Algorithmen und Datenstrukturen

Rekursion

Aufgabe 1: Türme von Hanoi [2 Punkte]

Sie haben in den Unterlagen einen Lösungsansatz für die Problemstellung der *Türme von Hanoi*. Implementieren Sie einen *HanoiServer* mit der rekursiven Methode *moveDisk*. Geben Sie einen Text aus, der einer Anleitung für die Lösung entspricht:

```
move A to C
move C to B
…
```

Die Anzahl der Scheiben soll als Parameter mitgegeben werden?

Aufgabe 2: Koch'sche Schneeflocke [4 Punkte]

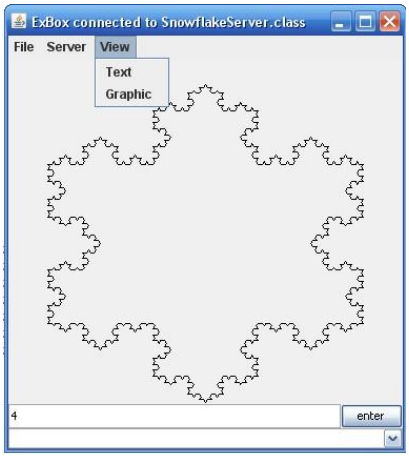
Es soll nicht direkt in der ExBox gezeichnet werden, sondern es soll ein CommandInterpreter `SnowflakeServer` entwickelt werden, der die Zeichnung mit Hilfe der Klasse Turtle erstellt und die fertige Graphik in getTrace() als XML String übergibt.

Hinweise:

- Instanzieren Sie in Ihrer `execute` Methode eine Turtle Klasse, mit der Sie die Schneeflocke nach der Vorlage aus dem Skript zeichnen.
- Mittels `getTrace()` kann der zurückgelegte Weg der `Turtle` als String abgefragt werden. Die Klasse `Turtle` liefert eine Spur des zurückgelegten Weges in der oben beschriebenen Form.

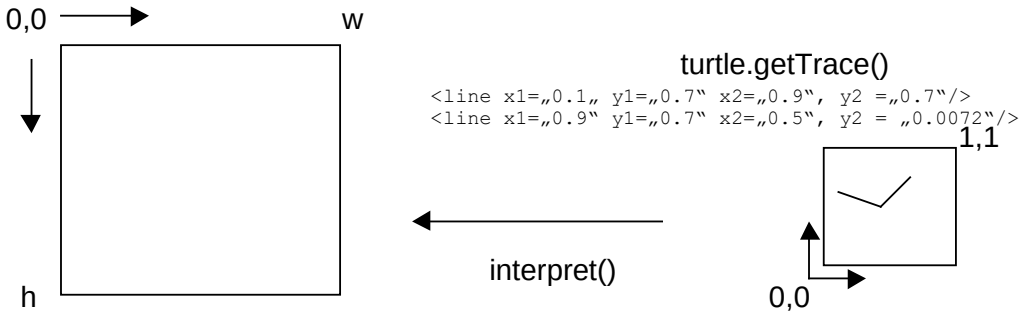
Hinweise zur Implementation:

Es soll die in der Vorlesung besprochene Schneeflockenkurve mittels Turtlegraphic gezeichnet werden. Für diesen Zweck wurde die ExBox zunächst um die Möglichkeit der graphischen Ausgabe erweitert.



Es existiert zusätzlich ein `GraphicPanel`, mit dessen Hilfe Zeichnungen dargestellt werden können. Das Menu ist entsprechend um den Eintrag `View` mit den Untereinträgen `Text` und `Graphic` erweitert. Damit kann zwischen graphischer und textueller Darstellung umgeschaltet werden. In der Behandlung der `interpret` Methode in der Experiementierbox wird das Resultat mittels `setFigure()` dem `GraphicPanel` übergeben werden, falls die graphische Ansicht aktiviert ist.

Die Graphik wird als XML-String erzeugt, wobei jeweils eine Gruppe von vier Werten



View Koordinaten-System

Modell Koordinaten-System

(startX, startY, endX, endY), denen das Wort "<line" vorangestellt wird, eine Linie definieren (siehe oben). Der Tokenizer wurde mit folgenden Trennzeichen aufgesetzt: `" <>=/,\"\n"`.

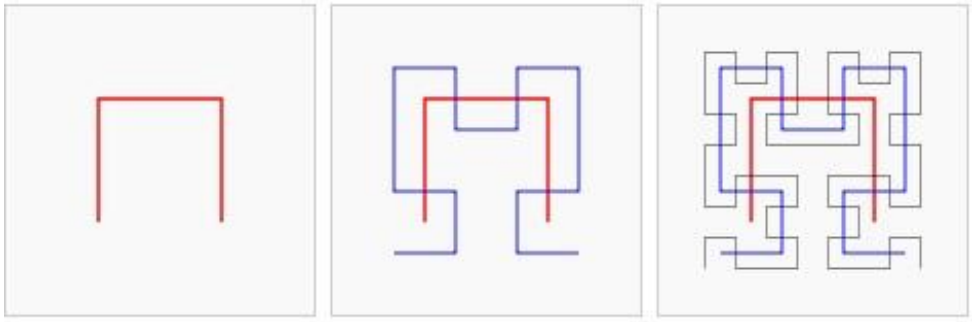
Dabei sind die Koordinaten der Modell-Zeichenfläche so normiert, dass die linke untere Ecke bei 0.0,0.0 liegt und die rechte obere Ecke bei 1.0,1.0. Die Normierung hat übrigens den Vorteil, dass für die Berechnung keine Annahmen über das View Koordinaten-System (z.B. aktuelle Grösse des Fensters) gemacht werden muss. Das Einheitsquadrat wird in maximaler Grösse dargestellt.

Mittels `getWitdth()` und `getHeight()` wird die aktuelle Grösse des `GraphicPanel` ermittelt.

Aufgabe 3: Hilbertkurve [4 Punkte]

In der Mathematik ist die Hilbert-Kurve eine stetige Kurve, die – durch Wiederholung ihres Konstruktionsverfahrens – jedem beliebigen Punkt einer quadratischen Fläche beliebig nahekommt und die Fläche vollständig ausfüllt. Die Hilbert-Kurve ist eine sogenannte raumfüllende oder FASS-Kurve. Sie wurde 1891 von dem deutschen

Mathematiker David Hilbert entdeckt[1]. Die Möglichkeit, mit einer stetigen eindimensionalen Kurve ein zweidimensionales Gebiet komplett abdecken zu können, war den Mathematikern des neunzehnten Jahrhunderts neu.



Hinweise:

Die Hilbert-Kurve Stufe 0 sieht wiefolgt aus:

```
private void hilbert(int depth, double dist,
                     double angle) {
      turtle.turn(-angle);
      // draw recursive
      turtle.move(dist);
      turtle.turn(angle);
      // draw recursive
      turtle.move(dist);
      // draw recursive
      turtle.turn(angle);
      turtle.move(dist);
      // draw recursive
      turtle.turn(-
      angle);
}

... int depth = Integer.parseInt(command);
double dist = 0.8 / (Math.pow(2,depth+1)-1);
turtle = new Turtle(0.1, 0.1);
hilbert(depth, dist, -90);
```

Überlegen Sie sich, wie Sie mittels rekursiven Aufrufen, von der roten Kurve zur blauen kommen. Dabei kann mit angle jeweils die Ausrichtung der U-förmigen Kurve festgelegt werden (alternierend 90 und -90 Grad).