Informatik I WS 2005/2006

Übungsblatt 7

Abgabe: 19.12.05 - 21.12.05

Aufgabe 1 (Klassen, Unit-Tests) Punkte: 10

Mit „Turtle-Grafik“ wird eine Bildbeschreibungssprache bezeichnet, bei der man sich vorstellt, dass eine Schildkröte, an der ein Stift befestigt ist, sich auf einem Blatt Papier bewegt und durch einfache Kommandos wie „Stift heben“, „Stift senken“, „Laufe vorwärts“ oder „Nach rechts drehen“ gesteuert werden kann.

a) In dieser Aufgabe soll eine Klasse `Turtle` für eine solche Turtle-Grafik implementiert werden. Dem Konstruktor sollen Höhe und Breite der Zeichenebene in Pixeln mitgeteilt werden können. Falls der Konstruktor ohne weitere Argumente aufgerufen wird, sollen vernünftige Standardwerte eingesetzt werden. Im Konstruktor soll dann ein weißes Bild erzeugt werden. Die Schildkröte startet an der Position (0,0) und blickt in Richtung der positiven y-Achse. Die Farbe des Stiftes soll zu Beginn schwarz sein und der Stift soll auf der Zeichenebene aufliegen.
Die Steuerung der Schildkröte erfolge über folgende Funktionen:

- `penDown()`: Der Stift wird auf das Papier aufgesetzt. Wenn die Schildkröte sich nun bewegt, zieht sie dabei einen Strich hinter sich her.
- `penUp()`: Der Stift wird von der Zeichenebene gehoben. Wenn die Schildkröte sich nun bewegt, bleiben keine Striche zurück.
- `forward( n )`: Die Schildkröte bewegt sich um n Pixel in Blickrichtung (n $>$ 0). Wenn der Stift auf der Zeichenebene aufliegt, hinterlässt sie dabei einen Strich.
- `backward( n )`: Die Schildkröte bewegt sich um n Pixel rückwärts, also entgegen der Blickrichtung (n $>$ 0). Wenn der Stift auf der Zeichenebene aufliegt, hinterlässt sie dabei einen Strich.
- `left( alpha )`: Die Schildkröte dreht sich um den Winkel alpha nach links.
- `right( alpha )`: Die Schildkröte dreht sich um den Winkel alpha nach rechts.
- `moveTo( (x,y) )`: Die Schildkröte wird vom Blatt gehoben und auf dem Punkt (x, y) wieder abgesetzt. Die Blickrichtung ändert sich bei dieser Operation nicht.
- `getPosition()`: Liefert die aktuell Position der Schildkröte als Tupel (x, y) zurück.
- `getOrientation()`: Liefert die aktuelle Drehung der Schildkröte zurück. Die Drehung ist der Winkel zwischen der aktuellen Blickrichtung und der Blickrichtung zu Beginn (also der positiven y-Achse).
- `setColor( (r, g, b) )`: Ändert die Farbe des Stiftes.
- `show()`: Zeigt das entstandene Bild an.

Achten Sie darauf, dass die Schildkröte nicht von der Zeichenebene fällt. D.h.: Sollte ein Befehl die Schildkröte über den Rand des Blattes hinausführen, so soll dieser Befehl nicht ausgeführt werden.

b) Schreiben Sie einen Unit-Test für diese Klasse. Erstellen Sie dazu ein Dreieck, ein Quadrat und ein Achteck mit Hilfe der `Turtle`-Klasse.

Hinweis: Diese Klasse wird im Kapitel 7 (Rekursion) noch nützlich sein.

Aufgabe 2 (Umgang mit externen Modulen: Math, Random, PIL) Punkte: 10

Die meisten von Ihnen haben sicherlich schon einmal mit einem Bildbearbeitungsprogramm gearbeitet, sei es Gimp oder Photoshop oder.... All diese Programme haben eine Vielzahl von Filtern integriert, mit denen Bilder verändert werden können. Auch die „Python Imaging Library“ besitzt im Modul ImageFilter einige solcher Filterfunktionen, um etwa ein Bild zu Schärfen, oder um die Konturen hervorzuheben.
In dieser Aufgabe sollen Sie selbst einige solcher Bildeffekte implementieren, die die „Python Imaging Library“ bisher noch nicht beherrscht:

a) Zuerst geht es um das Erzeugen eins Wellen-Effekts. Das Programm soll von der Kommandozeile eine Bilddatei einlesen und ein neues Bild erzeugen, welches das ursprüngliche Bild wellenförmig verzerrt darstellt.
Dabei entspricht die Farbe des neuen Pixels P_{neu} an der Position (i,j) der des Originalpixels an der Position (i_{org},j_{org}) mit:
i_{org} = i
$j_{org} = j + 20sin(2\pi\frac{i}{128})$
D.h.: Die Farbe des Pixels P_{neu} im neuen Bild an der Stelle (i,j) muss auf die Farbe des Originalpixels an der Position $(i, j + 20sin(2\pi\frac{i}{128}))$ im Originalbild gesetzt werden.

b) Als nächstes soll ein Effekt zum Verquirlen des Bildes entwickelt werden. Das Programm soll wieder von der Kommandozeile eine Bilddatei einlesen und ein neues Bild erzeugen, welches das ursprüngliche Bild um den Mittelpunkt des ursprünglichen Bildes gedreht darstellt.
Hierbei entspricht die Farbe des neuen Pixels P_{neu} an der Position (i,j) der des Originalpixels an der Position (i_{org},j_{org}) mit:
r = $\sqrt{(i - i_0)^2 + (j - j_0)^2}$
$\theta = \frac{r\pi}{256}$
$i_{org} = -(i - i_0)cos(\theta) + (j - j_0)sin(\theta) + i_0$
$j_{org} = (i - i_0)sin(\theta) + (j - j_0)cos(\theta) + j_0$
Dabei ist (i_0, j_0) der Mittelpunkt des Originalbildes.

c) Letztendlich soll noch ein Filter für einen Milchglaseffekt implementiert werden. Das Programm soll wieder von der Kommandozeile eine Bilddatei einlesen und ein neues Bild erzeugen. Hierbei entspricht die Farbe des neuen Pixels P_{neu} an der Position (i,j) der Farbe eines zufällig aus einem Quadrat der Grösse 10 um den Originalpixel ausgewählten Bildpunktes.

Hinweise:

- Bilder lesen kann man mit `Image.open(Pfad+Name des Bildes)`, also beispielsweise `orgimage = Image.open("c:/temp/nikolaus.jpg")`
- Breite und Höhe eines Bildes werden als 2-Tupel im size-Attribut gespeichert. Beispiel: `width, height = orgimage.size`
- Die Farbe eines Pixels ermittelt man mit der Funktion `getpixel((x, y))`. Anwendungsbeispiel: `newpixelColor = orgimage.getpixel((i, j))`

(a) Originalbild

(b) Welleneffekt

(c) Originalbild

(d) Verquirl-Effekt

(e) Originalbild

(f) Milchglaseffekt