1.Arbeit

Marl, 27. 03. 2009

Differenzierungskurs Informatik Jahrgangsstufe 8

Arbeit1_MSWLogo_Diff8.doc

1. Aufgabe: a) Folgendes Programm liegt vor:

```
to unbekannt1 :laenge
 repeat 6[fd :laenge back :laenge rt 60]
end
```

Wie sieht die Figur beim Aufruf **unbekannt1 100** aus? Zeichne.

b) Die Schleife aus a) wird benutzt, um einen rekursiven Aufruf durchzuführen. Wie sieht die Figur beim Aufruf **unbekannt2 100** aus? Skizziere.

```
to unbekannt2 :laenge
 if :laenge<1[stop]
 repeat 6[fd :laenge  unbekannt2 :laenge/3  back :laenge rt 60]
end
```

c) Erläutere die Programmiermethode REKURSIV an einem selbstgewählten Beispiel.

2. Aufgabe: a) Folgendes MSWLogo-Programm ist gegeben:

```
to unbekannt3 :laenge
if :laenge <5[stop]
pu  fd :laenge pd
fd :laenge back :laenge  rt 90 fd :laenge back :laenge lt 90

pu back :laenge pd
fd :laenge back :laenge rt 90 fd :laenge back :laenge lt 90

pu rt 90 fd :laenge lt 90 pd
fd :laenge back :laenge rt 90 fd :laenge back :laenge lt 90

pu rt 90 back :laenge
lt 90 pd
end
```

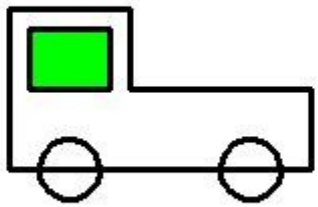
Wie sieht die Figur beim Aufruf **unbekannt3 100** aus? Zeichne.

b) In die Leerzeilen des obigen Programms werden jetzt jeweils rekursive Aufrufe eingebaut. Skizziere die dann entstehende Figur beim Aufruf von unbekannt4 100.

```
to unbekannt4 :laenge
if :laenge <5[stop]
pu  fd :laenge pd
fd :laenge back :laenge rt 90 fd :laenge back :laenge lt 90
unbekannt4 :laenge /2
pu back :laenge pd
fd :laenge back :laenge rt 90 fd :laenge back :laenge lt 90
unbekannt4 :laenge /2
pu rt 90 fd :laenge lt 90 pd
fd :laenge back :laenge rt 90 fd :laenge back :laenge lt 90
unbekannt4 :laenge /2
pu rt 90 back :laenge lt 90 pd
end
```

3. Aufgabe: a) Erläutere ausführlich, wie man mit Hilfe der Tastatur den Cursor auf dem Graphikbildschirm steuert.
b) Schreibe dazu die wichtigen Prozeduren und erläutere diese ausführlich.
c) Erläutere, wie beim Körner-Sammeln erreicht wird, dass die überfahrenen Körner auf dem Bildschirm verschwinden. Schreibe dazu ein Programm und erläutere die einzelnen Befehle.

,
4.Aufgabe: a) Erläutere den Befehl **setpos** (1. Angabe der Koordinaten durch Zahlen 2. Angabe der Koordinaten durch Variable)
.
b) Programmiere die nebenstehende Figur. Erstelle dazu eine Skizze des Fahrzeuges und beschrifte die Punkte mit den entsprechenden Koordinaten (der linke untere Punkt hat die Koordinaten (-400/10)). Schreibe dazu das entsprechende Programm. Das Fenster ist mit der Farbe Grün ausgefüllt. Für die Zeichnung muss der Befehl **setpos** verwendet werden.

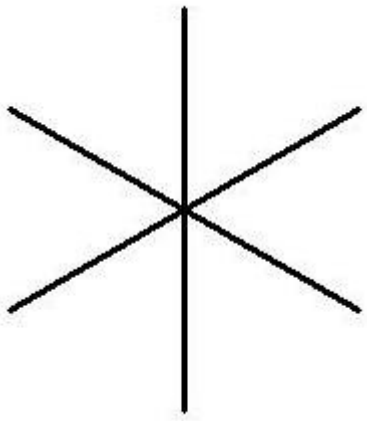


c) Die linke untere Ecke wird jetzt mit Hilfe zweier Variablen (XKoordinate und YKoordinate) angesprochen. Schreibe dazu das entsprechende Programm **LokomotiveVariable**. Der Aufruf sollte dann wie folgt erfolgen:
LokomotiveVariable -400 10

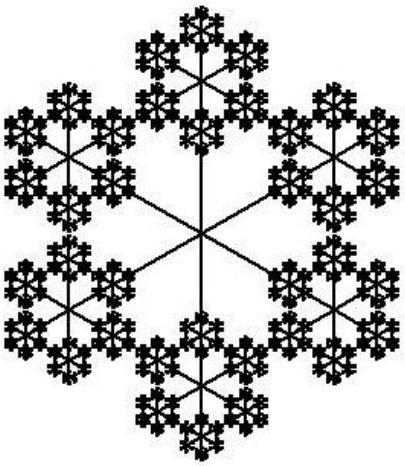
d) Die Lokomotive soll jetzt von links nach rechts bewegt werden. Erläutere das Verfahren, wie dies realisiert werden kann. Schreibe dazu die entsprechenden Prozeduren.

