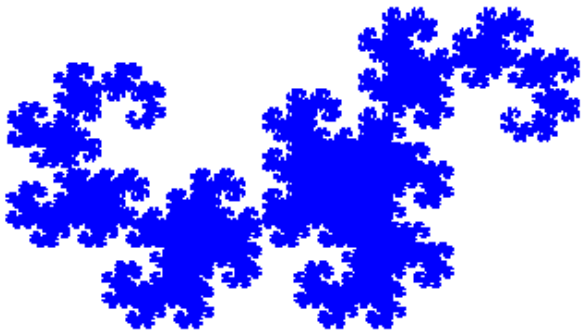
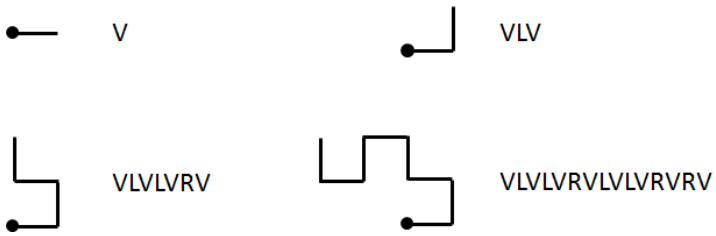
Dragon curve

A *dragon curve* is a repeated pattern that never intersects itself. You get a dragon curve of the order n ($n \geq 0$) when a strip of paper is folded n times in two and then unfolded so the folds form right angles. The name is based on the fact that the pattern is very similar to a certain mythical creature.



dragon curve

The instructions to draw a dragon curve can be written down as a string consisting only of the letters `F`, `L` and `R`. Here, the instruction `F` means "'draw a line while moving a step forward", the instruction `L` means "turn 90 degrees counterclockwise" and the instruction `R` means "turn 90 degrees clockwise". The key to writing down the instructions stems from the finding that a curve of the order n is ($n > 0$) formed by a curve of the order $ n-1 $, followed by the letter L and the curve of order $n-1$, which is made in reverse order (reverse the string instructions, `L` replaced by `R` and `R` replaced by `L`). The curve of the order 0 consists simply of a string with a single instruction: `F`. The figure below shows the instructions to draw dragon curves of the order 0, 1, 2 and 3.



Assignment

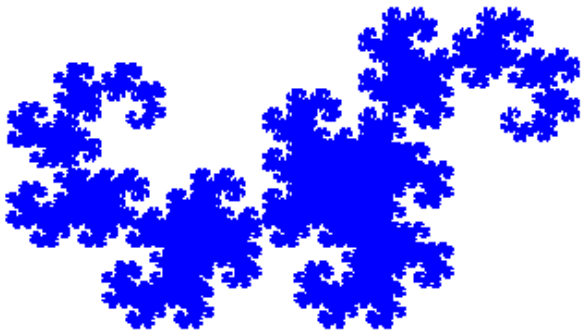
1. Write a function `dragoncurve` that returns the string with instructions needed to draw a dragon curve of the order n ($n \ geq 0$). The order n should be passed to the function as argument. Imagine a Cartesian plane where the x-axis of the coordinate system is directed to the right, and the y-axis upwards. Suppose we are initially at the origin (0,0) with our noses in the direction of the positive x-axis. Now, if we execute the instruction `F` of a dragon curve, we arrive at the point (1,0). If we then execute the instruction `L`, we stand with our noses towards the positive y-axis. If we re-execute the instruction `F`, we arrive at the point (1,1).
2. Write a function `curve2positons` to which a string must be passed consisting only of the letters `F`, `L` and `R`, in accordance with the instructions that are used to draw a dragon curve. The string with instructions will not necessarily deliver a dragon curve. The function must return the list of positions (points in the Cartesian plane represented as tuples with length two)

that are successively visited as the instructions from the given string are executed.

