Lektion 8: Bedingungen

Bisher wurden nur Prozeduren behandelt, in denen eine feste Abfolge von Befehlen durchzuführen war. Mit den Befehlswörtern

if { Bedingung } *[* { Befehlsfolge } *]*

ifelse { Bedingung } *[* { Befehlsfolge1 } *] [* { Befehlsfolge2 } *]*

kann man den Programmablauf davon abhängig machen, ob eine bestimmte Bedingung erfüllt ist oder nicht.
Wenn die nach *if* angegebene Bedingung erfüllt ist, dann wird zunächst die in der eckigen Klammer genannte Befehlsfolge ausgeführt. (Eine Befehlsfolge kann natürlich auch aus nur einem Befehl bestehen.)

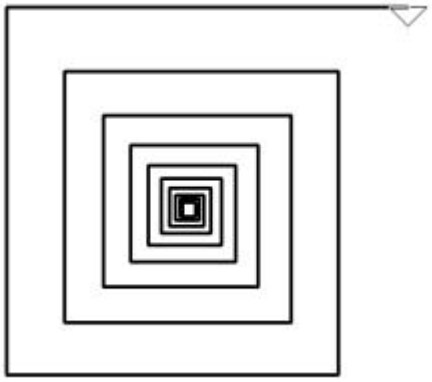
Nach der Überprüfung der Bedingung nach *if* und nach der Ausführung der Befehlsfolge in der eckigen Klammer geht der Rechner zur nächsten Zeile der Prozedur über, es sei denn, durch den Befehl würde eine neue Prozedur aufgerufen oder durch den Befehl *[stop]* die Programmausführung beendet.

Ist die Bedingung hinter *if* nicht erfüllt, geht der Rechner direkt zu der nächsten Prozedurzeile über.

Beim Befehl *ifelse* wird im Falle, dass die Bedingung nicht erfüllt ist, die Befehlsfolge in der zweiten eckigen Klammer abgearbeitet.

Problem 8.1

Schreibe eine Prozedur mit Namen *streckenzug* , durch die ein Streckenzug mit einem festen Winkel (Parameter *w*) zwischen den einzelnen Strecken gezeichnet werden soll. Die Anfangsstrecke (*länge*) soll variabel sein: die Streckenlängen sollen solange mit dem Faktor 1,1 zunehmen, bis die Länge der Strecke einen festen Wert *max* überschreitet.



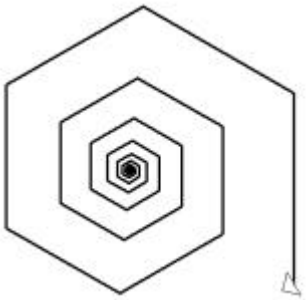
Lösung

In der Prozedur *streckenzug* treten die Parameter *länge*, *w* und *max* auf. Achten wir zunächst nicht auf die Abbruchbedingung, dann könnte die gesuchte rekursive Prozedur wie folgt aussehen:

```
to streckenzug :länge :w :max
fd :länge rt :w
streckenzug :länge*1.1 :w :max
end
```

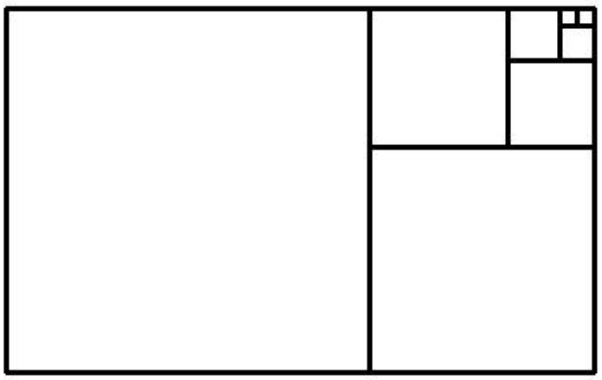
Die Abbruchbedingung verlangt, dass man den aktuellen Wert der Streckenlänge mit dem vorgegebenen Wert von *max* vergleicht. Wenn also *:länge > :max* ist, dann soll der Programmablauf unterbrochen werden. Die Abfrage der Gültigkeit der Bedingung muss am Anfang der Prozedur stehen, da sonst die Prozedur nicht rechtzeitig beendet wird.

```
to streckenzug :länge :w :max
if :länge > :max  [ stop  ]
fd :länge rt :w
streckenzug :länge*1.1 :w :max
end
```



