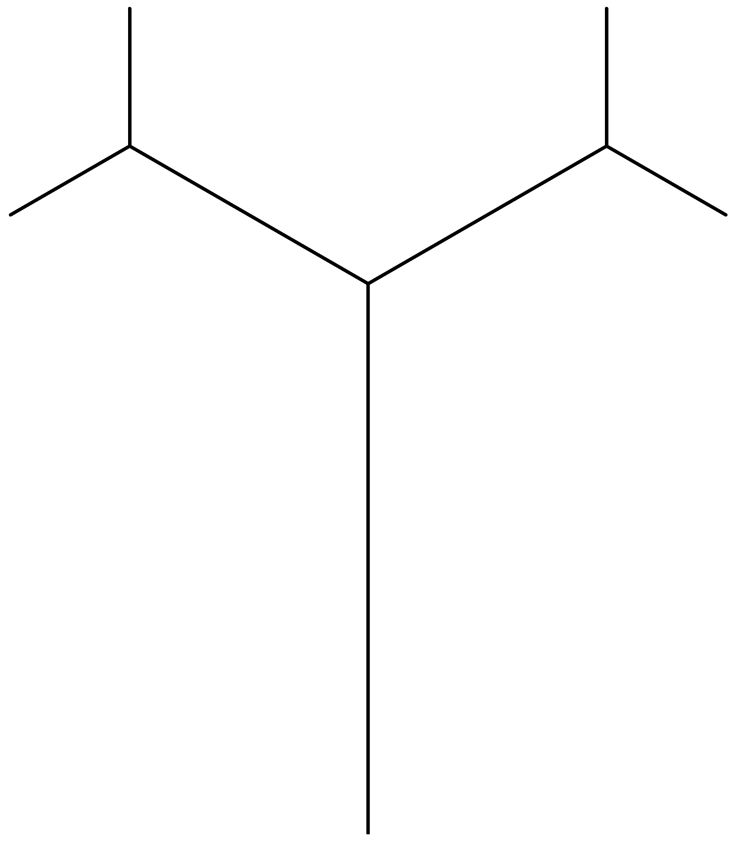
```
> TurtleCmd("FL SFLFBRRF RR FLFBRRF");
```



```
> TurtleCmd("FLSFBRRFBLGB");  # now turtle is at base, facing up
```



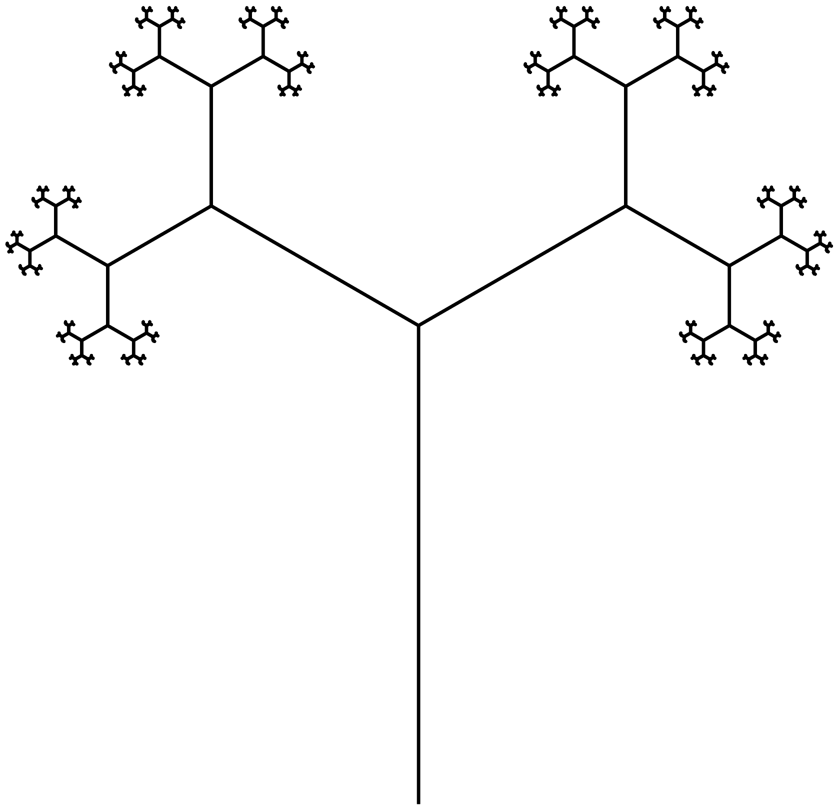
```
> TurtleCmd("FLS FLSFBRRRFBLGB RRFLSFBRRRFBLGB LGB");
```



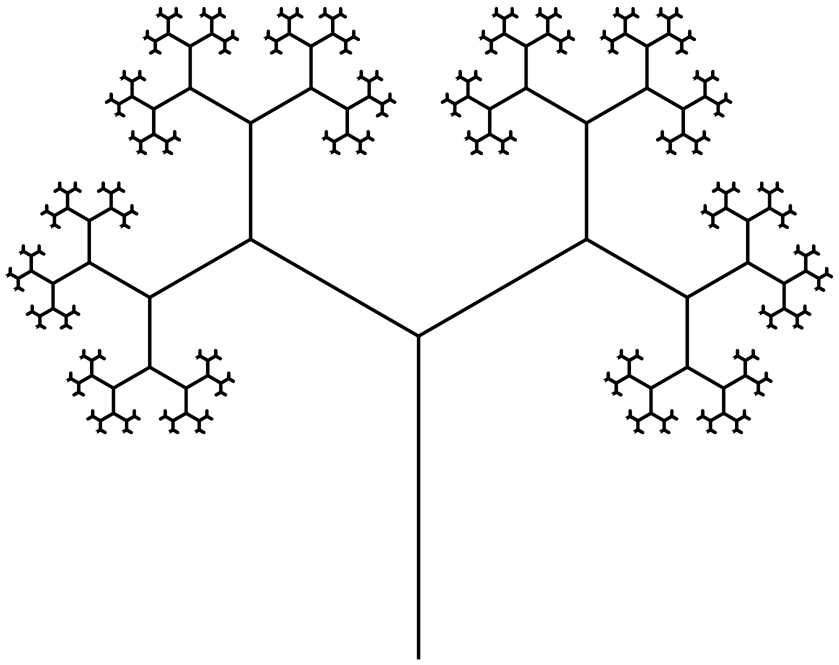
Tree structure is:
    base, left, branch, RR, branch, L, un-base.

```
> Tree:= proc(n)
    if (n=1) then return("F SLFBR RFBLG B"); fi;
    return( cat( "FSL", Tree(n-1), "RR", Tree(n-1), "LGB"));
  end:
```

```
> TurtleCmd(Tree(8));
```



```
> SetTurtleScale(.6);TurtleCmd(Tree(8));
```



```
> SetTurtleScale(.7); SetTurtleAngle(20);TurtleCmd(Tree(8));
```