Example

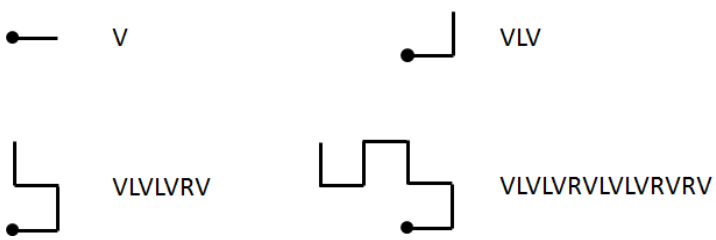
```
>>> dragon curve(0)
'F'
>>> dragon curve(1)
'FLF'
>>> dragon curve(2)
'FLFLFRF'
>>> dragon curve(3)
'FLFLFRFLFLFRFRF'
>>> curve2positons('FLFLFRFLFLFRFRF')
[(0, 0), (1, 0), (1, 1), (0, 1), (0, 2), (-1, 2), (-1, 1), (-2, 1), (-2, 2)]
>>> curve2positons('FFLRLFLRLFRFFFLF')
[(0, 0), (1, 0), (2, 0), (2, 1), (1, 1), (1, 2), (1, 3), (1, 4), (0, 4)]
>>> curve2positons('LLLFRFLLFRRRFLRLF')
[(0, 0), (0, -1), (-1, -1), (0, -1), (0, 0), (-1, 0)]
```

Een *drakencurve* vormt een herhaald patroon dat zichzelf nooit snijdt. Je krijgt een drakencurve van orde n ($n \geq 0$) wanneer je een strook papier n keer in twee vouwt, en daarna terug ontvouwt zodat de plooien rechte hoeken vormen. De naam is afgeleid van het feit dat het patroon sterke gelijkenissen vertoont met een bepaald mytisch wezen.



drakencurve

De instructies om een drakencurve te tekenen kunnen neergeschreven worden als een string die enkel bestaat uit de letters V, L en R. Hierbij staat de instructie V voor "teken een lijn terwijl je één stap vooruit zet", de instructie L betekent "draai 90° in tegenwijzerzin", en de instructie R betekent "draai 90° in wijzerzin". De sleutel tot het neerschrijven van de instructies komt voort uit de vaststelling dat een curve van orde n ($n > 0$) gevormd wordt door een curve van orde $n-1$, gevolgd door de letter L en de curve van orde $n-1$ die wordt afgelegd in omgekeerde volgorde (string met instructies omkeren, L vervangen door R, en R vervangen door L). De curve van orde 0 bestaat eenvoudigweg uit een string met één enkele instructie: V. Onderstaande figuur toont de instructies om drakencurves van orde 0, 1, 2 en 3 te tekenen.



Opgave

1. Schrijf een functie drakencurve die de string met instructies teruggeeft die nodig zijn om een drakencurve van orde n ($n \geq 0$) te tekenen. De orde n moet als argument aan de functie doorgegeven worden.
2. Veronderstel een cartesisch vlak waarbij de x-as van het coördinatenstelsel naar rechts gericht is, en de y-as naar boven. Stel dat we ons initieel in de oorsprong (0,0) bevinden met onze neus in de richting van de positieve x-as. Als we nu bijvoorbeeld de instructie V van een drakencurve uitvoeren, dan komen we in het punt (1,0) terecht. Als we daarna de instructie L uitvoeren, dan staan we met onze neus richting positieve y-as. Als we nu opnieuw de instructie V uitvoeren, dan komen we in het punt (1,1) terecht.
   Schrijf een functie curve2posities waaraan een string moet doorgegeven worden die enkel bestaat uit de letters V, L en R, overeenkomstig de instructies die gebruikt worden voor het tekenen van een drakencurve. De string met instructies zelf hoeft dus niet noodzakelijk een drakencurve op te leveren. De functie moet de lijst van posities (punten in het cartesich vlak voorgesteld als tuples van lengte twee) teruggeven die achtereenvolgens bezocht worden als de instructies uit de gegeven string uitgevoerd worden.

Voorbeeld

```
>>> drakencurve(0)
'V'
>>> drakencurve(1)
'VLV'
>>> drakencurve(2)
'VLVLVRV'
>>> drakencurve(3)
'VLVLVRVLVLVRVRV'
>>> curve2posities('VLVLVRVLVLVRVRV')
[(0, 0), (1, 0), (1, 1), (0, 1), (0, 2), (-1, 2), (-1, 1), (-2, 1), (-2, 2)]
>>> curve2posities('VVLRLVLRLVRVVVLV')
[(0, 0), (1, 0), (2, 0), (2, 1), (1, 1), (1, 2), (1, 3), (1, 4), (0, 4)]
>>> curve2posities('LLLVRVLLVRRRVLRLV')
[(0, 0), (0, -1), (-1, -1), (0, -1), (0, 0), (-1, 0)]
```