Problem 8.2

Schreibe eine Prozedur mit Namen *zerlegung* , durch die eine gegebene Rechteckfläche in möglichst wenige quadratische Teilflächen zerlegt wird.



Anleitung: Man zeichnet zunächst ein Rechteck und dann darin ein möglichst großes Quadrat. Es bleibt ein (kleineres) Rechteck übrig, das man genauso behandeln kann.Wann ist das Verfahren zu Ende?

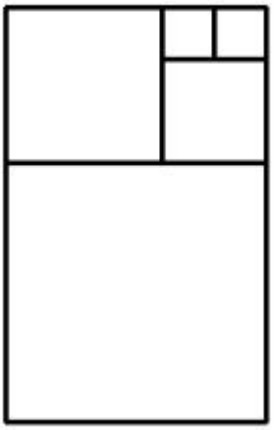
Lösung

Wir unterscheiden drei Fälle:

1. Liegt ein querformatiges Rechteck vor, dann lässt sich ein Quadrat (Prozedur *qu*) abtrennen, dessen Seitenlänge gleich der Breite des Rechtecks ist. In diesem Fall müssen wir anschließend die Turtle um diese Seitenlänge nach rechts wandern lassen und das in der Länge verkürzte Rechteck weiterzerlegen.

```
if :breite>:höhe [qu :höhe rt 90 fd :höhe lt 90 zerlegung :breite-:höhe :höhe ]
```

2. Liegt ein hochformatiges Rechteck vor, dann kann man ein Quadrat eintragen, dessen Seitenlänge gleich der Länge des vorliegenden Rechtecks ist. Anschließend muss die Turtle nach oben wandern. Danach muss man das in der Breite verkleinerte Rechteck bearbeiten.



```
if  :breite < :höhe
        [qu :breite fd :breite zerlegung :breite :höhe-:breite]
```

3. Liegt ein Quadrat vor, ist man fertig.

```
if :breite = :höhe  [stop]
```

Damit ergibt sich die nachstehende Prozedur. Wir zeichnen zu Beginn erst das gegebene Rechteck und danach die Teilfiguren.

```
to zerlegung :breite :höhe
rechteck :breite :höhe
if  :breite > :höhe [ qu :höhe rt 90 fd :höhe lt 90 zerlegung :breite-:höhe :höhe ]
if  :breite < :höhe [ qu :breite fd :breite zerlegung :breite :höhe-:breite ]
if  :breite = :höhe [ stop ]
end
```

Problem 8.3

Schreibe eine Prozedur *rechteckquer*, mit den Parametern *x* und *y* durch die unabhängig von der Eingabe der Parameterwerte *x* und *y* ein querliegendes Rechteck gezeichnet wird.

Lösung

Drei Fälle sind möglich:
1. x ist kleiner als y
2. x ist größer als y
3. x ist genauso groß wie y, d. h. es liegt ein Quadrat vor.

Im 1. Fall muss man die Werte der Parameter vertauschen, in den beiden anderen Fällen kann man das Rechteck ohne Vertauschung der Parameterwerte zeichnen.