Loesung

1.Aufgabe: a)



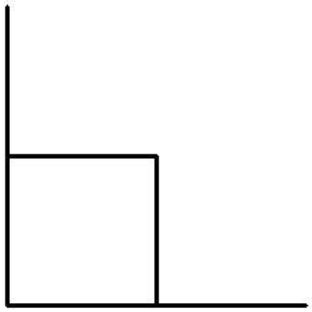
b)



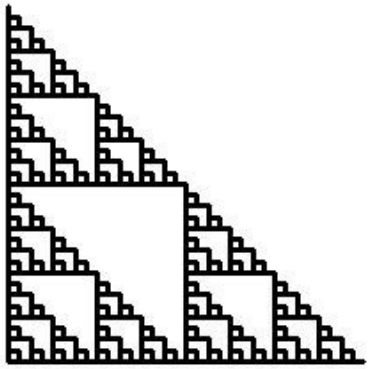
c) Rekursiv: Selbstaufruf, Veränderung des Übergabeparameters, Abbruchbedingung

```
to quadrat :laenge
if :laenge<0 [Stop]

repeat 4[fd :laenge lt 90]

quadrat :laenge-20
end
```

2.Aufgabe: a)



b)



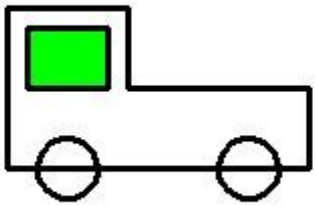
3. Aufgabe: a) und b)

```
to main
cs
make "keypress 12
setpensize[0 0]
fahren 1
end


to fahren :speed
setfocus [MSWLogo Screen]
keyboardon [make "keypress keyboardvalue steuerung :keypress keyboardoff]
if :keypress=113 [finish stop]
;;ASCI-Code: 113 -> q (Stoppt das Programm)
wait 2
pd
fd :speed
fahren :speed
end


to steuerung :richtung
if :richtung=44 [lt 10 stop]     ;;ASCI-Code: 44 -> , (Komma)
if :richtung=46 [rt 10 stop]     ;;ASCI-Code: 46 -> . (Punkt)
if :richtung=97 [make "speed :speed+1] ;;ASCI-Code: 97 -> a (beschleunigen)
if :richtung=115 [make "speed :speed-1] ;;ASCI-Code: 115 -> s (verlangsamen)
end


to main
cs
make "keypress 12
setpensize[0 0]
fahren 1
end


c) to kornweg
if pixel= [255 255 0] [setfc[255 255 255]fill]
make "anzahl :anzahl-1]
if :anzahl<1 [stop]
end
```

Beim Überfahren des Kreises wird dieser an der Stelle mit weiß übermalt.

4.Aufgabe: a) setpos bei Angabe der Koordinaten über konstante Werte mit eckigen Klammern: Beispiel -> setpos[-200 100]
Bei Angabe durch Variable mit runden Klammern: Beispiel -> setpos(list :xko :xko)
Wenn nur eine Koordinate als Variable angegeben wird, so muss ebenfalls mit runden Klammern geschrieben werden: Beispiel: -> setpos(list 0 :yko)

```
to lok
cs
setpensize[3 3]
setpc 0
pu
setpos[-400 10] pd
setpos[-250 10] setpos[-250 50] setpos[-340 50]
setpos[-340 90] setpos[-400 90] setpos[-400 10]
pu
setpos[-350 50] pd
setpos[-350 80] setpos[-390 80]
setpos[-390 50] setpos[-350 50]

pu
setpos[-380 70]  pd setfc 2 fill

pu
setpos[-370 10] pd circle 15

pu
setpos[-280 10] pd circle 15

ht
end
```



Mit Variable

```
to lokVariable :xko

setpc 0
pu
setpos(list :xko 10) pd
setpos(list :xko+150 10) setpos(list :xko+150 50) setpos(list :xko+60 50)
setpos(list :xko+60 90) setpos(list :xko 90) setpos(list :xko 10)
pu
setpos(list :xko+50 50) pd
setpos(list :xko+50 80) setpos(list :xko+10 80) setpos(list :xko+10 50)
setpos(list :xko+50 50)

pu setpos(list :xko+20 70) pd setfc 2 fill

pu setpos(list :xko+30 10) pd circle 15

pu setpos(list :xko+120 10) pd circle 15

ht
end
```

c) Die Bewegung wird dadurch erzeugt, dass die Figur erst in der gewünschten Farbe gezeichnet wird und dann die komplette Figur an derselben Stelle mit der Hintergrundfarbe (hier: weiß) gezeichnet wird.

```
to BewegungLok :xko
If :xko>300[lokVariable :xko stop]

lokVariable :xko
lokVariablewhite :xko
BewegungLok :xko+10

end


to mainLok
cs
make "xko -400
BewegungLok :xko

end
```