Coding Two: Advanced Frameworks
Assignment Element 1: Lab Work
Week 2 Exercise - Your first OF project
Select a JavaScript project you completed last term and port it to C++ using openFrameworks

Randomly twisted curve

This program wants to achieve a random twisting curve, and the color will change after a period of time.

```
def setup():
    size(500, 500)
    noFill()
    background(225)
    frameRate(30)
```

Set the size of the canvas.

Don't fill in color.

Set the border line thickness to 2 pixels.

Set the line color to light gray, 0 is pure black, and 255 is pure white.

```
def draw ():
  translate(width/2, height/2)
  r=map (sin(frameCount/200.0),-1,1,100,255)
  g=map(sin(frameCount/300.0),-1,1,0,255)
  b=map(sin(frameCount/400.0),-1,1,100,255)
  stroke(r,g,b,15)
```

Move the origin of the coordinate system to the center of the screen.

Random red component.

Random green component.

Random blue component.

Set the line color and transparency.

```
beginShape()
  for angle in range (0,360,2):
    radAngle =radians(angle)
```

Traverse the angle of a circle.

Converted to a radian value.

Use trigonometric functions to generate periodic data to avoid the problem of discontinuity at the beginning and end of the curve.

```
 noiseID=sin(radAngle)-cos(radAngle)\
          + 2*sin(radAngle)*sin(radAngle)
    radius=map(noise(noiseID*0.3,frameCount*0.01)\
                  ,0,1,100,300)
```

A two-dimensional noise function is used to achieve a random radius, which is affected by the angle radAngle and frameCount. While generating randomness, it is ensured that the radii of similar angles and adjacent frames are relatively close, and a certain continuity of randomness in space and time is realized. The code is too long and can be divided into multiple lines, add a backslash symbol "\", the system will automatically connect multiple lines of code to run. Use the periodicity of trigonometric functions such as sine and cosine. sin(0)=sin(2*PI), cos(0)=cos(2*PI), so it is composed of sin() and cos().

```
 x=radius*cos(radAngle)
    y=radius*sin(radAngle)
    curveVertex(x,y)
  endShape(CLOSE)
```

The function will avoid the problem of discontinuity at the beginning and end.
Set the XY coordinates corresponding to the angle.
Add corresponding vertices.