Wenn die Grund-Prozedur *rechteck* wie folgt lautet:

```
to rechteck :a :b
repeat 2 [ fd :b rt 90 fd :a rt 90 ]
end
```

dann ergibt sich die folgende gesuchte Prozedur:

```
to rechteckquer :x :y
if :x < :y [ rechteck :y :x ]
if :x > :y  [ rechteck :x :y ]
if :x = :y [ rechteck :x :x ]
end
```

Eigentlich sind dies nur zwei Fälle: Im ersten Fall müssen die Parameterwerte vertauscht werden, in den beiden anderen Fällen können sie ohne Vertauschung übernommen werden. Hier bietet es sich an, den *ifelse*-Befehl zu benutzen:

```
to rechteckquer :x :y
ifelse :x < :y [ rechteck :y :x ] [ rechteck :x :y ]
end
```

Die Fallunterscheidung kann noch übersichtlicher gestaltet werden, wenn man die folgenden LOGO-Befehle benutzt: Man schreibt in verschiedene Zeilen zur Überprüfung die Bedingung (*test*) und die Befehlsfolgen, die abgearbeitet werden sollen, wenn die Bedingung erfüllt ist (*iftrue*) oder wenn dies nicht der Fall ist (*iffalse*)

test { Bedingung }

iftrue [{ Befehlsfolge } *]* - kurz: *ift*

iffalse [{ Befehlsfolge } *]* - kurz: *iff*

Die Prozedur *rechteckquer* erhält damit die folgende Form:

```
to rechteckquer :x :y
test :x < :y
ift  [ rechteck :y :x]
iff  [ rechteck :x :y]
end
```

Überlege ...

a) Was geschieht, wenn in der Prozedur *streckenzug* die Abfrage an einer späteren Stelle steht?

b) Jemand ermittelt als Lösung des Problems *rechteckquer* die folgende Prozedur. Warum funktioniert sie nicht wie gewünscht? Benutze notfalls den Protokoll-Befehl, um den Fehler zu finden.

```
to rechteckquer :x :y
if :x < :y [ rechteckquer :y :x ]
rechteck :x :y
end
```

c) In der Prozedur *streckenzug* könnte man auch *ifelse* verwenden. Überlege wie und begründe dies.

d) Auch die folgende Prozedur kann dazu benutzt werden, ein Rechteck in möglichst große Quadrate zu zerlegen.

```
to rechteckzerlegung :breite :höhe
rechteck :breite :höhe
if  :breite = :höhe [ stop ]
ifelse  :breite > :höhe [ fd :höhe rt 90 rechteckzerlegung :höhe :breite ]
                [ quadrat :breite fd :breite rechteckzerlegung :breite :höhe-:breite ]
end
```

Analysiere diese Prozedur und vergleiche sie mit der Prozedur *zerlegung*. Schreibe alle Befehle auf, die durchgeführt werden, wenn die Parameterwerte 70 und 30 eingegeben werden.

e) In der Prozedur *rechteckzerlegung* kann ein Befehl weggelassen werden, wenn ein anderer Befehl ergänzt wird. Prüfe dies nach.

f) Manche Grafiken muss man unten auf dem Bildschirm beginnen, damit die Größe des Bildschirms ausreicht.
Jemand erstellt eine Grafik mit Hilfe einer rekursiven Prozedur. In der ersten Zeile dieser Prozedur steht die Befehlsfolge *pu sety –200 pd*
Wieso misslingt die Grafik? Wie muss man vorgehen?

Übungen

Ü8.1
Zeichne einen Streckenzug mit kleiner werdender Streckenlänge. Brich die Zeichnung ab,
a) wenn eine Streckenlänge unter 5 erreicht ist,
b) wenn die X-Koordinate größer als 200 ist,
c) wenn die Turtle wieder nach oben blickt.

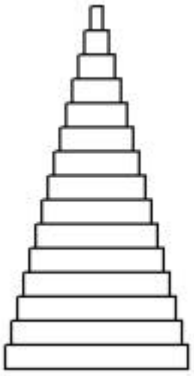
Ü8.2
Schreibe eine Prozedur *rechteckhoch*, durch die bei beliebiger Eingabe auf jeden Fall ein Rechteck im Hochformat gezeichnet wird.

