# SnowFlake L-System

## based on Tree L-system

20221124 Bohyun Choi

# Background

Snowflakes can be made a lot of regular fractals,
but I thought I could make a nice image if I used the Tree L-system over and over again.

So, I made a tree L-system and rotated it to make one snowflake,
then I created and combined several snowflakes of different sizes
to complete the final snowflake image.

- **Difficulty of works**

  Snowflake's leaves did not gather correctly when rotate each leaves
  because the center point had to be moved using the translate()
  even during rotation to make snowflake leaves and for utilize 'F' function.

  To this end, pushMatrix() and popMatrix() were used several times to solve it.

# L-System Code

```
String S = "A";
String Rule_F = ">F<";
String Rule_A = "[+X][−X]FA";
String Rule_X = "A";

float length_factor = 1.2; //factor how regulate length of line
float angle = radians(30);
```

## Rules

F → >F<

A → [+X][-X]FA

X → A

The length_factor was introduced to express that the branches become shorter and shorter as L-system recurs repeatedly.

\> : Multiply length factor to branch's length
< : Devide by length factor to branch's length

```java
void render(String S, float branchLen) {
  int strLen = S.length();
  for (int i=0; i<strLen; i++) {
    switch( S.charAt(i) ) {
    case 'F':
      line(0, 0, branchLen, 0);
      translate(branchLen, 0);
      break;
    case '+':
      rotate(angle);
      break;
    case '−':
      rotate(-angle);
      break;
    case '[':
      pushMatrix();
      break;
    case ']':
      popMatrix();
      break;
    case '<':
      branchLen /= length_factor;
      break;
    case '>':
      branchLen *= length_factor;
      break;
    }
  }
}
```

# L-System Image

## Rotate (5 times) & Repeat