Ü8.3
Schreibe eine rekursive Prozedur für das Zeichnen eines regelmäßigen n-Ecks. Gib die Bedingung ein, dass die Zeichnung beendet werden soll, wenn die Turtle wieder

(1) durch die Bildschirmmitte läuft,
(2) in die gleiche Richtung blickt hat wie am Anfang.

Bei verschiedenen Werten für n funktioniert die Abbruchbedingung nicht. Überlege, woran dies liegen kann.

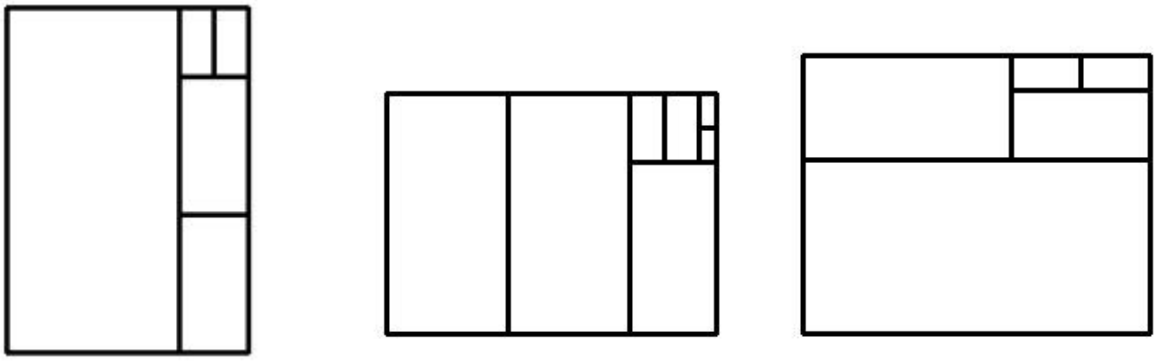
Ü8.4



Zeichne mehrere Rechtecke übereinander, so dass eine Pyramide entsteht. Das untere Rechteck soll die Breite 200, das nächste die Breite 180 ... haben. Brich die Zeichnung mit einer geeigneten Bedingung ab. Verallgemeinere die Prozedur durch Einführung von geeigneten Parametern.

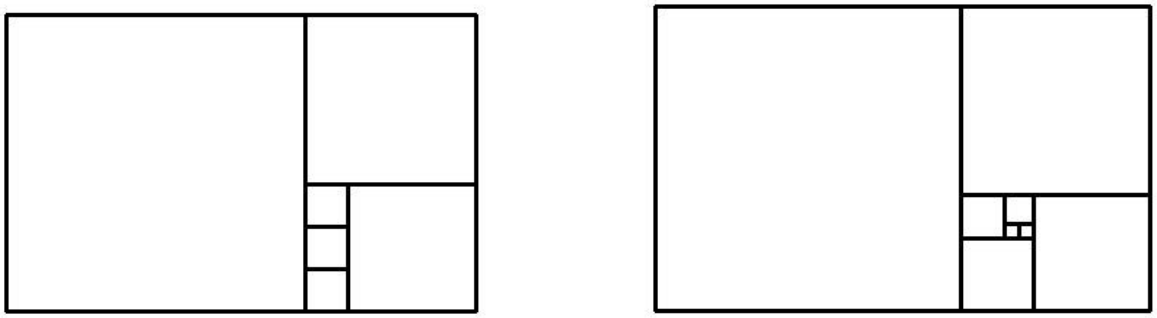
Ü8.5

Schreibe eine Prozedur analog zu *rechteckzerlegung* oder *zerlegung*, durch die ein vorgegebenes Rechteck in möglichst wenige Rechtecke zerlegt wird, deren Seitenlängen (Höhe zu Breite) sich wie 2:1 (1:2) verhalten sollen. Unterscheide drei Fälle. Wie müssen die Bedingungen lauten? Ist diese Zerlegung für alle Rechtecke möglich, d. h. nach endlich vielen Schritten beendet?

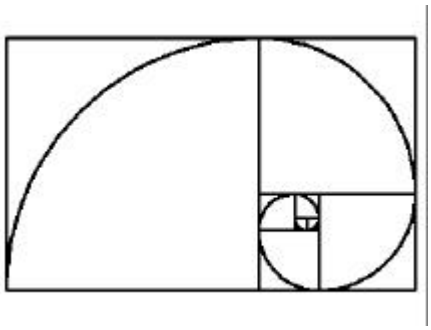


Ü8.6

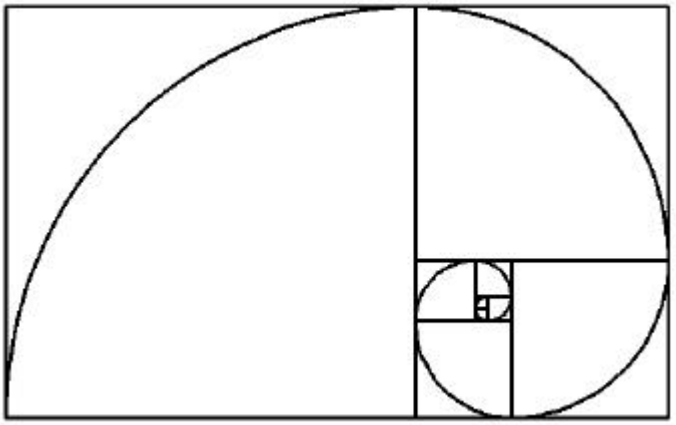
Man kann die Rechteckzerlegung in Quadrate so anordnen, dass dies „wie in einem Bogen“ (gegen den Uhrzeigersinn) geschieht.



Einen solchen Bogen aus Viertelkreisen kann man tatsächlich einzeichnen, wenn die Seitenlängen des Rechtecks benachbarte Zahlen aus der FIBONACCI-Folge (oder Vielfache dieser Zahlen) sind, also z. B. ... 50, 80, 130, 210, 340, 550, ...



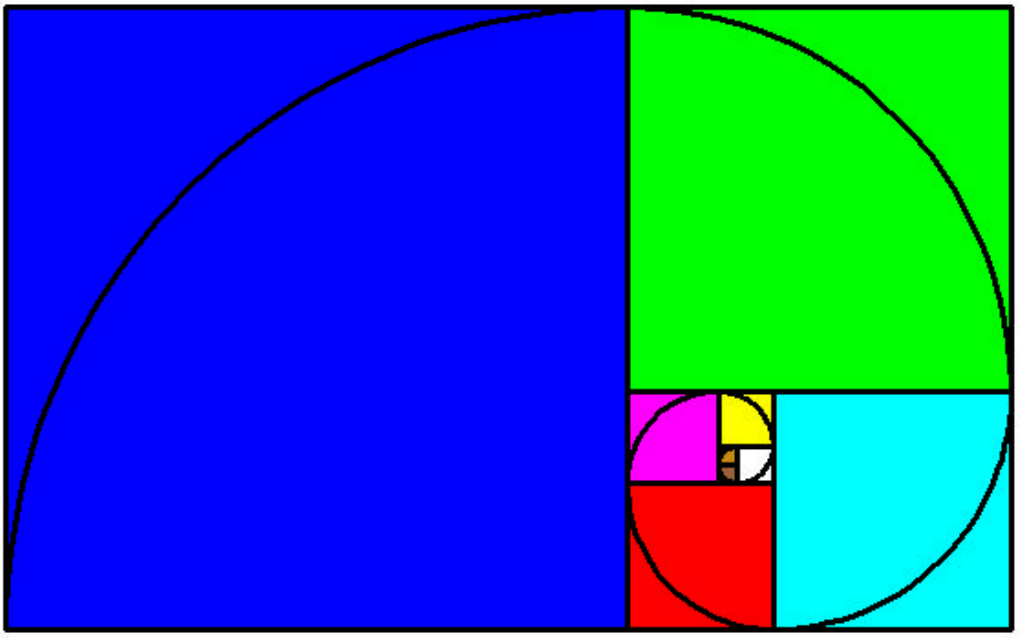
Rechteckzerlegung 210 x 340



Rechteckzerlegung 340 x 550

Tipp für das Zeichnen der Viertelkreise: Ein regelmäßiges 120-Eck kann man kaum von einem Kreis unterscheiden. Wenn man also ein Viertel hiervon zeichnet, hat man praktisch einen Viertelkreis geschlagen. Allerdings sieht der „Anfang“ und das „Ende“ dieses Viertelkreises nicht so gut aus. Was kann man hier verbessern?

Das Besondere an den FIBONACCI-Rechtecken ist, dass alle Quadrate nur einmal vorkommen – bis auf das kleinste Quadrat, das genau zweimal auftritt. Die Seitenlänge des kleinsten Quadrats ist gleich dem ggT der Seitenlängen des Ausgangsrechtecks. In Farbe sieht das Ganze noch besser aus ... Die Farben wurden nacheinander aus der LOGO-Farbpalette gewählt (blau = 1; grün = 2; ...).

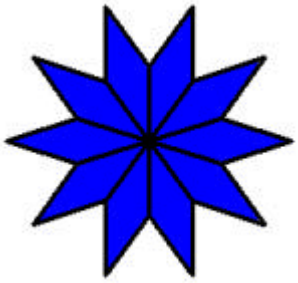


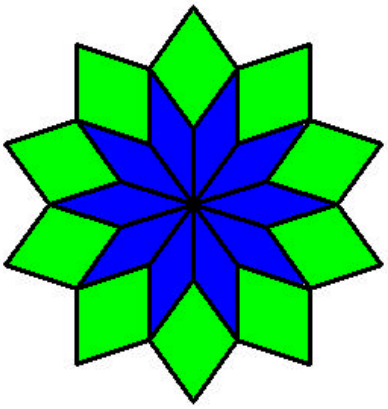
Ü8.7

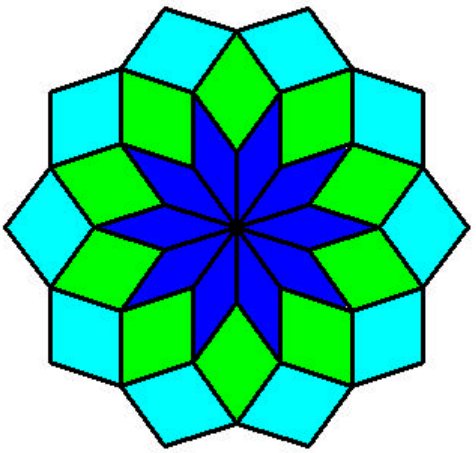
(1) Man kann Rauten so anordnen, dass sie einen Stern bilden – hier für einen spitzen Winkel von 36° = 360°/10.

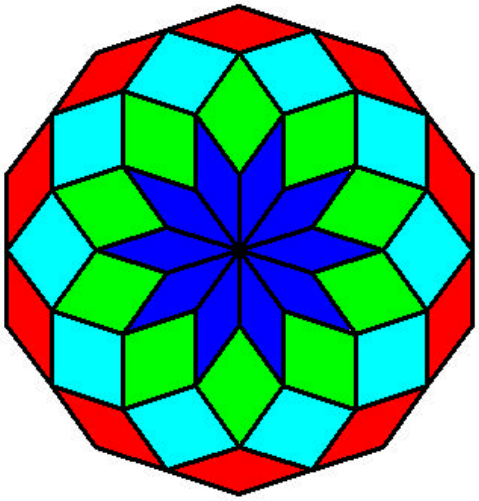
(2) In die Lücken lassen sich wieder Rauten einpassen (Wie groß ist der spitze Winkel?)

(3) In diese wiederum passen wiederum gewisse Rauten hinein usw.









Schreibe eine Prozedur, mit deren Hilfe man solche Parkettierungen von n-Ecken zeichnen kann.

Verändere auch die Anzahl der Rauten, mit denen die Figur begonnen wird (innerer Stern).